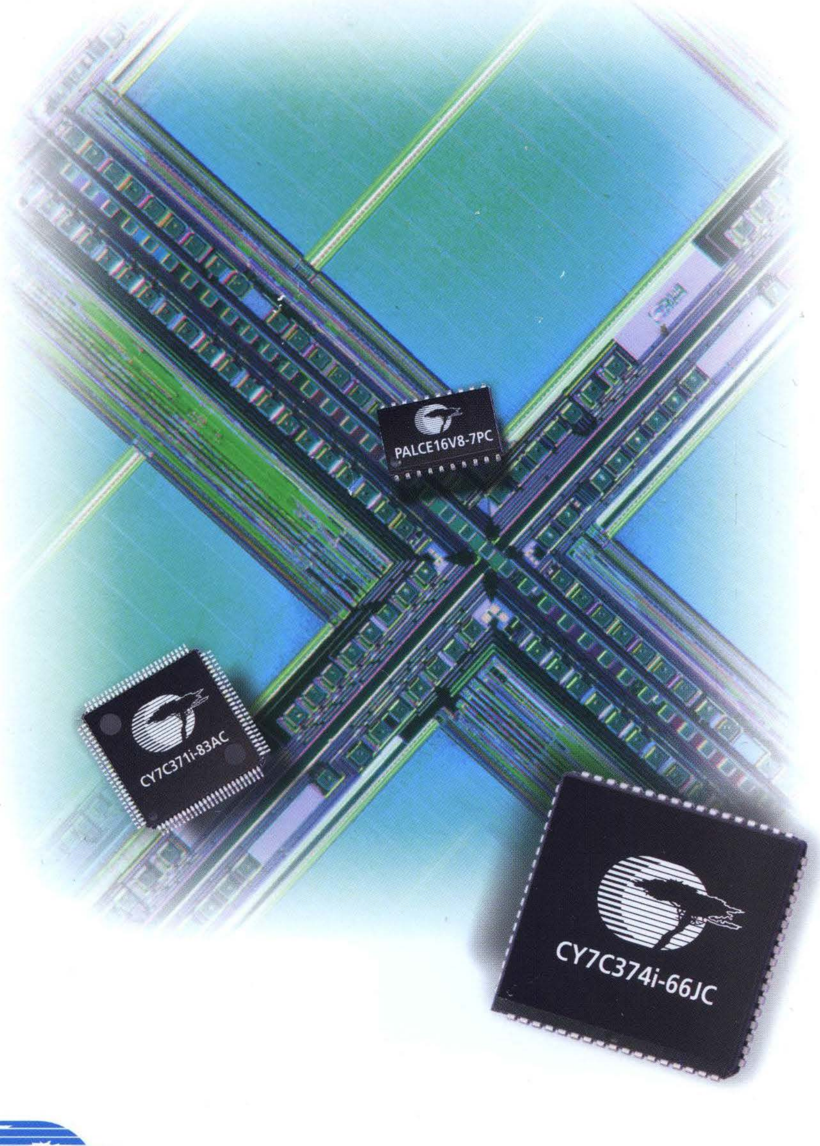
1997

CYPRESS PROGRAMMABLE LOGIC DATA BOOK

• PLDs • CPLDs • Tools

# CYPRESS PROGRAMMABLE LOGIC DATA BOOK

ULTRA LOGIC



CYPRESS

BANKS

CYPRESS

# Programmable Logic Data Book 1997

Cypress Semiconductor, 3901 North First Street, San Jose, CA 95134 (408) 943–2600
Telex: 821032 CYPRESS SNJ UD, TWX: 910 997 0753, FAX: (408) 943–2741
FAX-On-Demand: (800) 213–5120 or (408) 943–2798, Web Address: http://www.cypress.com
Literature Fulfillment/Advertisement Response: (800) 858–1810

# How To Use This Book

## Overall Organization

This book has been organized by product type, beginning with Product Information. The products are next, starting with Small PLDs, CPLDs, and Development Systems. A section containing Quality and Reliability information is next, followed by Package Diagrams.

## Recommended Search Paths

| To search by: | Use: |
|---|---|
| *Product line* | Table of Contents or flip through the book using the tabs on the right-hand pages. |
| *Size* | The Product Selector Guide in section 1. |
| *Numeric part number* | Numeric Device Index following the Table of Contents. |
| *Other manufacturer's part number* | The Cross Reference Guide in section 1. |

## Key to Waveform Diagrams

| | | |
|---|---|---|
| ▨ | = | Rising edge of signal will occur during this time. |
| ▧ | = | Falling edge of signal will occur during this time. |
| ☒ | = | Signal may transition during this time (don't care condition). |
| « | = | Signal changes from high-impedance state to valid logic level during this time. |
| » | = | Signal changes from valid logic level to high-impedance state during this time. |

Published May 1997

CYPRESS

# The Cypress Product Line

**Memory Products**

Static RAMs
SRAM Modules
Cache RAMs
Cache Modules

**Data Communications Products**

ATM Transceivers
SONET/SDH Serial Transceivers
Ethernet/Fast-Ethernet Transceivers and Repeaters
SMPTE-259M/DVB-ASI Controllers
HOTLink™ Transceiver
HOTLink Receiver
FIFOs
Dual Port SRAMs

**Computation Products**

Clock Generators/Synthesizers
Roboclock™ Programmable Clock-Skew Buffers
Roboclock™ Jr. Low-Skew Clock Buffers
Universal Serial Bus (USB) Controllers
PC Chipsets
FCT-T Logic
VMEbus

**Non-Volatile Memory**

EPROMs
PROMs

**Programmable Logic Products**

Industry Standard SPLDs
EPLDs and MAX® EPLDs
CPLDs
Compiler Synthesizer Tools

To request a free Cypress Semiconductor Data Book or Data Book CD-ROM, call 1-800-858-1810. For the most current information visit the Cypress website at http://www.cypress.com.

# TABLE OF CONTENTS

## Section 1 - General Information

Cypress Semiconductor Background ........ 1-1
Ordering Information ........ 1-5
Cypress Semiconductor Bulletin Board System (BBS) Announcement ........ 1-6
Product Selector Guide ........ 1-7
Product Line Cross Reference ........ 1-33
Cypress Military Products ........ 1-59

## Section 2 - Small PLDs

| | | |
|---|---|---|
| | Introduction to Cypress PLDs | 2-1 |
| PAL®C20 Series | Reprogrammable CMOS PALC 16L8, 16R8, 16R6, 16R4 | 2-7 |
| PALCE16V8 | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-8 |
| PALCE20V8 | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-18 |
| PLDC20G10B | CMOS Generic 24-Pin Reprogrammable Logic Device | 2-29 |
| PLDC20G10 | CMOS Generic 24-Pin Reprogrammable Logic Device | 2-29 |
| PLDC20RA10 | Reprogrammable Asynchronous CMOS Logic Device | 2-38 |
| PALCE22V10 | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-48 |
| PALC22V10 | Reprogrammable CMOS PAL® Device | 2-58 |
| PALC22V10B | Reprogrammable CMOS PAL® Device | 2-60 |
| PALC22V10D | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-62 |
| CY7C330 | CMOS Programmable Synchronous State Machine | 2-72 |
| CY7C331 | Asynchronous Registered EPLD | 2-73 |
| CY7C335 | Universal Synchronous EPLD | 2-87 |

## Section 3 - CPLDs

| | | |
|---|---|---|
| Ultra37000™ ISR™ CPLD Family | UltraLogic™ High-Performance CPLDs | 3-1 |
| Ultra37128 | UltraLogic™ 128-Macrocell ISR™ CPLD | 3-8 |
| Ultra37128V | UltraLogic™ 3.3V 128-Macrocell ISR™ CPLD | 3-21 |
| Ultra37256 | UltraLogic™ 256-Macrocell ISR™ CPLD | 3-34 |
| Ultra37256V | UltraLogic™ 256-Macrocell 3.3V ISR™ CPLD | 3-47 |
| FLASH370i™ ISR™ CPLD Family | UltraLogic™ High-Density Flash CPLDs | 3-60 |
| CY7C371i | UltraLogic™ 32-Macrocell Flash CPLD | 3-68 |
| CY7C372i | UltraLogic™ 64-Macrocell Flash CPLD | 3-77 |
| CY7C373i | UltraLogic™ 64-Macrocell Flash CPLD | 3-86 |
| CY7C374i | UltraLogic™ 128-Macrocell Flash CPLD | 3-96 |
| CY7C375i | UltraLogic™ 128-Macrocell Flash CPLD | 3-107 |
| FLASH370™ CPLD Family | UltraLogic™ High-Density FLASH CPLDs | 3-119 |
| CY7C371 | UltraLogic™ 32-Macrocell Flash CPLD | 3-126 |
| CY7C372 | UltraLogic™ 64-Macrocell Flash CPLD | 3-127 |
| CY7C373 | UltraLogic™ 64-Macrocell Flash CPLD | 3-128 |
| CY7C374 | UltraLogic™ 128-Macrocell Flash CPLD | 3-129 |
| CY7C375 | UltraLogic™ 128-Macrocell Flash CPLD | 3-130 |
| CY7C340 EPLD Family | Multiple Array Matrix High-Density EPLDs | 3-131 |
| CY7C341B | 192-Macrocell MAX® EPLD | 3-137 |
| CY7C341 | 192-Macrocell MAX® EPLD | 3-151 |
| CY7C342B | 128-Macrocell MAX® EPLDs | 3-153 |

**Section 3 - CPLDs** (continued)

CY7C342 128-Macrocell MAX® EPLDs ... 3-171

CY7C343/CY7C343B 64-Macrocell MAX® EPLD ... 3-172

CY7C344/CY7C344B 32-Macrocell MAX® EPLD ... 3-189

CY7C346/CY7C346B 128-Macrocell MAX® EPLDs ... 3-203

**Section 4 - Development Tools**

PLD Development Tools Overview ... 4-1

CY3120 Warp2® VHDL Compiler for PLDs ... 4-2

CY3122/CY3127 *Warp2*Sim™ VHDL Development System for PLDs ... 4-7

CY3130/CY3135 *Warp3*® VHDL Development System for PLDs ... 4-12

Third-Party Tool Support ... 4-18

CY3140 ABEL™/Synario™ Design Kit for FLASH370i™ ... 4-26

CY3144 Cypress Mentor Graphics Bolt-in Kit ... 4-27

PLD Programming Information ... 4-28

CY3500 *Impulse3* ™ DeviceProgrammer and Adapters ... 4-30

CY3600 InSRkit™: ISR™ Programming Kit ... 4-33

CY3620 *Warp2*ISR™ ISR Design Kit for CPLDs ... 4-35

**Section 5 - Application Notes**

Are Your PLDs Metastable? ... 5-1

Designing with the CY7C335 and *Warp2*® VHDL Compiler ... 5-23

Getting Started Converting .ABL Files to VHDL ... 5-51

Abel™-HDL vs. IEEE-1076 VHDL ... 5-74

State Machine Design Considerations and Methodologies ... 5-85

Efficient Arithmetic Designs Targeting FLASH370i™ CPLDs ... 5-115

The FLASH370i™ Family Of CPLDs and Designing with *Warp2*® ... 5-139

Implementing a Reframe Controller for the CY7B933 HOTLink™ Receiver in a CY7C371 CPLD ... 5-157

Implementing a 128Kx32 Dual-Port RAM Using the FLASH370™ ... 5-169

An Introduction to In System Reprogramming with FLASH370i™ ... 5-180

Designing With FLASH370i™ for PC Cable Programming ... 5-186

Using Hierarchy in VHDL Design ... 5-198

**Section 6 - Quality**

Quality, Reliability, and Process Flows ... 6-1

Moisture-Sensitive Devices Handling Information ... 6-16

Thermal Management and Component Reliability ... 6-18

Tape and Reel Specifications ... 6-28

**Section 7 - Packages** ... 7-1

## NUMERIC DEVICE INDEX

| | | |
|---|---|---|
| CY3120 | *Warp2*® VHDL Compiler for PLDs | 4-2 |
| CY3122 | *Warp2*Sim™ VHDL Development System for PLDs | 4-7 |
| CY3127 | *Warp2*Sim™ VHDL Development System for PLDs | 4-7 |
| CY3130 | *Warp3*® VHDL Development System for PLDs | 4-12 |
| CY3135 | *Warp3*® VHDL Development System for PLDs | 4-12 |
| CY3140 | ABEL™/Synario™ Design Kit for FLASH370i™ | 4-26 |
| CY3144 | Cypress Mentor Graphics Bolt-in Kit | 4-27 |
| CY3500 | *Impulse3*™ DeviceProgrammer and Adapters | 4-30 |
| CY3600 | InSRkit™: ISR™ Programming Kit | 4-33 |
| CY3620 | *Warp2*ISR™ ISR Design Kit for CPLDs | 4-35 |
| CY7C330 | CMOS Programmable Synchronous State Machine | 2-72 |
| CY7C331 | Asynchronous Registered EPLD | 2-73 |
| CY7C335 | Universal Synchronous EPLD | 2-87 |
| CY7C340 EPLD Family | Multiple Array Matrix High-Density EPLDs | 3-131 |
| CY7C341 | 192-Macrocell MAX® EPLD | 3-151 |
| CY7C341B | 192-Macrocell MAX® EPLD | 3-137 |
| CY7C342 | 128-Macrocell MAX® EPLDs | 3-171 |
| CY7C342B | 128-Macrocell MAX® EPLDs | 3-153 |
| CY7C343 | 64-Macrocell MAX® EPLD | 3-172 |
| CY7C343B | 64-Macrocell MAX® EPLD | 3-172 |
| CY7C344 | 32-Macrocell MAX® EPLD | 3-189 |
| CY7C344B | 32-Macrocell MAX® EPLD | 3-189 |
| CY7C346 | 128-Macrocell MAX® EPLDs | 3-203 |
| CY7C346B | 128-Macrocell MAX® EPLDs | 3-203 |
| CY7C371 | UltraLogic™ 32-Macrocell Flash CPLD | 3-126 |
| CY7C371i | UltraLogic™ 32-Macrocell Flash CPLD | 3-68 |
| CY7C372 | UltraLogic™ 64-Macrocell Flash CPLD | 3-127 |
| CY7C372i | UltraLogic™ 64-Macrocell Flash CPLD | 3-77 |
| CY7C373 | UltraLogic™ 64-Macrocell Flash CPLD | 3-128 |
| CY7C373i | UltraLogic™ 64-Macrocell Flash CPLD | 3-86 |
| CY7C374 | UltraLogic™ 128-Macrocell Flash CPLD | 3-129 |
| CY7C374i | UltraLogic™ 128-Macrocell Flash CPLD | 3-96 |
| CY7C375 | UltraLogic™ 128-Macrocell Flash CPLD | 3-130 |
| CY7C375i | UltraLogic™ 128-Macrocell Flash CPLD | 3-107 |
| FLASH370™ CPLD Family | UltraLogic™ High-Density Flash CPLDs | 3-119 |
| FLASH370i™ ISR™CPLD Family | UltraLogic™ High-Density Flash CPLDs | 3-60 |
| PALC20 Series | Reprogrammable CMOS PALC 16L8, 16R8, 16R6, 16R4 | 2-7 |
| PALC22V10 | Reprogrammable CMOS PAL® Device | 2-58 |
| PALC22V10B | Reprogrammable CMOS PAL® Device | 2-60 |
| PALC22V10D | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-62 |
| PALCE16V8 | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-8 |
| PALCE20V8 | Flash Erasable, Reprogrammable CMOS PAL, Device | 2-18 |
| PALCE22V10 | Flash Erasable, Reprogrammable CMOS PAL® Device | 2-48 |
| PLDC20G10 | CMOS Generic 24-Pin Reprogrammable Logic Device | 2-29 |
| PLDC20G10B | CMOS Generic 24-Pin Reprogrammable Logic Device | 2-29 |
| PLDC20RA10 | Reprogrammable Asynchronous CMOS Logic Device | 2-38 |
| Ultra37000™ ISR™ CPLD Family | UltraLogic™ High-Performance CPLDs | 3-1 |
| Ultra37128 | UltraLogic™ 128-Macrocell ISR™ CPLD | 3-8 |
| Ultra37128V | UltraLogic™ 3.3V 128-Macrocell ISR™ CPLD | 3-21 |
| Ultra37256 | UltraLogic™ 256-Macrocell ISR™ CPLD | 3-34 |
| Ultra37256V | UltraLogic™ 256-Macrocell 3.3V ISR™ CPLD | 3-47 |

CYPRESS

**General Information** 1

Small PLDs 2

CPLDs 3

Development Tools 4

Application Notes 5

Quality 6

Packages 7

Table of Contents Page Number

## General Information

Cypress Semiconductor Background ........ 1-1
Ordering Information ........ 1-5
Cypress Semiconductor Bulletin Board System (BBS) Announcement ........ 1-6
Product Selector Guide ........ 1-7
Product Line Cross Reference ........ 1-33
Cypress Military Products ........ 1-59

CYPRESS

# Cypress Semiconductor Background

Cypress Semiconductor Corporation is in its second decade as an international, broad-line manufacturer and supplier of integrated circuits for a range of growth markets. The company supplies its products to leading providers of data communications, telecommunications, personal computer, and military systems worldwide.

Cypress was founded in 1982 and has grown rapidly and profitably. In 1990, citing its successful innovation, execution, and leadership, Electronic Business magazine named Cypress the fastest-growing electronics company in America. The company has received honors for excellence in financial management, including three Bronze and two Silver awards for financial performance from *The Wall Street Transcript*. Cypress is led by its founder, president and CEO, T. J. Rodgers, whose aggressive, visionary management style has gained national prominence for the company. Cypress's sales and marketing efforts are directed by Dan McCranie, a 20-year industry veteran.

Today Cypress is an international company with over 2,000 employees and a worldwide sales network. Corporate headquarters are in San Jose, California. Cypress maintains a site on the worldwide web at http://www.cypress.com. Cypress enjoys a competitive advantage by continuously improving the cost-efficiency and productivity of its operations, and by implementing cycle-time and inventory-reduction programs. These improvements enable Cypress to compete in high-volume, cost-sensitive markets, including those for EPROMs, FCT logic, and personal computer cache memories, increasing sales opportunities and driving the company's rapid growth.

Cypress has made rapid strides in other fast-growing markets, entering the high-density programmable logic arena with its UltraLogic™ product family and data communications with leading-edge physical-layer products for emerging networking technologies, including ATM (Asynchronous Transfer Mode), Fibre Channel, and Fast Ethernet. Cypress also offers the hyperCache™ chipset for Pentium®-class personal computers, the first core logic chipset to incorporate cache memory.

Cypress operates four advanced wafer fabrication facilities, with another facility under construction. Its manufacturing sites include:

- Fab I in San Jose, California, the company's first fab, the focal point for research and development.
- Fab II in Round Rock, Texas, the company's largest wafer fabrication plant, founded in 1986.
- Fab III in Bloomington, Minnesota, founded in 1991.
- Fab IV, adjacent to Fab III in Bloomington. Fab IV is the company's first 8-inch wafer fab.
- Fab V, adjacent to Fab II, under construction and tentatively slated to begin production in 1998.

Cypress Semiconductor Philippines Inc. (CSPI), a state-of-the-art assembly and test facility, opened in the third quarter of 1996, complementing an existing assembly and test facility in Bangkok, Thailand. Cypress also has expanded its global design capacity, adding to its original Silicon Valley design center facilities in Mississippi, Colorado, Washington, Oregon, Minnesota, Texas, England, India, and Germany. The Philippines assembly plant has a design facility, and Cypress is forming a team for an anticipated Irish design center. With top design talent increasingly difficult to recruit, the new international base casts a wider net, reaching out to rapidly growing technology centers in Europe, Asia, and the Pacific Rim.

Cypress was incorporated in California in 1982 and went public in May 1986. Cypress's stock is listed on the New York Stock Exchange, trading under the symbol "CY."

Cypress's 1983 business plan outlined a strategy that propelled the company's fast growth over its first 10 years. It stated, "The basic premise of Cypress is that a multi-disciplinary group of founders can quickly put into production a state-of-the-art, high-speed CMOS [Complementary Metal Oxide Semiconductor] process. This technology will be applied to a targeted group of high-speed, high-average-selling-price products that will be outperformed significantly by Cypress's new CMOS pin-compatible circuits in every measure of IC performance: speed, power consumption, yield (cost), quality, and reliability."

Cypress consistently has outperformed the competition. The company attained profitability in 1985, just seven quarters of production after the introduction of its initial, flagship product, a 1-Kbit, 15-nanosecond SRAM. Cypress went public in 1986, attracting $77.6 million, at that time one of the largest initial public offerings. By year-end 1987, Cypress had recovered its start-up operating costs and showed a positive cash balance. The following year, Cypress topped the $100 million mark in revenues, eclipsing $200 million in 1990, $400 million in 1994, and currently approaching $600 million.

Cypress achieves strong results through technological innovation, solid management and execution. From the start, Cypress's groundbreaking CMOS technology outperformed competitive products that used far more power. Never satisfied, the company has become an industry leader through the introduction of newer and faster processing technologies. In 1987, Cypress implemented one of the first 0.8-micron CMOS process technologies, continuing its migration with an 0.65-micron CMOS process in 1992, and the development of world-class BiCMOS and Flash technologies. Cypress maintains its track record of innovation and leadership with the production of 0.5-micron CMOS and BiCMOS products, and the recent launch of its low-power, high-performance 0.5-micron RAM3™ process technology.

Cypress has achieved excellence in its first decade. The company's retained earnings (net income from past periods that has been retained by the business) ranks among the top U.S.

semiconductor companies. Cypress has fostered a strong, loyal customer base, built through years of delivering high-quality products. The company maintains strong ties with the academic community, donating equipment, software and technical expertise to target schools under its University Program. Cypress also has built one of the industry's strongest research and development organizations, positioning it to maintain technology leadership and to respond quickly to changing market conditions.

Over the past decade, the semiconductor market has been marked by great change and volatility. Electronics are an integral part of almost every facet of the economy, and the proliferation of computer and communications equipment has prompted phenomenal industry growth, driving down prices and sharpening competition.

In response, Cypress has adjusted its original strategy of focusing almost exclusively on high-end, niche markets. Now the company's strategy is to compete in higher-volume markets, maintaining a competitive cost structure and quickly introducing market-leading products. Cypress has taken a number of steps to support this new strategy. Product assembly and testing was moved offshore, resulting in cost savings of more than $100 million in 1995. Cypress has significantly reduced the number of product/package types and process technologies used in production, resulting in more cost-effective manufacturing. Other changes include a cycle-time reduction program and a renewed focus on improving manufacturing yields.

Competitive manufacturing costs allow Cypress to adopt a "no excuses" strategy for competition in high-volume markets. Implicit in this strategy is the desire to compete profitably in commodity products at any volume, at any competitive price, anywhere in the world. Cypress also will use the process development and yield improvements gained from doing business in high-volume areas to maximize its advantage in other product lines.

Cypress has shifted to a market-oriented approach in recent years, serving high-growth, profitable end-user markets such as data telecommunications and computation with focused product lines. For example, the acquisition of IC Designs in 1993 provides Cypress with more products developed specifically for the personal computer market. In the fast-growing data communications market, Cypress has designed products specifically for Ethernet, Fibre Channel, and ATM networking applications.

Cypress is fast approaching its billion-dollar financial goal while maintaining profitable growth. Cypress has the right products in the right markets, with solid positioning in three of the largest semiconductor markets—personal computers, networking, and communications. It is an acknowledged leader in an industry poised to embark upon a period of breathtaking growth. Consider that the annual average sales growth rate during the two worst five-year periods in the chip business was 10 percent, and that the industry has averaged 17 percent growth per year for 35 years. Even if average sales growth during the current half-decade falls back to 10 percent, the industry's dollar volume still will have grown to $232 billion in 2000 from $144 billion in 1995. Another positive sign is the chip business's rising share of world electronic sales.

Cypress has built a reputation throughout the industry and with its customers for providing high performance, high-quality products in every market it enters. It continues to bring to market new, leading-edge products, based on Cypress's CMOS, BiCMOS, and Flash process technologies. Cypress has 0.8-, 0.65-, and 0.5-micron CMOS and BiCMOS processes, and a 0.65-micron Flash process. These process technologies allow Cypress to offer state-of-the-art products, providing the optimal balance of speed and power use for any system.

Cypress also offers a broad range of packaging options for its products, giving customers a variety of choices in pinout configurations and temperature grades. In addition, Cypress products are designed to meet or exceed the full temperature and functional requirements of military products. This means that Cypress builds military products as a matter of course, rather than having to redesign to meet rigorous military specifications. The company received full military Qualified Manufacturer List (QML) certification in 1996.

Segmented into five divisions, Cypress offers products that are mainstays in a broad range of industries and markets. They are used in personal computers, workstations, servers, and super-minicomputers; telecommunications; data communications; networking products; military applications; and test and measurement equipment.

## Static Random Access Memories

Cypress is a market-leading supplier of SRAMs, providing a wide range of products for leading companies worldwide. SRAMs, which comprise the company's largest product segment, are used in high-performance personal computers, workstations, telecommunications systems, industrial systems, instrumentation devices, and networking products. Cypress's low production costs allow the company to compete effectively in high-volume markets for SRAMs, including data communications and telecommunications, and cache RAMs to support today's high-performance microprocessors such as Pentium and PowerPC. Cypress's patented RAM3 architecture produces the industry's best low-voltage products with high alpha immunity, making them ideal for mobile applications that require extended battery life, and for datacom and telecom applications demanding exceptional data reliability.

Among the first RAM3 products are the CY7C1335 3.3-volt 32K x 32 synchronous-pipelined SRAM for Pentium cache, and the CY7C1337, a companion product with 2.5-volt I/O for Pentium Pro® processors. Cypress also is rolling out a 1-megabit SRAM designed for the military with 15 ns access and a 64K x 8 SRAM targeted specially for disk drive applications.

- Multichip Modules—Modules are semiconductor chips mounted on tiny computer circuit boards. They rely on innovative surface-mount technology, minimizing necessary board space. Multichip-module technology allows engineers to design systems using integrated circuits a generation ahead of current production technology, simply plugging a new chip into an existing multichip-module socket. This allows companies to bring products to market more quickly, offering them a competitive advantage. Cache modules for personal computers are the mainstay of this product line; Cypress supplies modules for many of the leading PC manufacturers worldwide.

## Programmable Logic Devices

With increasing pressure on systems designers to bring products to market more quickly, the demand for programmable logic devices (PLDs) is surging, particularly in the communications and consumer-electronics businesses. PLDs are logic-control devices that can be easily programmed by engineers in the field and later erased and reprogrammed. This allows designers to make key systems changes late in the development cycle.

- UltraLogic—Cypress's UltraLogic product line addresses the high-density programmable logic market. UltraLogic includes the highest-performance complex PLDs, the FLASH370i™ family. This product family is supported by Cypress's VHDL (Very high-speed integrated circuit Hardware Description Language) based *Warp*™, the industry's most advanced software design tool. Cypress pioneered the use of VHDL for PLD programming, and *Warp* software is a key factor in the company's success in the PLD market. A textbook by Cypress applications engineer Kevin Skahill, "VHDL for Programmable Logic," is gaining acceptance in universities worldwide.
- Small PLDs—Cypress is a leading provider of small PLDs with a wide range of offerings. The company is committed to competing in all segments of the PLD market, including small, industry-standard devices such as the 16V8, 20V8, and 22V10; along with CPLDs. Similarly, Cypress's selection of tools and software for PLD programming is among the broadest in the business.

## Non-Volatile Memory

Non-volatile memory stores information even when power is turned off. It is used to store program code for a variety of applications, including computers, peripherals, and communication devices. Cypress is a long-term supplier of high-speed, non-volatile memory, and is continuing to add new products to its portfolio, demonstrating its strong commitment to the market.

- Programmable Read-Only Memories—Cypress offers a wide range of CMOS PROMs and EPROMs (Erasable PROMs). Like PLDs, they are programmable (some of them are reprogrammable) expediting the design process. Cypress owns a large share of the high-speed CMOS PROM market, and now competes in the mainstream EPROM market.

## Data Communications Division

Cypress's presence in the fast-growing datacom business underscores its new market-driven orientation. The company supports high-speed data communications with a range of products from the physical-connection layer to system-level solutions.

- Specialty Memory—Cypress produces a variety of First-In/First-Out (FIFO) memories, used to pass data between systems operating at different frequencies, and Dual-Port memories, used to distribute data to two different systems simultaneously.
- ATM and Fibre Channel Devices—HOTLink™ high-speed, point-to-point serial communications chips have been well received. HOTLink, along with the SONET/SDH Serial Transceiver (SST™), addresses the fast-growing ATM and Fibre-Channel communications markets.
- Ethernet Devices—Cypress has moved to support the Ethernet market with the introduction of the CY7C971 100BaseT-4 Fast Ethernet Transceiver and the CY7B8392 transceiver for coaxial cable Ethernet networks. These products address the fastest-growing networking standards with leading-edge solutions while remaining compatible with slower transmission protocols.
- RoboClock™—The data communications division also produces RoboClock, a programmable skew clock buffer that adjusts complex timing control signals for a broad range of systems.

## Computation Products Division

Focused on the high-volume, high-growth desktop-computer market, computation products is the second of Cypress's market-oriented divisions. It offers timing technology products used widely in personal computers and disk drives, as well as PC chipsets and Universal Serial Bus microcontroller.

- Frequency Synthesis—Cypress clock oscillators control the intricate timing of all aspects of a computer system, including signals for the computer's central processing unit (CPU), keyboard, disk drives, system bus, serial port, and real-time clock. They replace all of the metal-can oscillators used in the system. Cypress also has introduced a family of programmable clock generators that offer a close-to-universal solution for any system requiring multiple clocks. EPROM-programmable clock chips allow designers to choose custom frequencies, cutting design time and inventory costs.
- PC Chipsets—Cypress entered the PC chipset market with the 1994 acquisition of Contaq Microsystems and recently began selling the hyperCache Chipset for Pentium-class PCs. hyperCache is the industry's most complete chipset, integrating keyboard and mouse control, real-time clock, and local-bus enhanced IDE control. It also provides integrated second-level cache. Appropriate for high-speed personal computers, hyperCache offers a high-performance solution for other markets, as well, including industrial control, medical equipment, telephone-switching equipment, test and measurement, and game machines.
- Universal Serial Bus—USB, a technology developed by Microsoft and Intel, allows various PC peripherals to communicate with the host system via a single interface, eliminating cable clutter and offering easy connection of peripherals devices. Currently supported by more than 250 computer, telecommunication, and software companies, USB is expected to ship in high volumes beginning in 1997, and Cypress aims to be among the first companies to offer a line of USB products, beginning with microcontrollers for mice, joysticks, and keyboards.
- FCT Logic Devices—FCT (Fast CMOS Technology) logic devices are widely used by designers implementing bus interface and standard logic functions in high-speed systems. Cypress now offers over 50 standard logic and bus interface functions in several versions, FCT-T (TTL compatible outputs) and FCT2 (outputs with built-in series resistors). This broad product family-manufactured on Cypress's high-volume, CMOS manufacturing lines-offers a cheaper, cooler alternative to competing BiCMOS solutions. The FCT logic product line combines with Cypress's high-performance SRAM, PLD, data communications, PROM, and timing technology devices to offer total solu-

tions for designers of ultrafast computer and communications systems.

Cypress recognizes the importance of bringing new, leading-edge products to market on a consistent basis. The company's "Top 10 Program" tracks the most important new products through the production pipeline. The development of each product is supported by a cross-functional team of engineering, marketing, and production specialists, with oversight and support provided by a project "godfather," usually a vice president but in some cases CEO T. J. Rodgers. Through the Top 10 Program, Cypress has introduced products including HOTLink, the Flash 22V10 SPLD, RoboClock, and many others.

The cornerstone of Cypress's success is the very high quality of its products, services, and people. Cypress has been honored for product and service excellence by companies including AT&T, Unisys, Nortel, and Raytheon. Cypress has received the coveted "STACK" Level II certification, awarded to companies whose products meet tough standards for quality. The company also has received ISO 9000 registration, awarded to companies with exacting standards of quality management, production, and inspections. In 1996, the Defense Electronics Supply Center (DESC) awarded QML certification to Cypress's military offerings.

Cypress's corporate structure reinforces quality production, maintaining only the highest level of hiring practices, performance benchmarks, and individual employee standards. The result is a company that consistently delivers quality products, services, and financial results.

Cypress is well known for its effective, highly automated management systems, which are used companywide in areas such as manufacturing, purchasing, order-entry, and weekly goal tracking for all employees. Cypress's solid corporate leadership has been another important factor in the company's success. T. J. Rodgers, president and CEO, who founded the company in 1982, received his Ph.D. in electrical engineering from Stanford University in 1975. Citing Rodgers's technical expertise and progressive management, The Wall Street Journal characterized Cypress as "a quintessential entrepreneurial company." Cypress has received numerous awards for excellence in financial management, including a Kachina Award from the market-research company, In-Stat Inc., and an Encore Award from the Stanford University Business School as Entrepreneurial Company of the Year for 1988.

Dan McCranie, vice president of sales and marketing, provides exceptional leadership, drawing on more than seven years of experience as CEO for a major semiconductor manufacturer. Chief Financial Officer Manny Hernandez brings strong financial leadership with nearly 20 years of experience in a broad range of financial positions in the semiconductor industry. Jim Kupec, who started at Cypress in 1983-working in R&D, product engineering, and operations—is vice president of product divisions.

Cypress's culture is a product of the company's sophisticated technology and manufacturing. Operating its own wafer manufacturing plants, Cypress offers its customers not only reliable, high-quality semiconductors, but, the benefits of an innovative, cost-efficient, quality-driven culture.

Cypress goes to great lengths to hire and to keep the best people available. All employees are granted stock options and thereby participate in the success of the company. Also, each employee is eligible for quarterly profit sharing bonuses, based on corporate targets.

Cypress trusts its employees to make important decisions with a minimum of bureaucracy. In fact, each employee bears responsibility for achieving goals known as "critical success factors," which are designed to advance the company's strategic plan. Cypress's automated goals system helps employees to maintain focus, pushing decision-making down in the ranks. Such empowerment helps Cypress to achieve revenue-per-employee figures that are among the industry's best.

Cypress is a company that encourages individuals to do what it takes to get the job done, provides them with the proper tools to achieve these objectives, and rewards them for their efforts. These individuals have made, and continue to make Cypress successful.

CYPRESS

# Ordering Information

In general, the ordering codes for products follow the format below; e.g., CY7C128-45DMB, PALC16R8L-35PC

## PAL & PLD

| PREFIX | DEVICE | SUFFIX | FAMILY |
|---|---|---|---|
| PAL C | 16R8 | -25 L M B | PAL 20 |
| PAL C | 16R8 | L-35 P C | LOW POWER PAL 20 |
| PAL C | 22V10 | -25 W C | PAL 24 VARIABLE PRODUCT TERMS |
| PAL CE | 16V8 | -25 P C | FLASH-ERASABLE PAL20 |
| PLD C | 20G10 | -25 W C | GENERIC PLD 24 |
| CY | 7C335 | -83 P C | UNIVERSAL SYNCHRONOUS EPLD |
| CY | 7C374i | -100 J C | FLASH-ERASABLE CPLD |

PROCESSING
B = MIL-STD-883C FOR MILITARY PRODUCT
= LEVEL 2 PROCESSING FOR COMMERCIAL PRODUCT
T = SURFACE-MOUNTED DEVICES TO BE TAPE AND REELED
R = LEVEL 2 PROCESSING ON TAPE AND REELED DEVICES

TEMPERATURE RANGE
C = COMMERCIAL (0°C TO +70°C)
I = INDUSTRIAL (–40°C TO +85°C)
M = MILITARY (–55°C TO +125°C)

PACKAGE
A =THIN QUAD PLASTIC FLATPACK (TQFP)
B =PLASTIC PIN GRID ARRAY (PPGA)
D =CERAMIC DUAL IN-LINE PACKAGE (CERDIP)/BRAZED DIP
E =TAPE AUTOMATED BONDING (TAB)
F =FLATPACK (SOLDER-SEALED FLAT PACKAGE)
G =PIN GRID ARRAY (PGA)
H =WINDOWED LEADED CHIP CARRIER
J =PLASTIC LEADED CHIP CARRIER (PLCC)
K =CERPACK (GLASS-SEALED FLAT PACKAGE)
L =LEADLESS CHIP CARRIER (LCC)
N =PLASTIC QUAD FLATPACK (PQFP)
P =PLASTIC DUAL IN-LINE (PDIP)
Q =WINDOWED LEADLESS CHIP CARRIER (LCC)
Q =QUARTER SIZE OUTLINE PACKAGE (for PALCE16V8 and PALCE20V8 only)
R =WINDOWED PIN GRID ARRAY (PGA)
S =SOIC (GULL WING)
T =WINDOWED CERPACK
U =CERAMIC QUAD FLATPACK (CQFP)
V =SOIC (J LEAD)
W=WINDOWED CERAMIC DUAL IN-LINE PACKAGE (CERDIP)
X =DICE (WAFFLE PACK)
Y =CERAMIC LEADED CHIP CARRIER
BG=BALL GRID ARRAY

SPEED (ns or MHz)

L = LOW-POWER OPTION
B, D = REVISION LEVEL
V = 3,3V LOW-POWER OPTION

**Cypress FSCM #65786**

# Cypress Semiconductor Bulletin Board System (BBS) Announcement

Cypress Semiconductor supports a 24-hour electronic Bulletin Board System (BBS) that allows Cypress Applications to better serve our customers by allowing them to transfer files to and from the BBS.

The BBS is set up to serve in multiple ways. One of its purposes is to allow customers to receive the most recent versions of Cypress programming software. Another is to allow the customers to send PLD programming files that they are having trouble with to the BBS. Cypress Applications can then find the errors in the files, correct them, and place them back on the BBS for the customer to download. The customer may also ask questions in our open forum message area. The sysop (system operator) will forward these questions to the appropriate applications engineer for an answer. The answers then get posted back into the forum.

### Communications Set-Up

The BBS uses USRobotics HST Dual Standard modems capable of 14.4-Kbaud rates without compression and rates upwards of 19.2-Kbaud with compression. It is compatible with CCITT V.32 bis, V.32, V.22 (2400-baud), Bell 212A (1200-baud), CCITT V.42, and CCITT V.42 bis. It also handles MNP levels 2, 3, 4, and 5.

To call the BBS, set your communication package parameters as follows:

Baud Rate: 1200 baud to 19.2 Kbaud. Max. is determined by your modem.
Data Bits: 8
Parity: None (N)
Stop Bits: 1

In the U.S. the phone number for the BBS is (408) 943-2954. In Japan the BBS number is 81-423-69-8220. In Europe the BBS number is 49-810-62-2675. These numbers are for transmitting data only.

If the line is busy, please retry at a later time. When you access the BBS, an initial screen with the following statement will appear:

```
  Rybbs Bulletin Board
```

After you choose the graphics format you want to use, the system will ask for your first and last name. If you are a first-time user, you will be asked a few questions for the purposes of registration. Otherwise you will be asked for your password, and then you will be logged onto the BBS, which is completely menu driven.

If you have any problems or questions regarding the BBS, please contact Cypress Applications at (408) 943-2821 (voice).

# Fast Static RAMs

### Organization/Density

| Density | X1 | X4 | X4 SIO | X8 | X9 | X16 | X18 | X32 |
|---|---|---|---|---|---|---|---|---|
| 4K | 7C147<br>2147 | 7C123<br>7C148<br>7C149<br>7C150<br>2148<br>2149 | 7C122<br>9122<br>93422 | | | | | |
| 16K | 7C167A | 7C168A | | 7C128A<br>6116 | | | | |
| 64K to 72K | 7C187 | 7C164<br>7C166 | | 7C185<br>6264 | 7C182 | | | |
| 256K to 288K | 7C197 | 7C194<br>7C195<br>7C196 | 7C191 | 7C199<br>7C1399<br>7C1399V<br>62256<br>62256V | 7C188 | | | |
| 512K | | | | 7C1512 | | 7C1020<br>7C1020V | | |
| 1M | 7C1007 | 7C106A/<br>7C1006 | | 7C109/<br>7C1009<br>62128 | | 7C1021<br>7C1021V | 7C1031 *<br>7C1331 *<br>7C1032 *<br>7C1332 * | 7C1335 +<br>7C1337 + |
| 2M | | | | | | | | 7C1329 +<br>7C1330 + |

+ Synchronous-pipelined

* Synchronous

# Fast Static RAMs

| Size | Organization | Operating Voltage | Part Number | Max. Speed (ns) | Packages |
|---|---|---|---|---|---|
| 64 | 16 x 4—Non-Inverting | 4.5 V - 5.5 V | CY27S07A | $t_{AA} = 35$ | L |
| 1K | 256 x 4 | 4.5 V - 5.5 V | CY7C122 | $t_{AA} = 15$ | D, P, S |
| 1K | 256 x 4 | 4.5 V - 5.5 V | CY7C123 | $t_{AA} = 7$ | P, V |
| 1K | 256 x 4 | 4.5 V - 5.5 V | CY9122/91L22 | $t_{AA} = 25$ | P |
| 1K | 256 x 4 | 4.5 V - 5.5 V | CY93422A/93L422A | $t_{AA} = 35$ | P |
| 4K | 4K x 1—CS Power-Down | 4.5 V - 5.5 V | CY7C147 | $t_{AA} = 25$ | P |
| 4K | 4K x 1—CS Power-Down | 4.5 V - 5.5 V | CY2147/21L47 | $t_{AA} = 35$ | P |
| 4K | 1K x 4—CS Power-Down | 4.5 V - 5.5 V | CY7C148 | $t_{AA} = 25$ | D, P |
| 4K | 1K x 4—CS Power-Down | 4.5 V - 5.5 V | CY2148 | $t_{AA} = 35$ | D, P |
| 4K | 1K x 4 | 4.5 V - 5.5 V | CY7C149 | $t_{AA} = 25$ | D, P |
| 4K | 1K x 4 | 4.5 V - 5.5 V | CY2149/21L49 | $t_{AA} = 35$ | D, P |
| 4K | 1K x 4—Separate I/O, Reset | 4.5 V - 5.5 V | CY7C150 | $t_{AA} = 10$ | D, P, S |
| 16K | 2K x 8—CS Power-Down | 4.5 V - 5.5 V | CY7C128A | $t_{AA} = 15$ | D, P, V |
| 16K | 2K x 8—CS Power-Down | 4.5 V - 5.5 V | CY6116A | $t_{AA} = 20$ | D, L |
| 16K | 16K x 1—CS Power-Down | 4.5 V - 5.5 V | CY7C167A | $t_{AA} = 15$ | P, V |
| 16K | 4K x 4 CS Power-Down | 4.5 V - 5.5 V | CY7C168A | $t_{AA} = 15$ | D, P, V |
| 64K | 8K x 8—CS Power-Down | 4.5 V - 5.5 V | CY7C185/185A | $t_{AA} = 15$ | D, L, P, V |
| 64K | 8K x 8—CS Power-Down | 4.5 V - 5.5 V | CY7C186/186A | $t_{AA} = 15$ | D, P, Z |
| 64K | 64K x 1—CS Power-Down | 4.5 V - 5.5 V | CY7C187/187A | $t_{AA} = 15$ | D, P, V |
| 64K | 16K x 4—CS Power-Down | 4.5 V - 5.5 V | CY7C164/164A | $t_{AA} = 15$ | D, P, V |

Note: Please contact a Cypress Representative for product availability.

## Fast Static RAMs(continued)

| Size | Organization | Operating Voltage | Part Number | Max. Speed (ns) | Packages |
|---|---|---|---|---|---|
| 64K | 16K x 4—Output Enable | 4.5 V - 5.5 V | CY7C166/166A | $t_{AA}$ = 15 | P, V |
| 72K | 8K x 9 | 4.5 V - 5.5 V | CY7C182 | $t_{AA}$ = 25 | P, V |
| 256K | 32K x 8—CS Power-Down | 4.5 V - 5.5 V | CY7C198 | $t_{AA}$ = 25 | L, P |
| 256K | 32K x 8—CS Power-Down | 4.5 V - 5.5 V | CY7C199 | $t_{AA}$ = 10 | D, L, P, S, V, Z |
| 256K | 32K x 8—CS Power-Down (3.3V) | 3.0 V - 3.6 V | CY7C1399 | $t_{AA}$ = 12 | V, Z |
| 256K | 32Kx8 CS Power-Down (3.0V) | 2.7V-3.3V | CY7C1399V | $t_{AA}$=12 | V, Z |
| 256K | 64K x 4—CS Power-Down | 4.5 V - 5.5 V | CY7C194 | $t_{AA}$ = 12 | D, P, V |
| 256K | 64K x 4—CS Power Down with OE | 4.5 V - 5.5 V | CY7C196 | $t_{AA}$ = 15 | L, P, V |
| 256K | 64K x 4—Separate I/O, Transparent Write | 4.5 V - 5.5 V | CY7C191 | $t_{AA}$ = 15 | V |
| 256K | 64K x 4—CS Power-Down w/ OE | 4.5 V - 5.5 V | CY7C195 | $t_{AA}$ = 12 | D, P, V |
| 256K | 256K x 1—CS Power-Down | 4.5 V - 5.5 V | CY7C197 | $t_{AA}$ = 12 | D, P, V |
| 288K | 32K x 9—CS Power-Down | 4.5 V - 5.5 V | CY7C188 | $t_{AA}$ = 15 | V |
| 512K | 64K x 8-CS Power-Down | 4.5 V - 5.5 V | CY7C1512 | $t_{AA}$ = 15 | V, Z |
| 512K | 32K x 16-Asynch | 4.5 V - 5.5 V | CY7C1020 | $t_{AA}$ = 10 | V, Z |
| 512K | 32K x 16-Asynch | 3.0 V - 3.6 V | CY7C1020V | $t_{AA}$ = 12 | V, Z |
| 1M | 64K x 18—Burst | 4.5 V - 5.5 V | CY7C1031 | $t_{CDV}$ = 7 (@ 0 pF) | J |
| 1M | 64K x 18—Burst | 4.5 V - 5.5 V | CY7C1032 | $t_{CDV}$ = 7 (@ 0 pF) | J |
| 1M | 128K x 8—CS Power-Down | 4.5 V - 5.5 V | CY7C109 / 1009 | $t_{AA}$ = 12 | D, L, V, Z |
| 1M | 256K x 4—CS Power-Down | 4.5 V - 5.5 V | CY7C106A / 1006 | $t_{AA}$ = 12 | V |
| 1M | 1M x 1—CS Power-Down | 4.5 V - 5.5 V | CY7C1007 | $t_{AA}$ = 12 | V |
| 1M | 64K x 16-Asynch | 4.5 V - 5.5 V | CY7C1021 | $t_{AA}$ = 12 | V, Z |
| 1M | 64K x 16-Asynch | 3.0 V - 3.6 V | CY7C1021V | $t_{AA}$ = 12 | V, Z |
| 1M | 32K x 32-Synch-Pipelined (3.3V) | 3.14 V - 3.6 V | CY7C1335 | $t_{CO}$ = 5.5ns | A |
| 1M | 32K x 32-Synch-Pipelined (2.5V) | 3.14 V - 3.46 V | CY7C1337 | $t_{CO}$ = 5.0ns | A |
| 2M | 64Kx32-Synch-Pipelined (3.3V) | 3.14V-3.46V | CY7C1329 | $t_{CO}$ = 5.5ns | A |
| 2M | 64Kx32-Synch-Pipelined (2.5V) | 3.14V-3.46V | CY7C1330 | $t_{CO}$ = 5.0ns | A |

Note: Please contact a Cypress Representative for product availability.

## Standard Speed SRAMs

| Size | Organization | Operating Voltage | Part Number | Power Grade | Max. ICC | Max CMOS Standby (ISB2) | Packages |
|---|---|---|---|---|---|---|---|
| 64K | 8Kx8 | 4.5V-5.5V | CY6264 | Std. | 100 mA | 15 mA | SN |
| 256K | 32Kx8 | 4.5V-5.5V | CY62256 | Std. | 55 mA | 5 mA | P, RZ, SN, Z |
| 256K | 32Kx8 | 4.5V-5.5V | CY62256 | L | 50 mA | 50 µA | P, RZ, SN, Z |
| 256K | 32Kx8 | 4.5V-5.5V | CY62256 | LL | 50 mA | 5 µA | RZ, SN, Z |
| 256K | 32Kx8 | 2.7V-3.6V | CY62256V | Std | 50 mA | 500 uA | RZ, SN, Z |
| 256K | 32Kx8 | 2.7V-3.6V | CY62256V | L | 50 mA | 50 µA | RZ, S, SN, Z |
| 256K | 32Kx8 | 2.7V-3.6V | CY62256V | LL | 30 mA | 5 µA | RZ, SN, Z |
| 256K | 32Kx8 | 2.3V-2.7V | CY62256V25 | L | 30 mA | 20 µA | SN, Z |
| 256K | 32Kx8 | 2.3V-2.7V | CY62256V25 | LL | 30 mA | 5 µA | SN, Z |
| 1M | 128Kx8 | 4.5V-5.5V | CY62128 | Std. | 110 mA | 25 mA | RZ, S, V, Z |
| 1M | 128Kx8 | 4.5V-5.5V | CY62128 | L | 70 mA* | 250 µA* | RZ, S, Z |
| 1M | 128Kx8 | 4.5V-5.5V | CY62128 | LL | 70 mA* | 30 µA* | RZ, S, Z |
| 1M | 128Kx8 | 2.7V-3.6V | CY62128V | Std. | 70 mA* | 1 mA* | S, Z |
| 1M | 128Kx8 | 2.7V-3.6V | CY62128V | L | 70 mA* | 200 µA* | S, Z |
| 1M | 128Kx8 | 2.7V-3.6V | CY62128V | LL | 70 mA* | 20 µA* | S, Z |

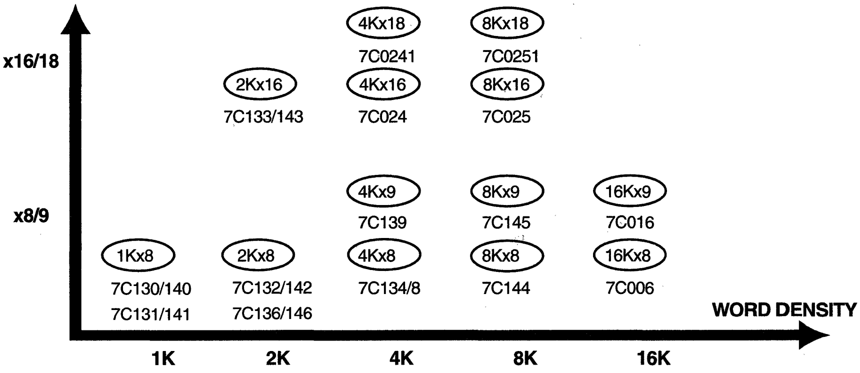
* Advanced Information

Note: Please contact a Cypress Representative for product availability.

## Dual-Port RAMs

- True Dual-Ported memory cells which allow simultaneous reads of the same memory location
- High speed access: 15ns
- Low power
- Automatic power down
- TQFP/PQFP/PLCC packaging available on all densities
- Various arbitration schemes available (Busy, Interrupt, Semaphore)

### Dual-Port RAM Product Portfolio

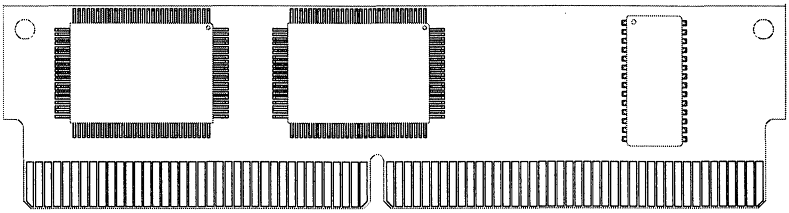


### Dual-Port RAMs

| Size | Organization | Pins | Part Number | Speed (ns) | ICC (mA @ ns) | Packages |
|---|---|---|---|---|---|---|
| 8K | 1Kx8—Dual-Port Master | 48 | CY7C130 | $t_{AA}$ = 25, 30, 35, 45, 55 | 170 @ 25 | P |
| 8K | 1Kx8—Dual-Port Slave | 48 | CY7C140 | $t_{AA}$ = 25, 30, 35, 45, 55 | 170 @ 25 | P |
| 8K | 1Kx8—Dual-Port Master | 52 | CY7C131 | $t_{AA}$ = 25, 30, 35, 45, 55 | 170 @ 25 | J, N |
| 8K | 1Kx8—Dual-Port Slave | 52 | CY7C141 | $t_{AA}$ = 25, 35, 55 | 170 @ 25 | J, N |
| 16K | 2Kx8—Dual Port Master | 48 | CY7C132 | $t_{AA}$ = 25, 30, 35, 45, 55 | 170 @ 25 | P |
| 16K | 2Kx8—Dual-Port Slave | 48 | CY7C142 | $t_{AA}$ = 25, 30, 35, 55 | 170 @ 25 | P |
| 16K | 2Kx8—Dual-Port Master | 52 | CY7C136 | $t_{AA}$ = 25, 30, 35, 45, 55 | 170 @ 25 | J, L, N |
| 16K | 2Kx8—Dual-Port Slave | 52 | CY7C146 | $t_{AA}$ = 25, 35 | 170 @ 25 | J, L, N |
| 32K | 4Kx8—Dual-Port, w/Semaph | 52 | CY7C1342 | $t_{AA}$ = 20, 25, 35, 55 | 240 @ 20 | J |
| 32K | 2Kx16—Dual-Port Slave | 68 | CY7C143 | $t_{AA}$= 15, 25, 35, 55 | 170 @25 | J |
| 32K | 2Kx16—Dual-Port Master | 68 | CY7C133 | $t_{AA}$= 15, 25, 35, 55 | 170 @ 25 | J |
| 32K | 4Kx8—Dual-Port, w/ Semaph, Busy, Int | 64, 68 | CY7C138 | $t_{AA}$ = 15, 35, 55 | 260 @ 15 | J |
| 32K | 4Kx8—Dual-Port, No Arbitration | 52 | CY7C135 | $t_{AA}$ = 20, 25, 35, 55 | 240 @ 20 | J |
| 32K | 4Kx9—Dual-Port, w/ Semaph, Busy, Int | 68, 80 | CY7C139 | $t_{AA}$ = 15, 25, 35, 55 | 260 @ 15 | J, A |
| 64K | 8Kx8—Dual-Port, w/ Semaph, Busy, Int | 64, 68 | CY7C144 | $t_{AA}$ = 15, 25, 35, 55 | 260 @ 15 | J, A |
| 64K | 8Kx9—Dual-Port, w/ Semaph, Busy, Int | 68, 80 | CY7C145 | $t_{AA}$ = 15, 25, 35, 55 | 260 @ 15 | J, A |
| 64K | 4Kx16—Dual-Port, w/ Semaph, Busy, Int | 84, 100 | CY7C024 | $t_{AA}$ = 15, 25, 35, 55 | 280 @ 15 | J,A |
| 64K | 4Kx18—Dual-Port, w/ Semaph, Busy, Int | 84, 100 | CY7C0241 | $t_{AA}$ = 15, 25, 35, 55 | 280 @ 15 | J,A |
| 128K | 8Kx16—Dual-Port w/ Semaph, Busy, Int | 84, 100 | CY7C025 | $t_{AA}$=15,25,35,55 | 280 @ 15 | J,A |
| 128K | 8Kx18—Dual-Port w/ Semaph, Busy, Int | 84, 100 | CY7C0251 | $t_{AA}$=15,25,35,55 | 280 @ 15 | J,A |
| 128K | 16Kx8—Dual-Port w/ Semaph, Busy, Int | 64, 68 | CY7C006 | $t_{AA}$=15,25,35,55 | 260 @ 15 | J,A |
| 128K | 16Kx9—Dual-Port w/ Semaph, Busy, Int | 68, 80 | CY7C016 | $t_{AA}$=15,25,35,55 | 260 @ 15 | J,A |

Note: Please contact a Cypress Representative for product availability.

## SRAM Modules



- x32-Bit Standard SRAM
- x8-, x16-, and x24-Bit Standard SRAM
- x32-Bit Standard SRAM

### Secondary Cache Subsystems

| Size | Organization | Pins | Part Number | Speed (MHz) | $I_{CC}/I_{SB}/I_{CCDR}$ (mA @ ns) | Packages |
|---|---|---|---|---|---|---|
| 256K | P54C Cache (Intel Neptune) | 160 | CYM74SP54 | $f_{max}$=60, 66 MHz | 1500 | PM |
| 512K | P54C Cache (Intel Neptune) | 160 | CYM74SP55 | $f_{max}$=60, 66 MHz | 1500 | PM |
| 256K | P54C Cache (Intel Triton) | 160 | CYM74S430 | 50, 60, 66 MHz | 1200 | PM |
| 512K | P54C Cache (Intel Triton) | 160 | CYM74S431 | 50, 60, 66 MHz | 1200 | PM |
| 256K | P54C Cache (Intel Triton II) | 160 | CYM74P430B | 50, 60, 66 MHz | 750 | PM |
| 512K | P54C Cache (Intel Triton II) | 160 | CYM74P431B | 50, 60, 66 MHz | 1400 | PM |
| 256K | P54C Cache (Intel Triton II ETag) | 160 | CYM74P434B | 50, 60, 66 MHz | 900 | PM |
| 512K | P54C Cache (Intel Triton II ETag) | 160 | CYM74P435B | 50, 60, 66 MHz | 1550 | PM |
| 256K | P54C Cache (Intel Triton II) | 160 | CYM74P436 | 50, 60, 66 MHz | TBD | PM |
| 256K | P54C Cache (Intel Triton II ETag) | 160 | CYM74P438 | 50, J60, 66 MHz | TBD | PM |
| 256K | P54C Cache (OPTi Viper) | 160 | CYM74P550A | 50, 60, 66 MHz | TBD | PM |
| 256K | P54C Cache (OPTi Viper) | 160 | CYM74S550 | 50, 60, 66 MHz | 1500 | PM |
| 512K | P54C Cache (OPTi Viper) | 160 | CYM74S551 | 50, 60, 66 MHz | 1500 | PM |
| 256K | P54C Cache (VLSI 590) | 160 | CYM74S590 | 60, 66 MHz | 1500 | PM |
| 512K | P54C Cache (VLSI 590) | 160 | CYM74S591 | 60, 66 MHz | 1500 | PM |
| 256K | Power PC | 136 | CYM76A256 | 50, 60, 66 MHz | 1250 | PM |
| 512K | Power PC | 136 | CYM76S512 | 50, 60, 66 MHz | 1200 | PM |
| 512K | Power PC | 160 | CYM76S641 | 50, 60, 66 MHz | TBD | PM |
| 512K | Power MAC | 160 | CYM76S681 | 66 MHz | TBD | PM |

Note: Please contact a Cypress Representative for product availability.

## SRAM Modules (continued)

### x8-, x16-, and x24-Bit SRAM Modules

| Size | Organization | Pins | Part Number | Speed (ns) | $I_{CC}/I_{SB}/I_{CCDR}$ (mA @ ns) | Packages |
|---|---|---|---|---|---|---|
| 2M | 256K x 8—JEDEC Sep I/O | 60 | CYM1441 | $t_{AA}$=20, 25, 35, 45 | 960 @ 25 | PZ |
| 4M | 512K x 8—JEDEC | 32 | CYM1464 | $t_{AA}$=20, 22, 25, 30, 35, 45, 55, | 350 @ 20 | PD |
| 4M | 512K x 8—JEDEC | 32 | CYM1465 | $t_{AA}$=70, 85, 100, 120, 150 | 110 @ 70 | PD |
| 16M | 2M x 8 | 36 | CYM1481 | $t_{AA}$=85, 100, 120 | 110 @ 85 | PS |
| 1M | 64K x 16 | 40 | CYM1622 | $t_{AA}$=15, 20, 25, 30, 35, 45 | 400 @ 25 | PV |
| 768K | 32K x 24 | 56 | CYM1720 | $t_{AA}$=15, 20, 25, 30, 35 | 330 @ 25 | PZ |
| 1.5M | 64K x 24 | 56 | CYM1730 | $t_{AA}$=25, 30, 35 | 510 @ 25 | PZ |

### x32-Bit Standard SRAM Module Family

| Size | Organization | Pins | Part Number | Speed (ns) | $I_{CC}/I_{SB}/I_{CCDR}$ (mA @ ns) | Packages |
|---|---|---|---|---|---|---|
| 512K | 16K x 32 | 64 | CYM1821 | $t_{AA}$=20, 25, 35, 45 | 720 @ 20 | PM, PZ |
| 2M | 64K x 32 | 64 | CYM1831 | $t_{AA}$=15, 20, 25, 35, 45 | 720 @ 25 | PM, PN, PZ |
| 4M | 128K x 32 | 64 | CYM1836 | $t_{AA}$=20, 25, 35, 45<br>$t_{AA}$=15 | 480 @ 20<br>760 @ 15 | PM, PZ |
| 4M | 128K x 32 | 66 | CYM1838 | $t_{AA}$=25, 30, 35 | 720 @ 25 | HG |
| 8M | 256K x 32 | 60 | CYM1840 | $t_{AA}$=20, 25, 30, 35, 45, 55 | 1120 @ 25 | PD |
| 8M | 256K x 32 | 64 | CYM1841A | $t_{AA}$=25, 35, 45, 55<br>$t_{AA}$=20<br>$t_{AA}$=12, 15 | 960 @ 25<br>1120 @ 20<br>1600 @ 12 | PM, PN, PZ |
| 8M | 256K x 32 (72-pin Superset) | 72 | CYM1841AP7 | $t_{AA}$=12, 15, 20, 25, 35, 45 | 960 @ 25<br>1120 @ 20<br>1600 @ 15 | PM |
| 16M | 512K x 32 (72-pin Superset) | 72 | CYM1846 | $t_{AA}$=15, 20, 25, 30, 35 | 800 @ 20 | PM, PZ |
| 32M | 1M x 32 (72-pin Superset) | 72 | CYM1851 | $t_{AA}$=15, 20, 25, 30, 35 | 1200 @ 20 | PM, PN, PZ |

### DRAM Controller Modules

| Organization | Bus Width | Part Number | Speed (MHz) | Package |
|---|---|---|---|---|
| DRAM Accelerator | 32-Bit | CYM7232 | 25/33/40 | PGC |
| DRAM Accelerator | 64-Bit | CYM7264 | 25/33/40 | PGC |

Note: Please contact a Cypress Representative for product availability.

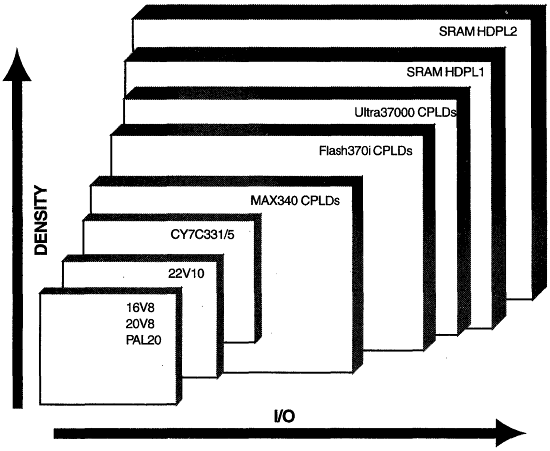
## Programmable Logic Devices (PLDs)

Cypress's Programmable Logic offering ranges from the industry-standard small Programmable Logic Devices (SPLDs) to Complex Programmable Logic Devices (CPLDs) and High-Density Programmable Logic (HDPL) devices. The entire family is supported by the Cypress *Warp*™ development tools for complete device independence during the design cycle.

**PLDs** (Small)

- CMOS Flash and UV erasable technology expertise
- Complete line of SPLDs
  - GAL® architectures like the 16V8, 20V8, PAL20, and 22V10
  - Application specific architectures in the CY7C330 family
- Packaging options include PDIP, PLCC, and LCC packages plus the space-saving QSOP package

**Cypress Programmable Products**



**PLDs**

| Part Number | $t_{PD}$ (ns) | f (MHz) | $I_{CC}$ (mA) | Pins | Packages |
|---|---|---|---|---|---|
| PALC16L8 | 20/25/35 | 28/18 | 70 | 20 | D, L, P, Q, W |
| PALC16L8L | 25/35 | 28/18 | 45 | 20 | D, L, P, Q, W |
| PALC16R4 | 20/25/35 | 28/18 | 70 | 20 | D, L, P, Q, W |
| PALC16R4L | 25/35 | 28/18 | 45 | 20 | D, L, P, Q, W |
| PALC16R6 | 20/25/35 | 28/18 | 70 | 20 | D, L, P, Q, W |
| PALC16R6L | 25/35 | 28/18 | 45 | 20 | D, L, P, Q, W |
| PALC16R8 | 20/25/35 | 28/18 | 70 | 20 | D, L, P, Q, W |
| PALC16R8L | 25/35 | 28/18 | 45 | 20 | D, L, P, Q, W |
| PALCE16V8 | 7.5 | 100 | 115 | 20 | D, J, L, P |
| PALCE16V8 | 10/15/25 | 69/45/33 | 90 | 20 | D, J, L, P, QSOP |
| PALCE16V8L | 15/25 | 45/33 | 55 | 20 | D, J, L, P, QSOP |
| PLDC20G10 | 25/35 | 33/18 | 55 | 24/28 | D, J, L, P, W |
| PLDC20G10B | 15/20/25 | 45/42/33 | 70 | 24/28 | D, J, L, P, W |
| PLDC20RA10 | 15/20/25/35 | 45/42/33/18 | 75 | 24/28 | D, J, L, P, W |
| PALCE20V8 | 7.5 | 100 | 115 | 24/28 | D, J, L, P |
| PALCE20V8 | 10/15/25 | 58/45/33 | 90 | 24/28 | D, J, L, P |
| PALCE20V8L | 15/25 | 45/33 | 55 | 24/28 | D, J, L, P, QSOP |
| PALC22V10* | 20/25/35 | 42/33/18 | 90 | 24/28 | D, J, L, P, Q, W |
| PALC22V10L* | 25/35 | 33/18 | 55 | 24/28 | D, J, L, P, Q, W |
| PALC22V10B* | 15/20 | 50/42 | 90 | 24/28 | D, J, L, P, Q, W |
| PALC22V10D* | 7.5 | 100 | 130 | 24/28 | D, J, L, P |
| PALC22V10D* | 10/15/25 | 77/55/33 | 90 | 24/28 | D, J, L, P |
| PALCE22V10 | 5/7.5 | 142.8/100 | 130 | 24/28 | D, J, L, P |
| PALCE22V10 | 10/15/25 | 77/55/33 | 90 | 24 | D, J, L, P |
| CY7C331 | 20/25/30/40 | 27/24/18/14 | 130 | 28 | D, J, P, Q, W |
| CY7C335 | 15/20/25 | 50/45/35 | 140 | 28 | D, J, P, W |

* Not recommended for new designs.

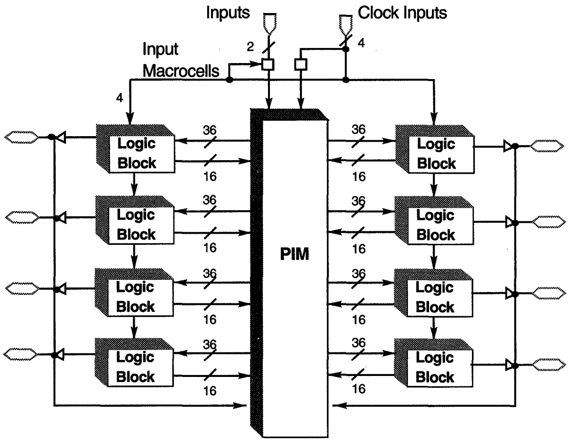
Note: Please contact a Cypress Representative for product availability.

## Complex PLDs (CPLDs)

### UltraLogic™ FLASH370i™

- In-System-Reprogrammable™ (ISR™).
- State-of-the-art Flash technology for electrical erasability.
- Guaranteed routability.
- Unique single product term sharing and steering.
- Architecture provides complete design flexibility for user assigned pinouts. Even when you make changes to your logic, you will be able to maintain the same pin configuration.
- Significant capacity advantages over all other CPLD architectures.
- Simple one case timing model for easy design and system simulation.
- Everything you liked about the 22V10 now available in high-performance, high-density logic.
- ISR versions are upward compatible from earlier devices using existing software tools.

**FLASH370i Family Architecture**



### FLASH370i

| Organization | Pins | Part Number | Speed (ns) | $I_{CC}/I_{SB}$ (mA) | Packages |
|---|---|---|---|---|---|
| 32-Macrocell ISR Flash CPLD | 44 | CY7C371i | $f_{MAX}/t_{PD}/t_S/t_{CO}$=143 MHz/8.5 ns/5 ns/6 ns | 100 | A, J |
| 32-Macrocell ISR Flash CPLD Low Power | 44 | CY7C371iL | $f_{MAX}/t_{PD}/t_S/t_{CO}$=83 MHz/8.5 ns/6.5 ns/6.5 ns | 50 | A, J |
| 64-Macrocell ISR Flash CPLD | 44, 84, 100 | CY7C372i/3i | $f_{MAX}/t_{PD}/t_S/t_{CO}$=125 MHz/10 ns/5.5 ns/6.5ns | 125 | G, J, Y |
| 64-Macrocell ISR Flash CPLD Low Power | 44, 84, 100 | CY7C372iL/3iL | $f_{MAX}/t_{PD}/t_S/t_{CO}$=66 MHz/10 ns/10 ns/10 ns | 60 | A, J |
| 128-Macrocell ISR Flash CPLD | 84, 100, 160 | CY7C374i/5i | $f_{MAX}/t_{PD}/t_S/t_{CO}$ =125 MHz/10 ns/5.5 ns/ 6.5 ns | 200 | A, G, J, Y |
| 128-Macrocell ISR Flash CPLD Low Powr | 84, 100, 160 | CY7C374iL/5iL | $f_{MAX}/t_{PD}/t_S/t_{CO}$ = 66 MHz/10 ns/10 ns/10 ns | 125 | A, G, J, U |

### MAX340

- Cypress's first-generation CPLD remains an innovator in the CPLD market.
- Only second-sourced family of CPLDs available.
- Offers a range of general-purpose programmable logic that makes it ideal for replacing large amounts of TTL logic.
- High-volume 0.8µ UV-erasable CMOS process.
- Family offers faster and denser devices than previous generations of Small PLDs and EPLDs.
- CY7C340B devices are 0.65µ shrinks of the original versions offering even faster speed options.

### MAX340

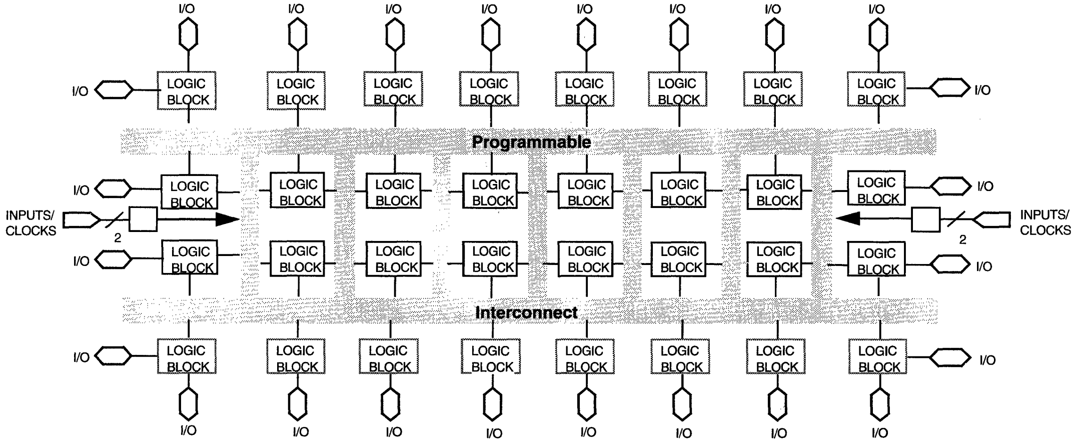
| Organization | Pins | Part Number | Speed (ns) | $I_{CC}/I_{SB}$ (mA) | Packages |
|---|---|---|---|---|---|
| 32 Macrocell CPLD | 28S | CY7C344/B | $t_{PD/S/CO}$ = 15/10/10, 10/6/5 | 200/150 | H, J, P, W |
| 64 Macrocell CPLD | 44 | CY7C343/B | $t_{PD/S/CO}$ = 20/12/12, 12/8/6 | 135/125 | H, J, R |
| 128 Macrocell CPLD | 68 | CY7C342/B | $t_{PD/S/CO}$ = 25/15/14, 12/8/6 | 250/225 | H, J, R |
| 128 Macrocell CPLD | 84, 100 | CY7C346/B | $t_{PD/S/CO}$ = 25/15/14, 15/10/7 | 250/225 | H, J, N, R |
| 192 Macrocell CPLD | 84 | CY7C341/B | $t_{PD/S/CO}$ = 25/15/14, 15/10/7 | 380/360 | H, J, R |

Note: Please contact a Cypress Representative for product availability.

## Complex PLDs (CPLDs) (continued)

### UltraLogic™ Ultra37000™

- In-System Reprogrammable CPLDs offering high density and high performance
- Fully compliant to PCI Local Bus Specification
- Simple timing model for easy design and system simulation
- Full JTAG (IEEE 1149.1) compatibility
- Operation at 3.3V or 5V
- Extends the density of Flash CPLDs up to 512 macrocells
- Supports fixed pin assignments



### Ultra37000™

| Organization | Pins | Part Number | Speed (ns) | $I_{CC}$ (mA) | Packages |
|---|---|---|---|---|---|
| 128-Macrocell Flash ISR CPLD | 84, 100, 160 | CY7C37128 | $f_{MAX}/t_{PD}/t_S/t_{CO}$=167 MHz/6.5 ns/3.5 ns/4.5 ns | 75 | A, J |
| 128-Macrocell Flash ISR CPLD 3.3 V | 84, 100, 160 | CY7C37128V | $f_{MAX}/t_{PD}/t_S/t_{CO}$=125 MHz/10 ns/5.5 ns/6.5 ns | TBD | A, J |
| 192-Macrocell Flash ISR CPLD | 160, 256 | CY7C37192 | $f_{MAX}/t_{PD}/t_S/t_{CO}$=154 MHz/7.5 ns/4 ns/4.5 ns | TBD | A, BG |
| 192-Macrocell Flash ISR CPLD 3.3 V | 160, 256 | CY7C37192V | $f_{MAX}/t_{PD}/t_S/t_{CO}$=100 MHz/12 ns/6 ns/7 ns | TBD | A, BG |
| 256-Macrocell Flash ISR CPLD | 160, 256 | CY7C37256 | $f_{MAX}/t_{PD}/t_S/t_{CO}$=154 MHz/7.5 ns/4 ns/4.5 ns | TBD | A, BG |
| 256-Macrocell Flash ISR CPLD 3.3 V | 160, 256 | CY7C37256V | $f_{MAX}/t_{PD}/t_S/t_{CO}$=100 MHz/12 ns/6 ns/7 ns | TBD | A, BG |
| 384-Macrocell Flash ISR CPLD | 160, 256 | CY7C37384 | $f_{MAX}/t_{PD}/t_S/t_{CO}$=125 MHz/10 ns/5.5 ns/6.5 ns | TBD | A, BG |
| 384-Macrocell Flash ISR CPLD 3.3 V | 160, 256 | CY7C37384V | $f_{MAX}/t_{PD}/t_S/t_{CO}$=83 MHz/15 ns/8 ns/8 ns | TBD | A, BG |
| 512-Macrocell Flash ISR CPLD | 160, 352 | CY7C37512 | $f_{MAX}/t_{PD}/t_S/t_{CO}$=125 MHz/10 ns/5.5 ns/6.5 ns | TBD | A, BG |
| 512-Macrocell Flash ISR CPLD 3.3 V | 160, 352 | CY7C37512V | $f_{MAX}/t_{PD}/t_S/t_{CO}$=83 MHz/15 ns/8 ns/8 ns | TBD | A, BG |

Note: Please contact a Cypress Representative for product availability.

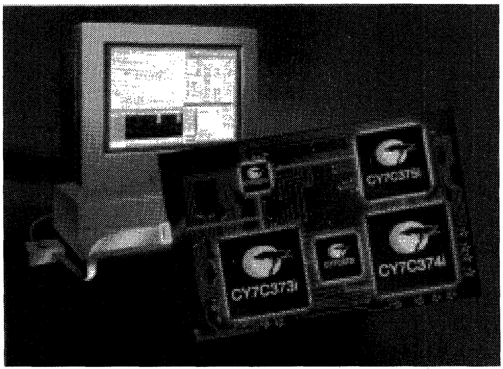
## Design and Programming Tools

The capabilities of our UltraLogic product line—SPLDs, CPLDs, and *Warp* design tools—greatly simplify your design effort and let you reach the optimal solution to your design problem in the shortest possible time.

- Supports entire Programmable Logic product offering
- Open, IEEE-STD-1164 VHDL (Very-high-speed integrated-circuit Hardware Description Language) for programmable logic design
- Accepts designs as VHDL text, via schematic capture or in any combination of these entry modes
- Shortens the design process through its extraordinary capabilities
  - VHDL source code verification
  - Debug
  - Graphical waveform simulation and editing
  - Graphical timing simulation and analysis
- Multiple entry points
  - *Warp2*, low-cost VHDL tool
  - *Warp2*Sim, adds a full timing simulator
  - *Warp3*, adds schematic entry, VHDL debugger, and a full timing simulator
- *Impulse3*™ programmer is a low cost, engineering support programmer
  - Supports all of Cypress's programmable devices
  - Gives you the ability to prove your design, in silicon, with a short cycle time
- InSRKit™ programmer is a low cost programmer for on-board programming of FLASH370i ISR devices
- *Warp2*ISR™ kits together a *Warp2* with InSRkit and CPLD samples for a complete low-cost ISR design and programming solution

### Design and Programming Tools

| Description | Type | Part Number |
|---|---|---|
| *Warp2*® for PC | VHDL Design Tool | CY3120 |
| *Warp2* for Workstation | VHDL Design Tool | CY3125 |
| *Warp2*Sim™ for PC | VHDL Design Tool with Simulator | CY3122 |
| *Warp2*Sim™ for Workstation | VHDL Design Tool with Simulator | CY3127 |
| *Warp3*® for PC | VHDL/CAE Design Tool | CY3130 |
| *Warp3* for Workstation | VHDL/CAE Design Tool | CY3135 |
| Abel™ Kit | FLASH370i Design Kit for Abel | CY3140ABLSW |
| Cypress Mentor Graphics Bolt-in Kit | Bolt-In Design Kit for Mentor Graphics | CY3144 |
| *Impulse3*™ | Programmer, High Pin Count Adapter | CY3500, CY3501A |
| InSRKit™ | ISR Programmer for PC | CY3600 |
| *Warp2*ISR™ | VHDL Design Tool with ISR Programmer | CY3620 |



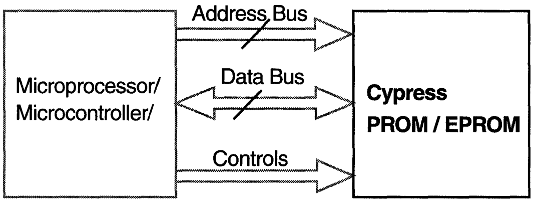
Complete ISR CPLD Solutions

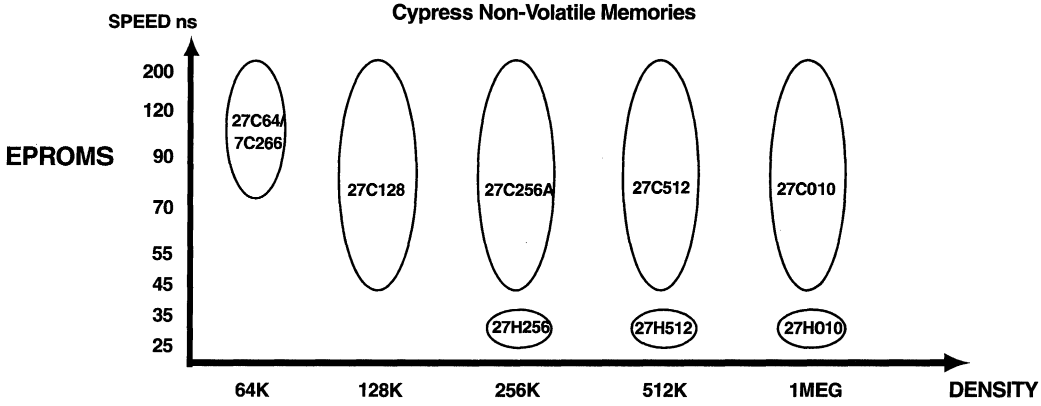
Note: Please contact a Cypress Representative for product availability.

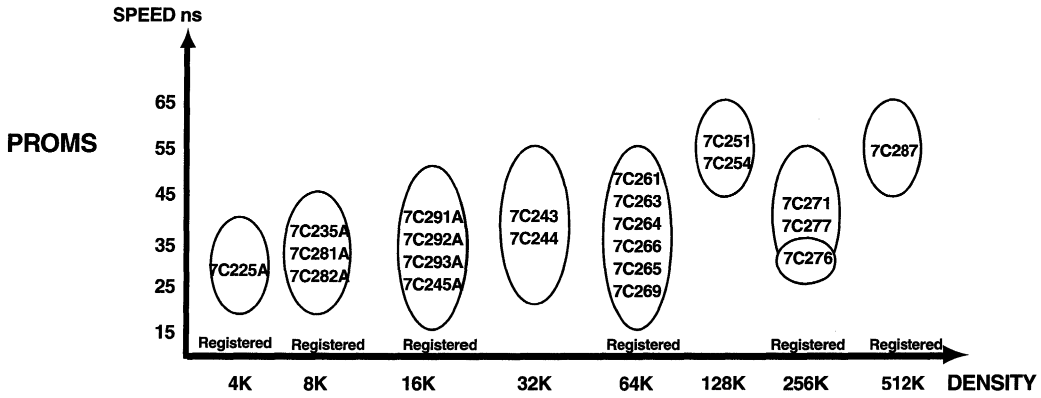
## Non-Volatile Memories

- World's fastest PROMs and EPROMs
- 4K to 1M density
- Industry-standard pinouts
- Low-power CMOS technology
- PDIP, PLCC, TSOP, CDIP, LCC packages
- Registered and x16 versions
- Direct replacement for Bipolar PROMs

**Ultra high-speed Cypress PROMs/EPROMs allow CPUs to execute directly with zero wait states.**







Note: Please contact a Cypress Representative for product availability.

## Non-Volatile Memories (continued)

### CMOS PROMs

| Density | Organization | Pins Count | Part Number | Access Time $t_{AA}$ (ns) | Feature |
|---|---|---|---|---|---|
| 8K | 1Kx8 | 24/28 | CY7C281A | 25/30/45 | 300-mil DIP |
| 16K | 2Kx8 | 24/28 | CY7C291A/AL | 20/25/35/50 | 300-mil DIP |
| 16K | 2Kx8 | 24 | CY7C292A/AL | 20/25/35/50 | 600-mil DIP |
| 16K | 2Kx8 | 24/28 | CY7C293A/AL | 20/25/35/50 | Power-Down |
| 32K | 4Kx8 | 24/28 | CY7C243 | 20/25/35/45/55 | 300-mil DIP |
| 32K | 4Kx8 | 24 | CY7C244 | 20/25/35/45/55 | 600-mil DIP |
| 64K | 8Kx8 | 24/28 | CY7C261 | 20/25/35/45/55 | Power-Down |
| 64K | 8Kx8 | 24/28 | CY7C263 | 20/25/35/45/55 | 300-mil DIP |
| 64K | 8Kx8 | 24 | CY7C264 | 20/25/35/45/55 | 600-mil DIP |
| 64K | 8Kx8 | 28/32 | CY7C266 | 20/25/35/45 | EPROM Pin Out |
| 128K | 16Kx8 | 28/32 | CY7C251 | 45/55/65 | Power-Down |
| 256K | 32Kx8 | 28/32 | CY7C271A | 25/30/35/45/55 | Power-Down |
| 256K | 16Kx16 | 44 | CY7C276 | 25/30/35 | 16 Bit Word Wide |

### CMOS Registered PROMs

| Density | Organization | Pins Count | Part Number | Set-Up $t_{SA}$ (ns) | Clock-to-Out $t_{CO}$ (ns) |
|---|---|---|---|---|---|
| 4K | 512x8 | 24/28 | CY7C225A | 18/25/30/35/40 | 12/12/15/20/25 |
| 8K | 1Kx8 | 24/28 | CY7C235A | 18/25/30/40 | 12/12/15/20 |
| 16K | 2Kx8 | 24/28 | CY7C245A/AL | 15/18/25/35/45 | 10/12/12/15/25 |
| 64K | 8Kx8 | 28 | CY7C265 | 15/25/40/50 | 12/15/20/25 |
| 64K | 8Kx8 | 28 | CY7C269 | 15/25/40/50 | 12/15/20/25 |
| 256K | 32Kx8 | 28/32 | CY7C277 | 30/40/50 | 15/20/25 |
| 512K | 64Kx8 | 28/32 | CY7C287 | 45/55/65 | 15/20/25 |

### CMOS EPROMs

| Density | Organization | Pin Count | Part Number | Access Time $t_{AA}$ (ns) | Feature |
|---|---|---|---|---|---|
| 64K | 8Kx8 | 28/32 | CY27C64 | 45/55/70/90/120/150/200 | High Speed, CY7C266 Algorithm |
| 128K | 16Kx8 | 28/32 | CY27C128 | 45/55/70/90/120/150/200 | High Speed, Cypress Proprietary Algorithm |
| 256K | 32Kx8 | 28/32 | CY27C256A | 45/55/70/90/120/150/200 | Industry Standard Algorithm |
| 256K | 32Kx8 | 28/32 | CY27H256 | 25/30/35 | High Speed, Industry Standard Algorithm |
| 512K | 64Kx8 | 28/32 | CY27C512 | 45/55/70/90/120/150/200 | Industry Standard Algorithm |
| 512K | 64Kx8 | 28/32 | CY27H512 | 25/30/35 | High Speed, Industry Standard Algorithm |
| 1024K | 128Kx8 | 32 | CY27C010 | 45/55/70/90/120/150/200 | Industry Standard Algorithm |
| 1024K | 128Kx8 | 32 | CY27H010 | 25/30/35 | High Speed, Industry Standard Algorithm |
| 512K | 32Kx16 | 40/44 | CY27C516* | 45/55/70/90/120/150/200 | Industry Standard Algorithm |
| 1024K | 64Kx16 | 40/44 | CY27C1024* | 45/55/70/90/120/150/200 | Industry Standard Algorithm |

* Available 2Q97

Note: Please contact a Cypress Representative for product availability.

## Communication Products

| Description | Pins | Part Number | Speed (MHz) | $I_{CC}$ (mA) | Packages |
|---|---|---|---|---|---|
| HOTLink Transmitter | 28 | CY7B923 | 155-400 | 65 | J, L, S |
| HOTLink Receiver | 28 | CY7B933 | 155-400 | 120 | J, L, S |
| HOTLink Evaluation Card | N/A | CY9266 | 155-400 | N/A | C, F, T* |
| SMPTE Encoder | 44 | CY7C9235 | 155-400 | 250 | J |
| SMPTE Decoder | 100, 84 | CY7C9335 | 155-400 | 300 | A, J |
| SONET/SDH LAN Serial Transceiver | 24 | CY7B951 | 51 & 155 | 50 | S |
| SONET/SDH WAN Serial Transceiver | 24 | CY7B952 | 51 & 155 | 50 | S |
| SONET/SDH LAN Framer/Transceiver | 128 | CY7C955 | 51 & 155 | TBD | N |
| 10BASE 2/5 Ethernet Coax Transceiver | 16, 28 | CY7B8392 | 10 | 25 | J, P |
| 10BASE-T Repeater | 84 | CY7C981 | 10 | 180 | J |
| 10BASE-T MIB | 28 | CY7C987 | 10 | TBD | J |
| 10BASE- FL Fiber Transceiver | 28 | CY7B4663 | 10 | 70 | J |
| 10BASE-T to FL Convertor | 20 | CY7B4665 | 10 | 70 | J |
| Fast Ethernet 100BASE-T4 Transceiver | 80 | CY7C971 | 10 & 100 | 300 | N |
| Fast Ethernet 100BASE-TX Transceiver | 44 | CY7C972 | 100 | 200 | N |
| Fast Ethernet 100BASE-T Repeater | 208 | CY7C9700 | 10 & 100 | 170 | J |

* Interface: C-Coax; T-twisted pair; F-fiber

**CY7B923 and CY7B933:** HOTLink™
- Transmitter/receiver chipset
- 155-400 Mb/s operation
- Fibre Channel/ESCON™ compliant
- On-chip 8B/10B encoding/decoding
- Built-In Self-Test (BIST)
- Evaluation boards available (CY9266)
- User's Guide available

**CY7B9235/9335:** SMPTE Encoder/Decoder for Video applications
- Compliant to SMPTE-259M and DVB-ASI standards
- Works with HOTLINK receiver/transmitter

**CY7B951:** SONET/SDH LAN Serial Transceiver
- SONET/SDH and ATM compatible
- OC-1 (51.8 MHz) and OC-3 (155.5 MHz) clock/data recovery
- Meets LAN Bellcore specifications
- Loop-back testing

**CY7B952:** SONET/SDH WAN Serial Transceiver
- SONET/SDH and ATM compatible
- OC-1 (51.8 MHz) and OC-3 (155.5 MHz) clock/data recovery
- Meets WAN Bellcore specifications
- Loop-back testing

**CY7C955:** SONET/SDH Framer/Transceiver
- Pin compatible to PMC Sierra 5346 (S/UNI-Lite)
- SONET/SDH and ATM compatible
- OC-1 (51.8 MHz) and OC-3 (155.5 MHz) framer/clock/data recovery

**CY7B8392:** Low-power Coax Transceiver for 10 Mbps Ethernet applications
- Pin compatible to industry standard 8392
- 10BASE5 and 10BASE2 applications
- Auto Attachment Unit Interface (AUI)
- Hybrid collision detection

**CY7B4663:** 10BASE-FL Transceiver for Ethernet applications
- Pin compatible ML4663
- Low power: Idle current < 70mA
- 100mA LED driver

**CY7B4665:** 10BASE-T to 10BASE-FL Convertor for Ethernet applications
- Pin compatible ML4665
- Low power: Idle current < 70mA
- Integrated quantizer and LED driver

**CY7C981:** 10BASE-T Mulitport Repeater for Ethernet applications
- Pin compatible to AM79C981
- Fully IEEE 802.3 compliant
- Interfaces with CY7C987 to form fully managed repeater

**CY7C987:** 10BASE-T MIB for Ethernet applications
- Pin compatible to AM79C987
- Interfaces with CY7C981 to form fully managed repeater

Note: Please contact a Cypress Representative for product availability.

## Communication Products (continued)

**CY7C971:** 100BASE-T4/10BASE-T
Fast Ethernet Transceiver

- Three operating modes:100BASE-T4, 10BASE-T Full Duplex, 10BASE-T
- Media Independant Interface (MII)
- N-way auto negotiation
- Receive filter/Adaptive equalization
- Low latency repeater mode
- Category 3 wiring

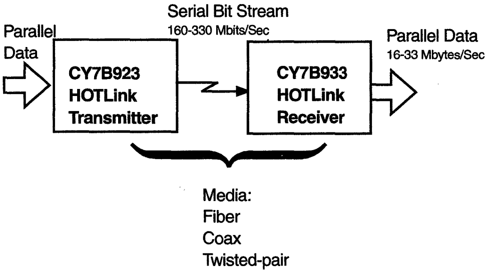
**CY7C972:** 100BASE-TX
Fast Ethernet Transceiver

- Supports 100 MHz operation
- Category 5 wiring

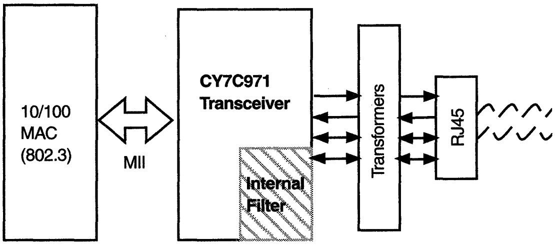
**CY7C9700:** 100BASE-T Repeater for Ethernet applications

- 12 + 1 port
- Expansion Port Interface
- T4 and TX capable
- MII Interface
- Point to point stackable option

**Point-to-Point HOTLink**



**Cypress 100BASE-T4 Ethernet Transceiver**



Note: Please contact a Cypress Representative for product availability.

## FIFOs

**Synchronous:**

- 100 MHz operation
- Low power
- Programmable flags
- All densities pin compatible (up to 128Kx9 or 64Kx18)
- TQFP packaging

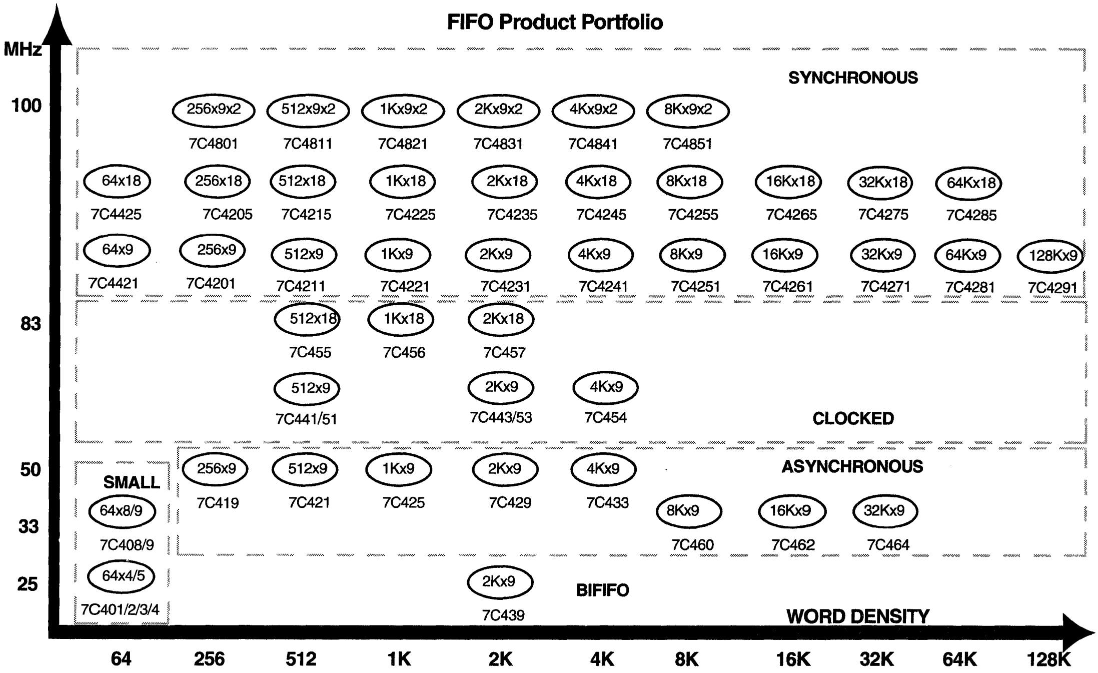
**Clocked:**

- 83 MHz operation
- Low power
- PLCC/PQFP Packaging

**Asynchronous:**

- 10-ns access time
- Industry-standard operation and pinout
- Densities up to 32Kx9—all densities pin compatible
- TQFP packaging

**FIFO Product Portfolio**



Note: Please contact a Cypress Representative for product availability.

# FIFOs (continued)

## Synchronous FIFOs

| Organization | Pins | Part Number | Speed (ns) | $I_{CC}$ (mA @20 MHz) | Packages |
|---|---|---|---|---|---|
| 64 x 9—Synchronous | 32 | CY7C4421 | 10, 15, 25, 35* | 50 | A, J |
| 256 x 9—Synchronous | 32 | CY7C4201 | 10, 15, 25, 35* | 50 | A, J |
| 512 x 9—Synchronous | 32 | CY7C4211 | 10, 15, 25, 35* | 50 | A, J |
| 1K x 9— Synchronous | 32 | CY7C4221 | 10, 15, 25, 35* | 50 | A, J |
| 2K x 9—Synchronous | 32 | CY7C4231 | 10, 15, 25, 35* | 50 | A, J |
| 4K x 9—Synchronous | 32 | CY7C4241 | 10, 15, 25, 35* | 50 | A, J |
| 8K x 9—Synchronous | 32 | CY7C4251 | 10, 15, 25, 35* | 50 | A, J |
| 64 x 18—Synchronous | 64, 68 | CY7C4425 | 10, 15, 25, 35* | 100 | A, J |
| 256 x 18—Synchronous | 64, 68 | CY7C4205 | 10, 15, 25, 35* | 100 | A, J |
| 512 x 18— Synchronous | 64, 68 | CY7C4215 | 10, 15, 25, 35* | 100 | A, J |
| 1K x 18—Synchronous | 64, 68 | CY7C4225 | 10, 15, 25, 35* | 100 | A, J |
| 2K x 18—Synchronous | 64, 68 | CY7C4235 | 10, 15, 25, 35* | 100 | A, J |
| 4K x 18—Synchronous | 64, 68 | CY7C4245 | 10, 15, 25, 35* | 100 | A, J |
| 16K x 9—Deep Sync | 32 | CY7C4261 | 10, 15, 25, 35* | 35 | A, J |
| 32K x 9—Deep Sync | 32 | CY7C4271 | 10, 15, 15, 35* | 35 | A, J |
| 8K x 18—Deep Sync | 64, 68 | CY7C4255 | 10, 15, 25, 35* | 45 | A, J |
| 16K x 18—Deep Sync | 64, 68 | CY7C4265 | 10, 15, 25, 35* | 45 | A, J |
| 64K x 9—Deep SyncII | 64, 32 | CY7C4281 | 10, 15, 25, 35* | 40 | A, J |
| 128K x 9—Deep SyncII | 64, 32 | CY7C4291 | 10, 15, 15, 35* | 40 | A, J |
| 32K x 18—Deep SyncII | 64, 68 | CY7C4275 | 10, 15, 25, 35* | 50 | A, J |
| 64K x 18—Deep SyncII | 64, 68 | CY7C4285 | 10, 15, 25, 35* | 50 | A, J |
| 256 x 9 Double Sync | 64 | CY7C4801 | 10, 15, 25, 35 | 60 | A |
| 512 x 9 Double Sync | 64 | CY7C4811 | 10, 15, 25, 35 | 60 | A |
| 1K x 9 Double Sync | 64 | CY7C4821 | 10, 15, 25, 35 | 60 | A |
| 2K x 9 Double Sync | 64 | CY7C4831 | 10, 15, 25, 35 | 60 | A |
| 4K x 9 Double Sync | 64 | CY7C4841 | 10, 15, 25, 35 | 60 | A |
| 8K x 9 Double Sync | 64 | CY7C4851 | 10, 15, 25, 35 | 60 | A |

## Clocked FIFOs

| Organization | Pins | Part Number | Speed (ns) | $I_{CC}$ (mA @ MHz) | Packages |
|---|---|---|---|---|---|
| 512x9—Clocked | 28S, 32 | CY7C441 | 12, 14, 20, 30* | 70 @ 20 | J |
| 512x9—Clocked w/ Prog. Flags | 32 | CY7C451 | 12, 14, 20, 30* | 70 @ 20 | J |
| 2Kx9—Clocked | 28S, 32 | CY7C443 | 12, 14, 20, 30* | 70 @ 20 | J |
| 2Kx9—Clocked w/ Prog. Flags | 32 | CY7C453 | 12, 14, 20, 30* | 70 @ 20 | J |
| 4Kx9—Clocked w/ Prog. Flags | 32 | CY7C454 | 12, 14, 20, 30* | 70 @ 20 | J |
| 512 x 18—Clocked w/Prog. Flags | 52 | CY7C455 | 12, 14, 20, 30* | 90 @ 20 | J, N |
| 1K x 18—Clocked w/Prog. Flags | 52 | CY7C456 | 12, 14, 20, 30* | 90 @ 20 | J, N |
| 2K x 18—Clocked w/Prog. Flags | 52 | CY7C457 | 12, 14, 20, 30* | 90 @ 20 | J, N |

* Cycle Times

Note: Please contact a Cypress Representative for product availability.

## FIFOs (continued)

### Asynchronous FIFOs

| Organization | Pins | Part Number | Speed (ns) | $I_{CC}$ (mA @ ns) | Packages |
|---|---|---|---|---|---|
| 64 x 4 | 16 | CY3341 | 1.2, 2 MHz | 45 | D, P |
| 64 x 4 | 16 | CY7C401 | 5, 10, 15, 25 MHz | 75 | DMB, P |
| 64 x 4—w/$\overline{\text{OE}}$ | 16 | CY7C403 | 10, 15, 25 MHz | 75 | DMB, P |
| 64 x 5 | 18 | CY7C402 | 10, 15, 25 MHz | 75 | DMB, P |
| 64 x 5—w/$\overline{\text{OE}}$ | 18 | CY7C404 | 10, 15, 25 MHz | 75 | P |
| 64 x 8—w/$\overline{\text{OE}}$ and Almost Flags | 28S | CY7C408A | 15, 25, 35 MHz | 115 @ 15 | P, V |
| 64 x 9—w/Almost Flags | 28S | CY7C409A | 15, 25, 35 MHz | 115 @ 15 | P, V |
| 256 x 9—w/Half Full Flag | 28S, 32 | CY7C419 | 10, 15, 20, 25, 30, 40, 65 | 35 @ 20 | A, LMB, P, V |
| 512 x 9—w/Half Full Flag | 28 | CY7C420 | 20, 25, 30, 40, 65 | 35 @ 20 | P |
| 512 x 9—w/Half Full Flag | 28S, 32 | CY7C421 | 10, 15, 20, 25, 30, 40, 65 | 35 @ 20 | A, DMB, J, P, V |
| 1K x 9—w/ Half Full Flag | 28 | CY7C424 | 20, 25, 30, 40, 65 | 35 @ 20 | DMB |
| 1K x 9—w/ Half Full Flag | 28S, 32 | CY7C425 | 10, 15, 20, 25, 30, 40, 65 | 35 @ 20 | A, J, L, P, V |
| 2K x 9—w/ Half Full Flag | 28 | CY7C428 | 20, 25, 30, 40, 65 | 35 @ 20 | P |
| 2K x 9—w/ Half Full Flag | 28S, 32 | CY7C429 | 10, 15, 20, 25, 30, 40, 65 | 35 @ 20 | A, J, L, P, V |
| 4K x 9—w/ Half Full Flag | 28 | CY7C432 | 25, 30, 40, 65 | 35 @ 20 | P |
| 4K x 9—w/Half Full Flag | 28S, 32 | CY7C433 | 10, 15, 20, 25, 30, 40, 65 | 35 @ 20 | A, J, L, P, V |
| 8Kx9—w/ Half Full Flag | 28 | CY7C460 | 15, 25, 40 | 105 @ 15 | DMB,LMB,J, P |
| 16Kx9—w/ Half Full Flag | 28 | CY7C462 | 15, 25, 40 | 105 @ 15 | DMB,LMB,J, P |
| 32Kx9—w/ Half Full Flag | 28 | CY7C464 | 15, 25, 40 | 105 @ 15 | DMB,LMB,J, P |
| 2K x 9—Bidirectional | 28S | CY7C439 | 30, 65 | 147@ 25 | J, P |

Note: Please contact a Cypress Representative for product availability.

## VMEbus Interface Products

Cypress offers a broad range of VME interface products ranging from low-cost slave I/O controllers to high end 32/64 - bit master controllers. All products are listed in detail in the VME Handbook.

**VIC068A and VAC068A**

- 32-bit VMEbus operation
- Complete VMEbus interface controller/ arbiter
- Complete master/slave capability
- Commercial / Industrial / Military

**VIC64**

- 64-bit MBLT operation
- Software and pin compatible to VIC068A
- Complete master/slave capability
- Commercial / Industrial / Military

**CY7C960**

- Low cost VME64-Compatible Slave Controller
- 64 pin TQFP
- Commercial / Military

**CY7C961**

- Low cost VME64-Compatible Master/Slave Controller
- 64 pin TQFP
- Commercial / Military

**CY7C964**

- Companion to VIC64, VIC068A, CY7C960 and CY7C961
- 100 pin TQFP
- Commercial / Military

**CY7C965:** RACEway

- High-speed Crosspoint Interconnect
- Implements Open Bus Standard (VITA 5-1994)
- 160 Mbyte/sec throughput per port
- 6 32-bit data ports

| Description | Pins | Part Number | Transfer Rate (MB/s) | $I_{CC}$ (mA) | Packages |
|---|---|---|---|---|---|
| VME Interface Controller | 144/160 | VIC068A | 40 | 150 | A, B, G, N, U |
| VME Address Controller | 144/160 | VAC068A | — | 150 | B, G, N, U |
| 64-Bit VIC | 144/160 | VIC64 | 80 | 150 | A, B, G, N, U |
| Low Cost VMEbus Slave Controller | 64/100 | CY7C960 | 80 | 150 | A, N, U |
| Low Cost VMEbus Master / Slave Controller | 64/100 | CY7C961 | 80 | 150 | N, U |
| Bus Interface Logic Circuit | 64/68 | CY7C964 | — | 120 | A,G, N, U |
| RACEway | 361 | CY7C965 | 160 | 100 | Ball Grid Array |

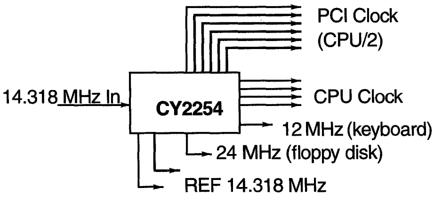
## Timing Technology Products

Today's high-performance digital designs often require multiple clock frequencies in order to control all the functions on the board. In the past, this required the designer to use multiple metal-can oscillators which increased the size and cost of the design. Cypress solves this problem by providing single chips which can synthesize multiple frequencies.

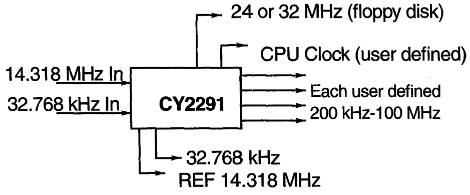
Cypress specializes in the development and production of high-performance frequency synthesis devices. Cypress motherboard frequency synthesizers are specifically designed to provide frequencies used in PC motherboard design. PC graphics frequency products provide frequencies most often required for graphics card designs. Cypress also offers general purpose, programmable clock synthesizers for use in any application that requires multiple clock frequencies and in-system frequency changes.

Cypress is the only manufacturer offering factory EPROM-programmability for custom frequency generation in a varied selection of products. With EPROM-programming capability, samples of custom frequency products can be provided in days rather than the weeks or months required by older technologies.

**Pentium Processor Compatble**



**All Purpose Clock Synthesizer**



Note: Please contact a Cypress Representative for product availability.

## Timing Technology Products (continued)

| Application | Part# | PLLs | Oputs | Features | Pkgs |
|---|---|---|---|---|---|
| Industry Standard Motherboard Frequency Synthesizers | CY2250 | 1 | 14 | Pentium/Pentium Pro servers: 12 skew controlled CPU clocks (250 ps pin-to-pin), 2 buffered reference clocks, 3.3V | 28 SOIC |
| | CY2252 | 2 | 14 | Pentium portables: 5 CPU/6 PCI clocks (2 "early" PCI for docking stations), 24 MHz, 2 buffered reference clocks, 3.3V | 28 SSOP |
| | CY2254A | 2 | 14 | Intel Triton chipset compatible: 4 CPU/6 PCI clocks, 12 MHz, 24 MHz, 2 buffered reference clocks, 3.3V | 28 SOIC |
| | CY2256 | 1 | 14 | OPTi Viper chipset compatible: 5 CPU, 7PCI clocks, 2 buffered reference clocks, 3.3V | 28 SOIC |
| | CY2257 | 1 | 14 | Ali Aladdin chipset compatible: 6 CPU/6 PCI clocks, 2 buffered reference clocks, 3.3V | 28 SOIC |
| | CY2260 | 2 | 14 | Intel Triton II chipset compatible: 4 CPU/6 PCI clocks, 48 MHz clock, 3 buffered reference clocks, 3.3V | 28 SOIC |
| SDRAM-Compatible Motherboard Frequency Synthesizers | CY2263 | 2 | 17 | For mobile applications; 7CPU/6PCI clocks, 24MHz, 48 MHz, 2 buffered reference clocks, 3.3V. | 34SS0P |
| | CY2264 | 2 | 18 | Intel VX- clipset compatible; 8CPU/6PCI clocks, 24 MHz, 48 MHz, 2 buffered reference clocks, 3.3 V. | 34SS0P |
| | CY2265 | 2 | 22 | Intel VX- clipset compatible; 12CPU/6PCI clocks, 24 MHz, 48 MHz, 2 buffered reference clocks, 3.3 V. | 34SS0P |
| | CY2267 | 2 | 20 | Intel TX- clipset compatible; 16 CPU/1 PCI clocks, 24 MHz, 48 MHz, buffered reference clocks, 3.3V. | 34SS0P |
| Industry Standard Clocks for 2.5V/ 3.3V Pentium and Pentium II applications (Include SDRAM support). (Please see datasheets for full descriptions of these devices. Multiple varieties of the given part numbers are available) | CY2273 | 3 | | For Pentium / Pentium II-based systems such as Intel 430TX and ALI Aladdin IV | 48 SSOP |
| | CY2274 | 3 | | For Pentium / Pentium II-based systems such as Intel 440FX (CK25) | 48 SSOP |
| | CY2275 | 3 | | For Pentium / Pentium II-based systems such as Intel 440LX (CK3D) | 48 SSOP |
| | CY2276 | 3 | | For Pentium / Pentium II-based systems such as Intel 440LX (CK4D) | 56 SSOP |
| | CY2277 | 3 | | For Pentium / Pentium II-based systems such as Intel 430TX (CKDM66-M) | 48 SSOP |
| | CY2278 | 3 | | For Portable Pentium / Pentium II-based systems such as those using Intel 430TX | 48 TSSOP |
| | CY2030 | 2 | | I/O Companion chip , Reference, I/O, USB, other clocks (CKIO) for use with CK3D and CK4D | 20 SSOP |
| General Purpose Programmable Products (486 Pentium/Pentium Pro motherboards, peripherals, cable TV, video games, etc.) | CY2071A | 1 | 3 | Factory EPROM programmable single PLL, 0.5-100 MHz, 5V/3.3V | 8 SOIC |
| | CY2907 | 1 | 2 | Factory EPROM programmable single selectable PLL, 2-120 MHz, 5V/3.3V | 8 SOIC or 14 SOIC |
| | CY2081 | 3 | 3 | Factory EPROM programmable triple PLL, 0.5-100 MHz, 5V/3.3V | 8 SOIC |
| | CY2291 | 3 | 8 | Factory EPROM progrmmable triple PLL, 0.2-100 MHz, 5V/3.3V | 20 SOIC |
| | CY2292 | 3 | 6 | Factory EPROM programmable triple PLL, 0.2-100 MHz, 5V/3.3V | 16 SOIC |
| | ICD2051 | 2 | 5 | User-programmable dual PLL, 0.3-120 MHz, 5V | 16 SOIC |
| PC Graphics Frequency Synthesizers | ICD2061A | 2 | 2 | User-programmable PC video/memory clocks, 0.4-120 MHz, 5V | |
| SDRAM Buffer (Zero-Delay Buffer) | CY2305 | 1 | 5 | 1-in/5-out, 25 to 75 MHz | 8 SOIC |
| | CY2309 | 1 | 9 | 1-in/9-out, 25 to 75 MHz, outputs arranged 4+4+1 | 16 SOIC |
| 3.3V Low Skew Clock Buffer | CY2308 | 1 | 8 | 1-in/8-out, 10 to 100 MHz, separate feedback control | 16 SOIC |
| Programmable Skew Clock Buffer (TTL Output) | CY7B991 | 1 | 8 | 3-80 MHz, Programmable Skew (700 ps increments) 250 ps pin-to-pin skew | J, L |
| Programmable Skew Clock Buffer (CMOS Output) | CY7B992 | 1 | 8 | 3-80 MHz, Programmable Skew (700 ps increments) 250 ps pin-to-pin skew | J, L |
| Low Skew Clock Buffer (TTL Output) | CY7B9910 | 1 | 8 | 15-80 MHz, tPD = 500 ps 250 ps pin-to-pin skew | S |
| Low Skew Clock Buffer (CMOS Output) | CY7B9920 | 1 | 8 | 15-80 MHz, tPD = 500 ps 250 ps pin-to-pin skew | S |

Note: Please contact a Cypress Representative for product availability.

## PC Chipsets

| Description | Part Number | Package |
|---|---|---|
| Single-chip solution for 386/486-based systems with Green features. Supports SMI/CPU interface/cache control/DRAM control/ISA Bus control/VESA control | CY82C597 | 160-pin PQFP |
| Intelligent PCI Bus Bridge Chip. Connects VESA Bus to the PCI Bus | CY82C599 | 160-pin PQFQ |
| hyperCache™ Chipset System Controller | CY82C691 | 208-pin PQFP |
| hyperCache Chipset Data-Path with Integrated Cache | CY82C692 / CY82C690 | 208-pin PQFP |
| hyperCache Chipset Peripheral Controller | CY82C693 / CY82C693U | 208-pin PQFP |
| hyperCache SRAM Expansion | CY82C694 | 128-pin PQFPA |
| hyperCache Chipset Peripheral Controller with USB support | CY82C693U | 208-pin PQFP |

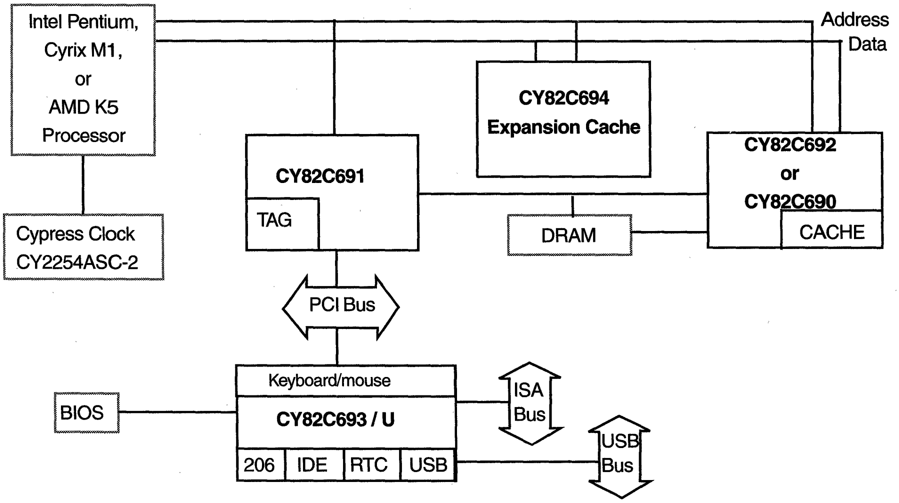
- 1992 Cypress developed a 386/486 single VESA/ ISA chip
- 1993 Cypress introduced the world's first PCI-VESA bridge
- 1994 Cypress updated the chipset with Deep Green features for VESA or VIP systems
- 1996 Cypress introduces the hyperCache™ Chipset for Pentium-class processors

The Cypress hyperCache Chipset is a family of three chipsets created to provide flexible solutions for today's x86 designs:

hC-ZX chipset: CY82C690, CY82C691, CY82C693/U *zero cache solution* with integrated microcache
hC-VX chipset: CY82C91, CY82C692, CY82C693/U *value solution* with integrated 128KB cache
hC-DX chipset: CY82C691, CY82C692, CY82C693/U, CY82C694 *performance solution* with integrated 256KB cache

The hC-ZX, hC-VX, hC-DX chipsets provide all the functions necessary to implement a 3.3V Pentium-based system with USB (Universal Serial Bus), PCI (Peripheral Component Interconnect), and ISA (Industry Standard Architecture) buses. The hyper-Cache Chipset includes a 2-way set associative cache tag (capable of addressing up to 512 MB of cache) along with integrated synchronous pipelined SRAM. Upgrading between the three chipsets is straight-forward since all solutions are pin-compatible. In addition to the integrated cache the chipset also integrates a RTC (Real-Time Clock) with 256 bytes of battery backed RAM, a PS/2 compatible mouse control, a 8042-compatible keyboard control, a dual-channel IDE controller, and two 8237-compatible DMA controllers.

**hyperCache™ Chipset System Block Diagram**



Note: Please contact a Cypress Representative for product availability.

## USB Microcontroller Family

Cypress's new family of 8-bit microcontrollers offers the lowest-cost solutions for USB peripherals. The family is built on the industry's smallest RISC core and are the first to offer EPROM programmability for easy code customization and fast time-to-market. The parts integrate data RAM, a USB Serial Interface Engine, and a transceiver for reduced cost and parts count, while a clock-doubler and Instant-On-Now features ensure low EMI and 70% lower power consumption. All solutions are supported by the CY365x emulator board.

### CY7C6xxxx Family

- Industry's smallest 8-bit RISC micro-controller core
- Integrated 128/256 bytes SRAM
- Integrated 2/4/6/8 KB EPROM
- Optimized USB instruction set (34 instructions)
- On-chip clock multiplication
- Full-custom USB Serial Interface Engine (SIE)
- Integrated USB transceiver
- 10-40 General Purpose I/Os
- Instant-On-Now Power Management

### CY365x Developer's Kit

- Hardware emulator board for real-time development of USB firmware and system drivers
- Register, RAM, and I/O display & modification
- User-configurable break-traps and single-stepping
- Access to key microcontroller signals for trace and complex break-points
- Stand-alone mode for portable operation
- Assembler software
- Complete support documentation
- USB code library
- USB reference Designs

### Low-speed microcontrollers (1.5Mbps)

| Part Number | RAM (Bytes) | EPROM (KB) | #I/Os | Packages |
|---|---|---|---|---|
| CY7C63000 | 128 | 2 | 12 | 20-pin PDIP, 20-pin SOIC |
| CY7C63001 | 128 | 4 | 12 | 20-pin PDIP, 20-pin SOIC |
| CY7C63100 | 128 | 2 | 16 | 24-pin SOIC |
| CY7C63101 | 128 | 4 | 16 | 24-pin SOIC |
| CY7C63200 | 128 | 2 | 10 | 18-pin PDIP |
| CY7C63201 | 128 | 4 | 10 | 18-pin PDIP |
| CY7C63410 | 128 | 2 | 32 | 40-pin PDIP, 48-pin SSOP |
| CY7C63411 | 128 | 4 | 32 | 40-pin PDIP, 48-pin SSOP |
| CY7C63412 | 128 | 6 | 32 | 40-pin PDIP, 48-pin SSOP |
| CY7C63413 | 128 | 8 | 32 | 40-pin PDIP, 48-pin SSOP |
| CY7C63510 | 128 | 2 | 40 | 48-pin SSOP |
| CY7C63511 | 128 | 4 | 40 | 48-pin SSOP |
| CY7C63512 | 128 | 6 | 40 | 48-pin SSOP |
| CY7C63513 | 128 | 8 | 40 | 48-pin SSOP |

### High-speed stand-alone hubs with integrated $I^2C$ (12Mbps)

| Part Number | RAM (Bytes) | EPROM (KB) | # hub ports | #I/Os | Packages |
|---|---|---|---|---|---|
| CY7C65013 | 256 | 8 | 8 | 23 | 48-pin PDIP, 48-pin SSOP |
| CY7C65113 | 256 | 8 | 4 | 11 | 28-pin PDIP, 28-pin SOIC |

### High-speed microcontroller + hub with integrated $I^2C$ (12Mbps)

| Part Number | RAM (Bytes) | EPROM (KB) | # hub ports | #I/Os | Packages |
|---|---|---|---|---|---|
| CY7C66011 | 256 | 4 | 4 | 31 | 48-pin PDIP, 48-pin SSOP |
| CY7C66012 | 256 | 6 | 4 | 31 | 48-pin PDIP, 48-pin SSOP |
| CY7C66013 | 256 | 8 | 4 | 31 | 48-pin PDIP, 48-pin SSOP |
| CY7C66111 | 256 | 4 | 4 | 39 | 56-pin SSOP |
| CY7C66112 | 256 | 6 | 4 | 39 | 56-pin SSOP |
| CY7C66113 | 256 | 8 | 4 | 39 | 56-pin SSOP |

### USB Emulator Board

| Part Number | Description | CY Parts Supported |
|---|---|---|
| CY3650 | Low-Speed Emulator Board (1.5 Mbps) | CY63xxx |
| CY3651 | High-Speed Emulator Board (12 Mbps) | CY65xxx, CY66xxx |

Note: Please contact a Cypress Representative for product availability.

## FCT-T Logic Family

The FCT-T CMOS logic family offers the lowest power solution for high speed logic designs. High Drive (64 mA), Balanced Drive (24 mA), or 25Ω Output (12 mA) devices provide a choice of drive capability to meet the applications specific need.

- TTL compatible inputs and outputs provide significant noise reduction over older FCT designs.
- All Cypress FCT-T devices have been designed for use in "Live Insertion" applications.
- All functions are available in space saving QSOP or TSSOP packages as well as the standard 300-mil SOIC.
- Extended temperature range (-40°C to +85°C) is standard on all devices.

### Octal High Drive Logic Products ($V_{CC}$=5 Volts)

| | | | Propagation Delay (ns) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | C | | B | | A | | Standard | | |
| Part Number | Organization | Pins | Com'l | Mil | Com'l | Mil | Com'l | Mil | Com'l | Mil | Package |
| CY29FCT52T | 8-Bit Registered Transceiver | 24 | 6.3 | | 7.5 | | 10.0 | | | | Q, SO |
| CY29FCT520T | Multilevel Pipeline Register | 24 | 6.0 | | 7.5 | 8.0 | 14.0 | 16.0 | | | D, P, Q, SO |
| CY29FCT818T | Diagnostic Scan Register | 24 | 6.0 | | | | 9.0 | 12.0 | | | D, L, P, Q, SO |
| CY54/74FCT138T | 1-of-8 Decoder | 16 | 5.0 | 6.0 | | | 5.8 | | 9.0 | | D, L, Q, SO |
| CY54/74FCT157T | Quad 2-input Multiplexers | 16 | 4.3 | 5.0 | | | 5.0 | 5.8 | 6.0 | | L, Q, SO |
| CY54/74FCT163T | 4-Bit Binary Counter with Synchronous Reset | 16 | 5.8 | 6.1 | | | 7.2 | 7.5 | 11.0 | 11.5 | L, Q, SO |
| CY74FCT191T | 4-Bit Up/Down Binary Counter | 16 | 6.2 | | | | 7.8 | | | | Q, SO |
| CY54/74FCT240T | 8-Bit Inverting Buffer/Line Driver with $\overline{OE}$ | 20 | 4.3 | 4.7 | | | 4.8 | 5.1 | 8.0 | 9.0 | D, P, Q, SO |
| CY54/74FCT244T | 8-Bit Buffer/Line Driver with $\overline{OE}$ | 20 | 4.1 | 4.6 | | | 4.8 | 5.1 | 6.5 | 7.0 | D, L, P, Q, SO |
| CY54/74FCT245T | 8-Bit Transceiver with $\overline{OE}$ | 20 | 4.1 | 4.5 | | | 4.6 | 4.9 | 7.0 | 7.5 | D, L, P, Q, SO |
| CY74FCT257T | Quad 2-input Multiplexers with $\overline{OE}$ | 16 | 4.3 | | | | 5.0 | | 6.0 | | P, Q, SO |
| CY54/74FCT273T | 8-Bit Register with Asynchronous Reset | 20 | 5.8 | 6.5 | | | 7.2 | 8.3 | 13.0 | 15.0 | D, L, P, Q, SO |
| CY54/74FCT373T | 8-Bit Latch with $\overline{OE}$ | 20 | 4.2 | 5.1 | | | 5.2 | 5.6 | 8.0 | 8.5 | D, L, P, Q, SO |
| CY54/74FCT374T | 8-Bit Register with $\overline{OE}$ | 20 | 5.2 | 6.2 | | | 6.5 | 7.2 | 10.0 | 11.0 | D, L, P, Q, SO |
| CY54/74FCT377T | 8-Bit Register with Clock Enable | 20 | 5.2 | 5.5 | | | 7.2 | 8.3 | 13.0 | 15.0 | L, Q, SO |
| CY74FCT399T | Quad 2-input Registers | 16 | 6.1 | | | | 7.0 | | 10.0 | | SO |
| CY54/74FCT480T | Dual 8-Bit Odd-Parity Generators/ Checkers | 24 | | | 5.6 | 7.0 | 7.5 | | | | D, L, P, Q, SO |
| CY54/74FCT540T | 8-Bit Inverting Buffer/Line Driver with OE and Flow-Through Pinout | 20 | 4.1 | 4.7 | | | 4.8 | 5.1 | 8.5 | 9.5 | D, Q, SO |
| CY54/74FCT541T | 8-Bit Buffer/Line Driver with $\overline{OE}$ and Flow-Through Pinout | 20 | 4.1 | 4.6 | | | 4.8 | 5.1 | 8.0 | 9.0 | D, P, Q, SO |
| CY54/74FCT543T | 8-Bit Latched Transceiver with $\overline{OE}$ | 24 | 5.3 | 6.1 | | | 6.5 | 7.5 | 8.5 | 10.0 | D, Q, SO |
| CY54/74FCT573T | 8-Bit Latch with $\overline{OE}$ and Flow-Through Pinout | 20 | 4.2 | 5.1 | | | 5.2 | 5.6 | 8.0 | 8.5 | D, P, Q, SO |
| CY54/74FCT574T | 8-Bit Register with $\overline{OE}$ and Flow-Through Pinout | 20 | 5.2 | 6.2 | | | 6.5 | 7.2 | 10.0 | 11.0 | D, L, P, Q, SO |
| CY54/74FCT646T | 8-Bit Registered Transceiver with $\overline{OE}$ | 24 | 5.4 | 6.0 | | | 6.3 | 7.7 | 9.0 | 11.0 | D, L, Q, SO |
| CY54/74FCT652T | 8-Bit Registered Transceiver with $\overline{OE}$ | 24 | 5.4 | | | | 6.3 | | 9.0 | | Q, SO |
| CY54/74FCT821T | 10-Bit Register with $\overline{OE}$ | 24 | 6.0 | | 7.5 | | 10.0 | | | | P, Q, SO |
| CY54/74FCT823T | 9-Bit Register with $\overline{OE}$ | 24 | 6.0 | | 7.5 | | 10.0 | | | | P, Q, SO |
| CY54/74FCT825T | 8-Bit Register with $\overline{OE}$ | 24 | 6.0 | | 7.5 | | 10.0 | | | | Q, SO |
| CY54/74FCT827T | 10-Bit Buffer with $\overline{OE}$ | 24 | 4.4 | | 5.0 | | 8.0 | | | | P, Q, SO |
| CY54/74FCT841T | 10-Bit Latch with $\overline{OE}$ | 24 | 5.5 | 6.3 | 6.5 | | 9.0 | 10.0 | | | D, P, Q, SO |

Note: Please contact a Cypress Representative for product availability.

## FCT-T Logic Family (continued)

### Bus Switch

| Part Number | Organization | Pins | Propagation Delay (ns) | Packages |
|---|---|---|---|---|
| CYBUS3384 | 10-Bit Bus Switch | 24 | 0.25 | Q, SO |

### Octal Logic Products with Resistor ($V_{CC}$=5 Volts)

| Part Number | Organization | Pins | Propagation Delay (ns) | | | | Packages |
|---|---|---|---|---|---|---|---|
| | | | C | B | A | Standard | |
| | | | Com'l | Com'l | Com'l | Com'l | |
| CY74FCT2240T | 8-Bit Inverting Buffer/Line Driver with $\overline{OE}$ and 25Ω Resistor | 20 | 4.1 | | 4.8 | 8.0 | Q, SO |
| CY74FCT2244T | 8-Bit Buffer/Line Driver with $\overline{OE}$ and 25Ω Resistor | 20 | 4.1 | | 4.6 | 6.5 | P, Q, SO |
| CY74FCT2245T | 8-Bit Transceiver with $\overline{OE}$ and 25Ω Resistor | 20 | 4.1 | | 4.6 | 7.0 | P, Q, SO |
| CY74FCT2257T | Quad 2-input Multiplexers with OE and 25Ω Resistor | 20 | 4.3 | | 5.0 | 6.0 | Q, SO |
| CY74FCT2373T | 8-Bit Latch with $\overline{OE}$ and 25Ω Resistor | 20 | 4.2 | | 5.2 | 8.0 | Q, SO |
| CY74FCT2374T | 8-Bit Register with $\overline{OE}$ and 25Ω Resistor | 20 | 5.2 | | 6.5 | 10.0 | Q, SO |
| CY74FCT2541T | 8-Bit Buffer/Line Driver with $\overline{OE}$, Flow-Through Pinout and 25Ω Resistor | 20 | 4.1 | | 4.8 | 8.0 | Q, SO |
| CY74FCT2543T | 8-Bit Latched Transceiver with $\overline{OE}$ and 25Ω Resistor | 20 | 5.5 | | 6.5 | 8.5 | Q, SO |
| CY74FCT2573T | 8-Bit Latch with $\overline{OE}$, Flow-Through Pinout and 25Ω Resistor | 20 | 4.2 | | 5.2 | 8.0 | Q, SO |
| CY74FCT2574T | 8-Bit Register with $\overline{OE}$, Flow-Through Pinout and 25Ω Resistor | 20 | 5.2 | | 6.5 | 10.0 | Q, SO |
| CY74FCT2646T | 8-Bit Registered Transceiver with $\overline{OE}$ and 25Ω Resistor | 20 | 5.4 | | 6.3 | 9.0 | Q |
| CY74FCT2652T | 8-Bit Registered Transceiver with $\overline{OE}$ and 25Ω Resistor | 20 | 5.4 | | 6.3 | 9.0 | Q, SO |
| CY74FCT2827T | 10-Bit Buffer with $\overline{OE}$ and 25Ω Resistor | 20 | 4.4 | 5.0 | 8.0 | | Q, SO |

1

Note: Please contact a Cypress Representative for product availability.

## FCT-T Logic Family (continued)

### 16-Bit High Drive Logic Products ($V_{CC}$=5 Volts)

| | | | Propagation Delay (ns) | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | E | C | B | A | Standard | |
| Part Number | Organization | Pins | Com'l | Com'l | Com'l | Com'l | Com'l | Package |
| CY74FCT16240T | 16-Bit Inverting Buffer/Line Driver with $\overline{\text{OE}}$ | 48 | 3.2 | 4.3 | | 4.8 | 8.0 | PA, PV |
| CY74FCT16244T | 16-Bit Buffer/Line Driver with $\overline{\text{OE}}$ | 48 | 3.2 | 4.1 | | 4.8 | 6.5 | PA, PV |
| CY74FCT16245T | 16-Bit Transceiver with $\overline{\text{OE}}$ | 48 | 3.2 | 4.1 | | 4.5 | 7.0 | PA, PV |
| CY74FCT16373T | 16-Bit Latch with $\overline{\text{OE}}$ | 48 | 3.4 | 4.2 | | 5.2 | 8.0 | PA, PV |
| CY74FCT16374T | 16-Bit Register with $\overline{\text{OE}}$ | 48 | 3.7 | 5.2 | | 6.5 | 10.0 | PA, PV |
| CY74FCT16444T | 16-Bit 244 with Single $\overline{\text{OE}}$ | 48 | | 4.1 | | 4.8 | 6.5 | PA, PV |
| CY74FCT16445T | 16-Bit 245 with Single $\overline{\text{OE}}$ and DIR | 48 | | 4.1 | | 4.5 | 7.0 | PA, PV |
| CY74FCT16500T | 18-Bit Universal Bus Transceiver | 56 | | 4.6 | | 5.1 | | PA, PV |
| CY74FCT16501T | 18-Bit Universal Bus Transceiver | 56 | 3.8 | 4.6 | | 5.1 | | PA, PV |
| CY74FCT16543T | 16-Bit Latched Transceiver with $\overline{\text{OE}}$ | 56 | 3.4 | 5.1 | | 6.5 | 8.5 | PA, PV |
| CY74FCT16646T | 16-Bit Registered Transceiver with $\overline{\text{OE}}$ | 56 | 3.8 | 5.4 | | 6.3 | 9.0 | PA, PV |
| CY74FCT16652T | 16-Bit Registered Transceiver with $\overline{\text{OE}}$ | 56 | 3.8 | 5.4 | | 6.3 | 9.0 | PA, PV |
| CY74FCT16823T | 18-Bit Register with $\overline{\text{OE}}$ | 56 | 4.4 | 6.0 | 7.5 | 10.0 | | PA, PV |
| CY74FCT16827T | 20-Bit Buffer with $\overline{\text{OE}}$ | 56 | 3.2 | 4.2 | 5.0 | 8.0 | | PA, PV |
| CY74FCT16841T | 20-Bit Latch with $\overline{\text{OE}}$ | 56 | | 5.5 | 6.5 | 9.0 | | PA, PV |
| CY74FCT16952T | 16-Bit Registered Transceiver | 56 | 3.7 | 6.3 | 7.5 | 10.0 | | PA, PV |

### 16-Bit Balanced Drive Logic Products ($V_{CC}$=5 Volts)

| | | | Propagation Delay (ns) | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | E | C | B | A | Standard | |
| Part Number | Organization | Pins | Com'l | Com'l | Com'l | Com'l | Com'l | Package |
| CY74FCT162240T | 16-Bit Inverting Buffer/Line Driver with $\overline{\text{OE}}$ | 48 | 3.2 | 4.3 | | 4.8 | 8.0 | PA, PV |
| CY74FCT162244T | 16-Bit Buffer/Line Driver with $\overline{\text{OE}}$ | 48 | 3.2 | 4.1 | | 4.8 | 6.5 | PA, PV |
| CY74FCT162245T | 16-Bit Transceiver with $\overline{\text{OE}}$ | 48 | 3.2 | 4.1 | | 4.5 | 7.0 | PA, PV |
| CY74FCT162373T | 16-Bit Latch with $\overline{\text{OE}}$ | 48 | 3.4 | 4.2 | | 5.2 | 8.0 | PA, PV |
| CY74FCT162374T | 16-Bit Register with $\overline{\text{OE}}$ | 48 | 3.7 | 5.2 | | 6.5 | 10.0 | PA, PV |
| CY74FCT162500T | 18-Bit Universal Bus Transceiver | 56 | | 4.6 | | 5.1 | | PA, PV |
| CY74FCT162501T | 18-Bit Universal Bus Transceiver | 56 | 3.8 | 4.6 | | 5.1 | | PA, PV |
| CY74FCT162543T | 16-Bit Latched Transceiver with $\overline{\text{OE}}$ | 56 | 3.4 | 5.1 | | 6.5 | 8.5 | PA, PV |
| CY74FCT162646T | 16-Bit Registered Transceiver with $\overline{\text{OE}}$ | 56 | 3.8 | 5.4 | | 6.3 | 9.0 | PA, PV |
| CY74FCT162652T | 16-Bit Registered Transceiver with $\overline{\text{OE}}$ | 56 | 3.8 | 5.4 | | 6.3 | 9.0 | PA, PV |
| CY74FCT162823T | 18-Bit Register with $\overline{\text{OE}}$ | 56 | 4.4 | 6.0 | 7.5 | 10.0 | | PA, PV |
| CY74FCT162827T | 20-Bit Buffer with $\overline{\text{OE}}$ | 56 | 3.2 | 4.2 | 5.0 | 8.0 | | PA, PV |
| CY74FCT162841T | 20-Bit Latch with $\overline{\text{OE}}$ | 56 | | 5.5 | 6.5 | 9.0 | | PA, PV |
| CY74FCT162952T | 16-Bit Registered Transceiver | 56 | 3.7 | 6.3 | 7.5 | 10.0 | | PA, PV |

Note: Please contact a Cypress Representative for product availability.

## FCT-T Logic Family (continued)

**16-Bit Balanced Drive, Bus Hold Logic Products** ($V_{CC}$=5 Volts)

| | | | Propagation Delay (ns) | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | E | C | B | A | Standard | |
| Part Number | Organization | Pins | Com'l | Com'l | Com'l | Com'l | Com'l | Package |
| CY74FCT162H244T | 16-Bit Buffer/Line Driver with $\overline{OE}$ with Bus Hold | 48 | 3.2 | 4.1 | | 4.8 | 6.5 | PA, PV |
| CY74FCT162H245T | 16-Bit Transceiver with $\overline{OE}$ with Bus Hold | 48 | 3.2 | 4.1 | | 4.5 | 7.0 | PA, PV |
| CY74FCT162H501T | 18-Bit Universal Bus Transceiver with Bus Hold | 56 | 3.4 | 4.6 | | 5.1 | | PA, PV |
| CY74FCT162H952T | 16-Bit Registered Transceiver with Bus Hold | 56 | 3.7 | 6.3 | 7.5 | 10.0 | | PA, PV |

## 3.3V FCT Logic Family

The 3.3V FCT CMOS logic family offers the lowest power solution for high speed logic designs. 5V tolerant I/O's (standard on all Cypress 3.3V FCT logic) are ideal for use in mixed voltage systems.

- Inputs and outputs are 5V tolerant.
- Extended VCC range of 2.7V to 3.6V
  Derate AC specs by 20% for VCC < 3.0V.
- +/- 24 mA drive current.
- All functions are available in TSSOP or SSOP packages.
- Extended temperature range (–40°C to +85°C) on all devices.

**16-Bit 3.3V Balanced Drive Logic Products** ($V_{CC}$=3.3 V +/-10%)

| | | | Propagation Delay (ns) | | |
|---|---|---|---|---|---|
| | | | C | A | |
| Part Number | Organization | Pins | Com'l | Com'l | Package |
| CY74FCT163244 | 16-Bit Buffer/Line Driver with $\overline{OE}$ | 48 | 4.1 | 4.8 | PA, PV |
| CY74FCT163H244 | 16-Bit Buffer/Line Driver with $\overline{OE}$ and Bus Hold | 48 | 4.1 | 4.8 | PA, PV |
| CY74FCT163245 | 16-Bit Transceiver with $\overline{OE}$ | 48 | 4.1 | 4.6 | PA, PV |
| CY74FCT163H245 | 16-Bit Transceiver with $\overline{OE}$ and Bus Hold | 48 | 4.1 | 4.6 | PA, PV |
| CY74FCT163373 | 16-Bit Latch with $\overline{OE}$ | 48 | 4.2 | 5.2 | PA, PV |
| CY74FCT163H373 | 16-Bit Latch with $\overline{OE}$ and Bus Hold | 48 | 4.2 | 5.2 | PA, PV |
| CY74FCT163374 | 16-Bit Register with $\overline{OE}$ | 48 | 5.2 | 6.5 | PA, PV |
| CY74FCT163H374 | 16-Bit Register with $\overline{OE}$ and Bus Hold | 48 | 5.2 | 6.5 | PA, PV |
| CY74FCT163501 | 18-Bit Universal Bus Transceiver | 56 | 4.1 | 4.6 | PA, PV |
| CY74FCT163H501 | 18-Bit Universal Bus Transceiver with Bus Hold | 56 | 4.1 | 4.6 | PA, PV |
| CY74FCT163646 | 16-Bit Registered Transceiver with $\overline{OE}$ | 56 | 4.6 | 5.1 | PA, PV |
| CY74FCT163652 | 16-Bit Registered Transceiver with $\overline{OE}$ | 56 | 5.3 | 6.5 | PA, PV |
| CY74FCT163827 | 20-Bit buffer with OE | 56 | 4.2 | 8.0 | PA, PV |
| CY74FCT163952 | 16-Bit Registered Transceiver | 56 | 4.4 | 8.0 | PA, PV |
| CY74FCT163H952 | 16-Bit Registered Transceiver with Bus Hold | 56 | 4.4 | 8.0 | PA, PV |

Note: Please contact a Cypress Representative for product availability.

**Package Code:**

| | |
|---|---|
| A = | Thin Quad Flat Pack (TQFP) |
| B = | Plastic Pin Grid Array |
| D = | CerDIP |
| E = | Tape Automated Bond (TAB) |
| F = | Flatpack |
| G = | Pin Grid Array (PGA) |
| H = | Windowed Hermetic LCC |
| J = | PLCC |
| K = | Cerpack |
| L = | Leadless Chip Carrier (LCC) |
| N = | Plastic Quad Flatpack |
| P = | Plastic |
| Q = | Windowed LCC |
| Q = | QSOP |
| R = | Windowed PGA |
| S = | SOIC |
| T = | Windowed Cerpack |
| U = | Ceramic Quad Flatpack |
| V = | SOJ |
| W = | Windowed Cerdip |
| X = | DICE |
| Y = | Ceramic LCC |
| Z = | TSOP |
| HD = | Hermetic DIP (Module) |
| HG = | Ceramic PGA (Module) |
| PA = | TSSOP |
| PD = | Plastic DIP (Module) |
| PM = | Plastic SIMM |
| PN = | Plastic Angled SIMM |
| PS = | Plastic SIP |
| PV = | SSOP |
| PZ = | Plastic ZIP |
| SO = | SOIC |

**Notes:**

Military temperature range (-55°C to +125°C) product processed to MIL-STD-883 Revision C is also available for most products. Speed and power selections may vary from those above. Contact your local sales office for more information.

Commercial grade product is available in plastic, CerDIP, or LCC. Military grade product is available in CerDIP, LCC, or PGA.

Power supplies for most product lines are $V_{CC}$ = 5V 10%.

22S, 24S, 28S stands for 300 mil. 22-pin, 24-pin, 28-pin, respectively. 28.4 stands for 28-pin 400 mil, 24.4 stands for 24-pin 400 mil.

PLCC, SOJ, and SOIC packages are available on some products.

F, K, and T packages are special order only.
Please contact a Cypress representative for product availability.

MAX and MAX+PLUS are registered trademarks of Altera Corporation.
Pentium is a trademark of Intel Corporation.
*Warp*, UltraLogic, pASIC380, *Impulse3*, and hyperCache are trademarks of Cypress Semiconductor Corporation.
GAL is a registered trademark of Lattice Semiconductor.

Note: Please contact a Cypress Representative for product availability.

| CYPRESS | CYPRESS |
|---|---|
| 2147-35C | 7C147-35C |
| 2147-45C | 2147-35C |
| 2147-45C | 7C147-45C |
| 2147-45M+ | 7C147-45M+ |
| 2147-55C | 2147-45C |
| 2147-55M | 2147-45M |
| 2148-35C | 21L48-35C |
| 2148-35C | 7C148-35C |
| 2148-35M | 7C148-35M |
| 2148-45C | 2148-35C |
| 2148-45C | 21L48-45C |
| 2148-45M | 2148-35M |
| 2148-45M+ | 7C148-45M+ |
| 2148-55C | 2148-45C |
| 2148-55C | 21L48-55C |
| 2148-55M | 2148-45M |
| 2149-35C | 21L49-35C |
| 2149-35C | 7C149-35C |
| 2149-35M | 7C149-35M |
| 2149-45C | 21L49-45C |
| 2149-45M | 2149-35M |
| 2149-45M | 7C149-45M |
| 2149-55C | 2149-45C |
| 2149-55C | 21L49-55C |
| 2149-55M | 2149-45M |
| 21L48-35C | 7C148-35C |
| 21L48-45C | 21L48-35C |
| 21L48-45C | 7C148-45C |
| 21L48-55C | 21L48-45C |
| 21L49-35C | 7C149-25C |
| 21L49-45C | 21L49-35C |
| 21L49-45C | 7C149-45C |
| 21L49-55C | 21L49-45C |
| 27S03AC | 7C189-25C |
| 27S03AM | 7C189-25M |
| 27S03C | 27S03AC |
| 27S03C | 74S189C |
| 27S03M | 27S03AM |
| 27S03M | 54S189M |
| 27S07AC | 7C190-25C |
| 27S07AM | 7C190-25M |
| 27S07C | 27S07AC |

| CYPRESS | CYPRESS |
|---|---|
| 27S07M | 27S07AM |
| 27S07M | 7C190-25M |
| 54S189M | 27S03M |
| 6116A-45C | 6116A-35C |
| 6116A-55C | 6116A-45C |
| 6116A-55M | 6116A-45M |
| 74S189C | 27S03C |
| 7C122-25C | 7C122-15C+ |
| 7C122-35C | 7C122-25C |
| 7C122-35M | 7C122-25M |
| 7C123-12C | 7C123-7C |
| 7C128A-35C | 7C128A-25C |
| 7C128A-45C | 7C128A-35C |
| 7C128A-45M | 7C128A-35M+ |
| 7C128A-55C | 7C128A-45C+ |
| 7C128A-55M | 7C128A-45M+ |
| 7C147-35C | 7C147-25C+ |
| 7C147-45C | 7C147-35C |
| 7C148-35C | 7C148-25C+ |
| 7C148-45C | 7C148-35C |
| 7C149-35C | 7C149-25C+ |
| 7C149-45C | 7C149-35C |
| 7C149-45M | 7C149-35M |
| 7C150-25C | 7C150-15C |
| 7C150-35C | 7C150-25C |
| 7C150-35M | 7C150-25M |
| 7C167A-35C | 7C167A-25C |
| 7C167A-45M | 7C167A-35M+ |
| 7C168A-35C | 7C168A-25C |
| 7C168A-45M | 7C168A-35M+ |
| 7C169A-35C | 7C169A-25C |
| 7C169A-40M | 7C169A-35M+ |
| 7C170A-35C | 7C170A-25C |
| 7C170A-45C | 7C170A-35C |
| 7C170A-45M | 7C170A-35M |
| 7C171A-35C | 7C171A-25C |
| 7C171A-45M | 7C171A-35M+ |
| 7C172A-35C | 7C172A-25C |
| 7C172A-45M | 7C172A-35M+ |
| 7C189-25C | 7C189-15C+ |
| 7C190-25C | 7C190-15C+ |
| 7C191-45M | 7C191-35M |

| CYPRESS | CYPRESS |
|---|---|
| 7C192-45M | 7C192-35M |
| 7C194-35C | 7C194-25C |
| 7C194-45C | 7C194-35C+ |
| 7C194-45M | 7C194-35M |
| 7C196-35C | 7C196-25C |
| 7C196-45C | 7C196-35C+ |
| 7C197-35C | 7C197-25C |
| 7C197-45C | 7C197-35C+ |
| 7C197-45M | 7C197-35M |
| 7C198-45C | 7C198-35C |
| 7C198-55C | 7C198-45C+ |
| 7C198-55M | 7C198-45M |
| 7C199-45C | 7C199-35C |
| 7C199-55C | 7C199-45C+ |
| 7C199-55M | 7C199-45M |
| 7C225 | 7C225A |
| 7C235 | 7C235A |
| 7C245 | 7C245A |
| 7C271 | 7C271A |
| 7C274 | 27H256 |
| 7C281 | 7C281A |
| 7C286 | 27H512 |
| 7C291 | 7C291A |
| 7C292 | 7C292A |
| 9122-25C | 7C122-15C |
| 9122-25C | 91L22-25C |
| 9122-35C | 9122-25C |
| 9122-35C | 91L22-35C |
| 9122-45C | 93L422C |
| 91L22-25C | 7C122-25C |
| 91L22-35C | 7C122-35C |
| 91L22-45C | 93L422AC |
| 93422AC | 7C122-35C |
| 93422AC | 9122-35C |
| 93422AM | 7C122-35M |
| 93422C | 93L422AC |
| 93422M | 93422AM |
| 93422M | 93L422AM |
| 93L422AC | 7C122-35C |
| 93L422AC | 91L22-45C |
| 93L422AM | 7C122-35M |
| 93L422C | 93L422AC |

| CYPRESS | CYPRESS |
|---|---|
| 93L422M | 93L422AM |
| PALC16L8-25C | PALC16L8L-25C |
| PALC16L8-30M | PALC16L8-20M |
| PALC16L8-35C | PALC16L8-25C |
| PALC16L8-40M | PALC16L8-30M |
| PALC16L8L-35C | PALC16L8L-25C |
| PALC16R4-25C | PALC16R4L-25C |
| PALC16R4-30M | PALC16R4-20M |
| PALC16R4-35C | PALC16R4-25C |
| PALC16R4-40M | PALC16R4-30M |
| PALC16R4L-35C | PALC16R4L-25C |
| PALC16R6-25C | PALC16R6L-25C |
| PALC16R6-30M | PALC16R6-20M |
| PALC16R6-35C | PALC16R6-25C |
| PALC16R6-40M | PALC16R6-30M |
| PALC16R6L-35C | PALC16R6L-25C |
| PALC16R8-25C | PALC16R8L-25C |
| PALC16R8-30M | PALC16R8-20M |
| PALC16R8-35C | PALC16R8-25C |
| PALC16R8-40M | PALC16R8-30M |
| PALC16R8L-35C | PALC16R8L-25C |
| PALC22V10-35C | PALC22V10-25C |
| PALC22V10-40M | PALC22V10-30M |
| PALC22V10L-25C | PALC22V10-25C |
| PALC22V10L-35C | PALC22V10L-25C |
| PLDC20G10-35C | PLDC20G10-25C |
| PLDC20G10-40M | PLDC20G10-30M |

| ALLIANCE | CYPRESS |
|---|---|
| PREFIX: AS | PREFIX: CY |
| 7C1024 | 7C1009 |
| 7C1028 | 7C1006 |
| 7C259 | 7C188 |
| 7C3256 | 7C1399 |
| 7C33232F-5 | 7C1335-100A |
| 7C33232F-6 | 7C1335-75AC |
| 7C33232F-7 | 7C1335-66AC |
| 7C3512 | 7C1512 |

| ALTERA | CYPRESS |
|---|---|
| PREFIX EPM | PREFIX: CY |

| ALTERA | CYPRESS |
|---|---|
| PREFIX: EP | PREFIX: PALC |
| 22V10-10C | PALC22V10D-7C |
| 22V10-10C | PALC22V10D-10C |
| 22V10-10C | PAL22V10C-7C+ |
| 22V10-10C | PAL22V10C-10C+ |
| 22V10-15C | PALC22V10B-15C |
| 22V10-15C | PALC22V10D-15C |
| 5032DC | 7C344-25WC |
| 5032DC-2 | 7C344-20WC |
| 5032DC-15 | 7C344-15WC |
| 5032DC-17 | Call Factory |
| 5032DC-20 | 7C344-20WC |
| 5032DC-25 | 7C344-25WC |
| 5032DM | 7C344-25WMB |
| 5032DM-25 | 7C344-25WMB |
| 5032JC | 7C344-25HC |
| 5032JC-2 | 7C344-20HC |
| 5032JC-15 | 7C344-15HC |
| 5032JC-17 | Call Factory |
| 5032JC-20 | 7C344-20HC |
| 5032JC-25 | 7C344-25HC |
| 5032JI-20 | 7C344-20HI |
| 5032JM | 7C344-25HMB |
| 5032JM-25 | 7C344-25HMB |
| 5032LC | 7C344-25JC |
| 5032LC-2 | 7C344-20JC |
| 5032LC-15 | 7C344-15JC |
| 5032LC-17 | Call Factory |
| 5032LC-20 | 7C344-20JC |
| 5032LC-25 | 7C344-25JC |
| 5032PC | 7C344-25PC |
| 5032PC-2 | 7C344-20PC |
| 5032PC-15 | 7C344-15PC |
| 5032PC-17 | Call Factory |
| 5032PC-20 | 7C344-20PC |
| 5032PC-25 | 7C344-25PC |
| 5064JC | 7C343-35HC |
| 5064JC-1 | 7C343-25HC |
| 5064JC-2 | 7C343-30HC |
| 5064JI | 7C343-35HI |
| 5064JM | 7C343-35HMB |
| 5064LC | 7C343-35JC |
| 5064LC-1 | 7C343-25JC |
| 5064LC-2 | 7C343-30JC |
| 5128AGC-12 | 7C342B-12RC |
| 5128AGC-15 | 7C342B-15RC |
| 5128AGC-20 | 7C342B-20RC |
| 5128AJC-12 | 7C342B-12HC |
| 5128AJC-15 | 7C342B-15HC |
| 5128AJC-20 | 7C342B-20HC |
| 5128ALC-12 | 7C342B-12JC |
| 5128ALC-15 | 7C342B-15JC |
| 5128ALC-20 | 7C342B-20JC |
| 5128GC | 7C342-35RC |
| 5128GC-1 | 7C342-25RC |
| 5128GC-2 | 7C342-30RC |
| 5128GM | 7C342-35RMB |
| 5128JC | 7C342-35HC |
| 5128JC-1 | 7C342-25HC |
| 5128JC-2 | 7C342-30HC |
| 5128JI | 7C342-35HI |
| 5128JI-2 | 7C342-30HI |
| 5128JM | 7C342-35HMB |
| 5128LC | 7C342-35JC |
| 5128LC-1 | 7C342-25JC |
| 5128LC-2 | 7C342-30JC |
| 5128LI | 7C342-35JI |
| 5128LI-2 | 7C342-30HI |
| 5130GC | 7C346-35RC |
| 5130GC-1 | 7C346-25RC |
| 5130GC-2 | 7C346-30RC |
| 5130GM | 7C346-35RM |
| 5130JC | 7C346-35HC |
| 5130JC-1 | 7C346-25HC |
| 5130JC-2 | 7C346-30HC |
| 5130JM | 7C346-35HM |
| 5130LC | 7C346-35JC |
| 5130LC-1 | 7C346-25JC |
| 5130LC-2 | 7C346-30JC |
| 5130LI | 7C346-35JI |
| 5130LI-2 | 7C346-30JI |
| 5130QC | 7C346-35NC |
| 5130QC-1 | 7C346-25NC |
| 5130QC-2 | 7C346-30NC |

| ALTERA | CYPRESS |
|---|---|
| 5130QI | 7C346-35NI |
| 5192AGC-15 | 7C341B-15RC |
| 5192AGC-20 | 7C341B-20RC |
| 5192AJC-15 | 7C341B-15HC |
| 5192AJC-20 | 7C341B-20HC |
| 5192ALC-1 | 7C341B-15JC |
| 5192ALC-2 | 7C341B-20JC |
| 5192GC | 7C341-35RC |
| 5192GC-1 | 7C341-25RC |
| 5192GC-2 | 7C341-30RC |
| 5192JC | 7C341-35HC |
| 5192JC-1 | 7C341-25HC |
| 5192JC-2 | 7C341-30HC |
| 5192JI | 7C341-35HI |
| 5192LC | 7C341-35JC |
| 5192LC-1 | 7C341-25JC |
| 5192LC-2 | 7C341-30JC |

| AMD | CYPRESS |
|---|---|
| PREFIX: Am | PREFIX: CY |
| PREFIX: SN | PREFIX: CY |
| SUFFIX: B | SUFFIX: B |
| SUFFIX: D | SUFFIX: W |
| SUFFIX: E | SUFFIX: Z |
| SUFFIX: F | SUFFIX: F |
| SUFFIX: J | SUFFIX: J |
| SUFFIX: L | SUFFIX: L |
| SUFFIX: P | SUFFIX: P |
| 5962-85155 01RX | 5962-88713 09RX |
| 5962-85155 02RX | 5962-88713 10RX |
| 5962-85155 03RX | 5962-88713 11RX |
| 5962-85155 04RX | 5962-88713 12RX |
| 5962-85155 05RX | 5962-88713 09RX |
| 5962-85155 06RX | 5962-88713 10RX |
| 5962-85155 07RX | 5962-88713 11RX |
| 5962-85155 08RX | 5962-88713 12RX |
| 5962-85155 012X | 5962-88713 09XX |
| 5962-85155 022X | 5962-88713 10XX |
| 5962-85155 032X | 5962-88713 11XX |
| 5962-85155 042X | 5962-88713 12XX |
| 5962-85155 052X | 5962-88713 09XX |

| AMD | CYPRESS |
|---|---|
| 5962-85155 062X | 5962-88713 10XX |
| 5962-85155 072X | 5962-88713 11XX |
| 5962-85155 082X | 5962-88713 12XX |
| 5962-86053 01LA | 5962-89841 01LX |
| 5962-86053 01KA | 5962-89841 01KX |
| 5962-86053 02LA | 5962-89841 01LX |
| 5962-86053 02KA | 5962-89841 01KX |
| 5962-86053 04LA | 5962-89841 02LX |
| 5962-86053 04KA | 5962-89841 02KX |
| 5962-86053 05KA | 5962-89841 06KX |
| 5962-86053 05LA | 5962-89841 06LX |
| 5962-86053 013A | 5962-89841 013X |
| 5962-86053 023A | 5962-89841 013X |
| 5962-86053 043A | 5962-89841 023X |
| 5962-86053 053A | 5962-89841 063X |
| 5962-88515 01RX | 5962-88713 09RX |
| 5962-88515 02RX | 5962-88713 10RX |
| 5962-88515 03RX | 5962-88713 11RX |
| 5962-88515 04RX | 5962-88713 12RX |
| 5962-88515 012X | 5962-88713 09XX |
| 5962-88515 022X | 5962-88713 10XX |
| 5962-88515 032X | 5962-88713 11XX |
| 5962-88515 042X | 5962-88713 12XX |
| 7205A-15 | 7C460-15 |
| 7204A-15 | 7C433A-15 |
| 7203A-15 | 7C429A-15 |
| 7202A-15 | 7C425A-15 |
| 7205A-25 | 7C460-25 |
| 7204A-25 | 7C432-25 |
| 7203A-25 | 7C428-25 |
| 7203A-25R | 7C429-25 |
| 7202A-25 | 7C424-25 |
| 7202A-25R | 7C425-25 |
| 7201-25 | 7C420-25 |
| 7201-25R | 7C421-25 |
| 7204A-35 | 7C432-30 |
| 7203A-35 | 7C428-30 |
| 7203A-35R | 7C429-30 |
| 7202A-35 | 7C424-30 |
| 7202A-35R | 7C425-30 |
| 7201-35 | 7C420-30 |
| 7201-35R | 7C421-30 |

| AMD | CYPRESS |
|---|---|
| 7204A-50 | 7C432-40 |
| 7203A-50 | 7C428-40 |
| 7203A-50R | 7C429-40 |
| 7202A-50 | 7C424-40 |
| 7202A-50R | 7C425-40 |
| 7201-50 | 7C420-40 |
| 7201-50R | 7C421-40 |
| 7204A-65 | 7C432-65 |
| 7203A-65 | 7C428-65 |
| 7203A-65R | 7C429-65 |
| 7202A-65 | 7C424-65 |
| 7202A-65R | 7C425-65 |
| 7201-65 | 7C420-65 |
| 7201-65R | 7C421-65 |
| 7204A-80 | 7C432-65 |
| 7203A-80 | 7C428-65 |
| 7203A-80R | 7C429-65 |
| 7202A-80 | 7C424-65 |
| 7202A-80R | 7C425-65 |
| 7201-80 | 7C420-65 |
| 7201-80R | 7C421-65 |
| 79C981 | 7C981 |
| 27LS291M | 7C291A-35M |
| 27PS191AC | 7C292A-50C |
| 27PS191AM | 7C292A-50M+ |
| 27PS191C | 7C292A-50C |
| 27PS191M | 7C292A-50M+ |
| 27PS291AC | 7C293A-50C |
| 27PS291AM | 7C293A-50M+ |
| 27PS291C | 7C293A-50C |
| 27PS291M | 7C293A-50M+ |
| 27S181AC | 7C282A-30C |
| 27S181AM | 7C282A-45M |
| 27S181C | 7C282A-45C |
| 27S181M | 7C282A-45M |
| 27S191AC | 7C292A-35C |
| 27S191AM | 7C292A-50M |
| 27S191C | 7C292A-50C |
| 27S191M | 7C292A-50M |
| 27S191SAC | 7C292A-25C |
| 27S191SAM | 7C292A-30M |
| 27S25AC | 7C225A-30C |

| AMD | CYPRESS |
|---|---|
| 27S25AM | 7C225A-35M |
| 27S25C | 7C225A-40C |
| 27S25M | 7C225A-40M |
| 27S25SAC | 7C225A-25C |
| 27S25SAM | 7C225A-30M |
| 27S43AC | 7C244-45C |
| 27S43C | 7C244-55C |
| 27S281AC | 7C281A-30C |
| 27S281AM | 7C281A-45M |
| 27S281C | 7C281A-45C |
| 27S281M | 7C281A-45M |
| 27S291AC | 7C291A-35C |
| 27S291AM | 7C291A-50M |
| 27S291C | 7C291A-50C |
| 27S291M | 7C291A-50M |
| 27S291SAC | 7C291A-25C |
| 27S291SAM | 7C291A-30M |
| 27S35AC | 7C235A-30C |
| 27S35AM | 7C235A-40M |
| 27S35C | 7C235A-40C |
| 27S35M | 7C235A-40M |
| 27S45AC | 7C245A-35C |
| 27S45AM | 7C245A-45M |
| 27S45C | 7C245A-45C |
| 27S45M | 7C245A-45M |
| 27S45SAC | 7C245A-25C |
| 27S45SAM | 7C245A-25M- |
| 27S49A | 7C264-40C |
| 27S49AM | 7C264-55M |
| 27S49C | 7C264-55C |
| 27S49M | 7C264-55M |
| 27S49SAC | 7C264-25C |
| 27S49SAM | 7C264-25M |
| PALCE16V8H-5JC/4 | PALCE16V8-5JC |
| PALCE16V8H-7JC/4 | PALCE16V8-7JC |
| PALCE16V8H-7PC/4 | PALCE16V8-7PC |
| PALCE16V8H-10JC/4 | PALCE16V8-10JC |
| PALCE16V8H-10PC/4 | PALCE16V8-10PC |
| PALCE16V8H-15JC/4 | PALCE16V8-15JC |
| PALCE16V8H-15PC/4 | PALCE16V8-15PC |

| AMD | CYPRESS |
|---|---|
| PALCE16V8H-25JC/4 | PALCE16V8-25JC |
| PALCE16V8H-25PC/4 | PALCE16V8-25PC |
| PALCE16V8Q-15JC/4 | PALCE16V8L-15JC |
| PALCE16V8Q-15PC/4 | PALCE16V8L-15PC |
| PALCE16V8Q-25JC/4 | PALCE16V8L-25JC |
| PALCE16V8Q-25PC/4 | PALCE16V8L-25PC |
| PAL16L8A-4C | PALC16L8L-35C |
| PAL16L8A-4M | PALC16L8-40M |
| PAL16R4A-4C | PALC16R4L-35C |
| PAL16R4A-4M | PALC16R4-40M |
| PAL16R6A-4C | PALC16R6L-35C |
| PAL16R6A-4M | PALC16R6-40M |
| PAL16R8A-4C | PALC16R8L-35 |
| PAL16R8A-4M | PALC16R8-40M |
| PAL22V10-7JC | PALC22V10D-7JC |
| PAL22V10-7PC | PALCE22V10-7PC |
| PALCE22V10H-7JC | PALCE22V10-10JC |
| PAL22V10-10DC | PALCE22V10-10DC |
| PAL22V10-10JC | PALCE22V10-7JC |
| PAL22V10-10PC | PALCE22V10-10PC |
| PALCE22V10H-10PC | PALCE22V10-10PC |
| PALCE22V10H-10JC | PALCE22V10-10JC |
| PALCE22V10H-10PC | PALCE22V10-10PC |
| PAL22V10-15DC | PALC22V10B–15DC |
| PAL22V10-15JC | PALC22V10B-15JC |
| PAL22V10-15PC | PALC22V10B-15PC |
| PALCE22V10H-15JC | PALCE22V10-15JC |
| PALCE22V10H-15PC | PALCE22V10-15PC |
| 27C128-45C | 27C128-45C |
| 27H010-45 | 27H010-45 |
| 27H256-45C | 27C256A-45C |
| 27H256-45M | 27C256A-45M |

| AMD | CYPRESS |
|---|---|
| PALCE22V10H-25JC | PALCE22V10-25JC |
| PALCE22V10H-25PC | PALCE22V10-25PC |
| 27C64-55C | 7C266-55C |
| 27H256-35C | 27H256A-35C |
| 27C128-55C | 27C128-55C |
| 27C256-55C | 27C256–55C |
| 27H010-55 | 27H010-55 |
| 27H256-55C | 27C256A-55C |
| 27H256-55M | 27C256A-55M |
| 27C64-70C | 27C64-70C |
| 27C128-70C | 27C128-70C |
| 27C256-70C | 27C256a-70C |
| 27H010-70 | 27C010-70 |
| 27H256-70C | 27C256A-70C |
| 27C512-75C | 27C512-70C |
| 27C64-90C | 27C64-90C |
| 27C010-90C | 27C010-90C |
| 27C128-90C | 27C128-90C+ |
| 27C256-90C | 27C256A-90C |
| 27C512-90C | 27C512-90C |
| 27H010-90 | 27C010-90 |
| 27C64-120C | 27C64-120C |
| 27C010-120C | 27C010-120C |
| 27C128-120C | 27C128-120C+ |
| 27C256-120C | 27C256-120C+ |
| 27C512-120C | 27C512-120C |
| 27C64-150C | 27C64-150C |
| 27C010-150C | 27C010-150C |
| 27C128-150C | 27C128-150C+ |
| 27C256-150C | 27C256-150C+ |
| 27C512-150C | 27C512-150C |
| 27C64-200C | 27C64-200C |
| 27C010-200C | 27C010-200C |
| 27C128-200C | 27C128-200C+ |
| 27C256-200C | 27C256A-200C+ |
| 27C512-200C | 27C512-200C |
| PAL16L8AC | PALC16L8-25C |
| PAL16L8ALC | PALC16L8-25C |
| PAL16L8ALM | PALC16L8-30M |
| PAL16L8AM | PALC16L8-30M |
| PAL16L8BM | PALC16L8-20M |

# Product Line Cross Reference

| AMD | CYPRESS |
|---|---|
| PAL16L8C | PALC16L8-35C |
| PAL16L8LC | PALC16L8-35C |
| PAL16L8LM | PALC16L8-40M |
| PAL16L8M | PALC16L8-40M |
| PAL16L8QC | PALC16L8L-35C |
| PAL16L8QM | PALC16L8-40M |
| PAL16R4ALC | PALC16R4-25C |
| PAL16R4ALM | PALC16R4-30M |
| PAL16R4AM | PALC16R4-30M |
| PAL16R4BM | PALC16R4-20M |
| PAL16R4C | PALC16R4-35C |
| PAL16R4LC | PALC16R4-35C |
| PAL16R4LM | PALC16R4-40M |
| PAL16R4M | PALC16R4-40M |
| PAL16R4QC | PALC16R4L-35C |
| PAL16R4QM | PALC16R4-40M |
| PAL16R6AC | PALC16R6-25C |
| PAL16R6ALC | PALC16R6-25C |
| PAL16R6ALM | PALC16R6-30M |
| PAL16R6AM | PALC16R6-30M |
| PAL16R6BM | PALC16R6-20M |
| PAL16R6C | PALC16R6-35C |
| PAL16R6LC | PALC16R6-35C |
| PAL16R6LM | PALC16R6-40M |
| PAL16R6M | PALC16R6-40M |
| PAL16R6QC | PALC16R6L-35C |
| PAL16R6QM | PALC16R6-40M |
| PAL16R8AC | PALC16R8-25C |
| PAL16R8ALC | PALC16R8-25C |
| PAL16R8ALM | PALC16R8-30M |
| PAL16R8AM | PALC16R8-30M |
| PAL16R8BM | PALC16R8-20M |
| PAL16R8C | PALC16R8-35C |
| PAL16R8LC | PALC16R8-35C |
| PAL16R8LM | PALC16R8-40M |
| PAL16R8M | PALC16R8-40M |
| PAL16R8QC | PALC16R8L-35 |
| PAL16R8QM | PALC16R8-40M |
| PAL22V10-12/B3A | PALC22V10B-10LMB |
| PAL22V10-12/BLA | PALC22V10B-10DMB |

| AMD | CYPRESS |
|---|---|
| PAL22V10-20/B3A | PALC22V10B-20LM |
| PAL22V10-20/BLA | PALC22V10B-20DM |
| PAL22V10/B3A | PALC22V10-35LMB |
| PAL22V10/BLA | PALC22V10-35DMB |
| PAL22V10A/B3A | PALC22V10-25LMB |
| PAL22V10A/BLA | PALC22V10-25DMB |
| PAL22V10ADC | PALC22V10-25DC |
| PAL22V10AJC | PALC22V10-25JC |
| PAL22V10APC | PALC22V10-25PC |
| PAL22V10DC | PALC22V10-35DC |
| PAL22V10JC | PALC22V10-35JC |
| PAL22V10PC | PALC22V10-35PC |
| PALCE22V10H-15/B3A | PALCE22V0-15LMB |
| PALCE22V10H-15/BLA | PALCE22V10-15DMB |
| PALCE22V10H-20/B3A | PALCE22V10-20LMB |
| PALCE22V10H-20/BLA | PALCE22V10-20DMB |
| PALCE22V10H-25/B3A | PALCE22V10-25LMB |
| PALCE22V10H-25/BLA | PALCE22V10-25DMB |
| PALCE22V10H-30/B3A | PALCE22V10-25LMB |
| PALCE22V10H-30/BLA | PALCE22V10-25DMB |

| ATMEL | CYPRESS |
|---|---|
| PREFIX: AT | PREFIX: CY |
| SUFFIX: D | SUFFIX: W |
| SUFFIX: K | SUFFIX: H |
| SUFFIX: L | SUFFIX: Q |
| SUFFIX: J | SUFFIX: J |
| SUFFIX: P | SUFFIX: P |
| SUFFIX: T | SUFFIX: Z |
| 22V10 | PALC22V10 |
| 22V10-15 | PALC22V10B |
| 27HC256R-35C | 27H256-35C |
| 27HC641-35C | 7C264-35C |
| 27HC642-35C | 7C261-35C |

| ATMEL | CYPRESS |
|---|---|
| 27C010-45C | 27H010-45C |
| 27HC256R-45C | 27C256A-45C |
| 27HC641-45C | 7C264-45C |
| 27HC641-45M | 7C264-45M |
| 27HC642-45C | 7C261-45C |
| 27HC642-45M | 7C261-45M |
| 27C010-55C | 27H010-55C |
| 27HC256R-55C | 27C256A-55C |
| 27HC641-55C | 7C264-55C |
| 27HC641-55M | 7C264-55M |
| 27HC642-55C | 7C261-55C |
| 27HC642-55M | 7C261-55M |
| 27C010-70C | 27C010-70C |
| 27C512-70C | 27C512-70C |
| 27C256R-70C | 27C256A-70C |
| 27HC256R-70C | 27C256A-70C |
| 27HC256R-70M | 27C256A-70M |
| 27HC641-70C | 7C264-70C |
| 27HC642-70C | 7C261-55C |
| 27C010-90C | 27C010-90C |
| 27C512-90C | 27C512-90C |
| 27C256R-90C | 27C256A-90C+ |
| 27C010-120C | 27C010-120C |
| 27C512-120C | 27C512-120C |
| 27C256R-120C | 27C256A-120C+ |
| 27C010-150C | 27C010-150C |
| 27C512-150C | 27C512-150C |
| 27C256R-150C | 27C256A-150C+ |
| 27C010-200C | 27C010-200C |
| 27C512-200C | 27C512-200C |
| 27C256R-200C | 27C256A-200C+ |

| AUSTIN SEMI | CYPRESS |
|---|---|
| PREFIX: MT | PREFIX: CY |
| 5C6401-20M | 7C187A-20MB |
| 5C6404-20M | 7C164A-20MB |
| 5C6408-20M | 7C185A-20MB |
| 5C1608-25M | 7C128A-25M |
| 5C2561-25M | 7C197-25MB |
| 5C2564-25M | 7C194-25MB |
| 5C2568-25M | 7C199-25MB |

| AUSTIN SEMI | CYPRESS |
|---|---|
| 5C2568CW-25M | 7C198-25MB |
| 5C2568W-25M | 7C198-25MB |
| 5C6401-25M | 7C187A-25MB |
| 5C6404-25M | 7C164A-25MB |
| 5C6408-25M | 7C185A-25M |
| 5C1608-30M | 7C128A-25M |
| 5C6401-30M | 7C187A-25MB |
| 5C6404-30M | 7C164A-25MB |
| 5C6408-30M | 7C185A-25MB |
| 5C1608-35M | 7C128A-35M |
| 5C2561-35M | 7C197-35MB |
| 5C2564-35M | 7C194-35MB |
| 5C2568-35M | 7C199-35MB |
| 5C2568CW-35M | 7C198-35MB |
| 5C2568W-35M | 7C198-35MB |
| 5C6401-35M | 7C187A-35MB |
| 5C6404-35M | 7C164A-35MB |
| 5C6408-35M | 7C185A-35MB |
| 5C2561-45M | 7C197-45MB |
| 5C2564-45M | 7C194-45MB |
| 5C2568-45B | 7C199-45MB |
| 5C2568CW-45B | 7C198-45MB |
| 5C2568W-45B | 7C198-45MB |

| CATALYST | CYPRESS |
|---|---|
| PREFIX: CAT | PREFIX: CY |
| 27HC256-55L | 27C256-55C+ |
| 27HC256L/LI-55 | 27C256-55C/I |
| 27HC256-70L | 27C256-70C+ |
| 27HC256L/LI-70 | 27C256-70C/I |
| 27HC256-90L | 27C256-70C+ |
| 27HC256-120L | 27C256-120C+ |

| DALLAS | CYPRESS |
|---|---|
| PREFIX: DS | PREFIX: CY |
| 2009 | 7C421-25C |
| 2010 | 7C425-25C |
| 2011 | 7C429-25C |

| DENSEPAK | CYPRESS |
|---|---|
| PREFIX: DPS | PREFIX: CYM |

| DENSEPAK | CYPRESS |
|---|---|
| 6432-45C | 1830HD-45C |
| 6432-55C | 1830HD-55C |

| EDI | CYPRESS |
|---|---|
| PREFIX: ED | PREFIX: CYM |
| 8464C-45 | 7C194-45 |
| 8F32256C | 1841PZ |
| 8F3264C | 1831PZ |
| 8F8512CXXBC | 1465PC-XXC |
| 8F8512LPXXB6C | 1465LPD-XXC |
| 8F8512PXXB6C | 1465LPD-XXC |
| 8M3264CXXC6B | M1830HD-XXMB |
| 8M3264CXXC6C | M1830HD-XXC |
| 8M32256CXXC6B | M1840HD-XXMB |
| 8M32256CXXC6B | M1840HD-XXC |
| 8M8512CXXM6C | 1464PD-XXC |

| FUJITSU | CYPRESS |
|---|---|
| PREFIX: MB | PREFIX: CY |
| PREFIX: MBM | PREFIX: CY |
| SUFFIX: F | SUFFIX: F |
| SUFFIX: M | SUFFIX: P |
| SUFFIX: Z | SUFFIX: D |
| 8464L-70 | 7C185-45C+ |
| 8464L-100 | 7C185-55C+ |
| 8299 | 7C188 |
| 8287-35 | 7C199-35 |
| 8287-45 | 7C199-45 |
| 8128-10 | 7C128A-55C |
| 8171-55 | 7C187-45C |
| 8128-15 | 7C128A-55C |
| 8168-55 | 7C168A-45C |
| 8167A-55 | 7C167A-45C |
| 8171-70 | 7C187-45C |
| 8167-70W | 7C167A-45M |
| 8167A-70 | 7C167A-45C |
| 7238RA-20 | 7C245A-18C |
| 7238RA-25 | 7C245A-25C |
| 7232RA-25 | 7C235A-25C |
| 7226RA/S-25 | 7C225A-25C |
| 7144E | 7C264-55C |

| FUJITSU | CYPRESS |
|---|---|
| 7144E-W | 7C264-55M |
| 7144H | 7C264-55C |
| 7138E-W | 7C291/2A-50M |
| 7132E | 7C282A-45C |
| 7132E-SK | 7C281A-45C |
| 7132E-W | 7C282A-45M |
| 7132H | 7C282A-45C |
| 7132H-SK | 7C281A-45C |
| 7132Y | 7C282A-30C |
| 7132Y-SK | 7C281A-30C |
| 7138Y-35 | 7C291/2A-35C |
| 7138H-45 | 7C291/2A-35C |
| 7138E-55 | 7C291/2A-50C |
| 7132L-70 | 7C281/2A-45C |
| 2147H-35 | 2147-35C |
| 2149-45 | 2149-45C |
| 2147H-45 | 2147-45C |
| 81C74-25 | 7C164-25C |
| 81C75-25 | 7C166-25C |
| 71A38-25 | 7C291/2A-25C |
| 81C67-35 | 7C167A-35C |
| 81C74-35 | 7C164-35C+ |
| 81C75-35 | 7C166-35C |
| 81C81A-35 | 7C197-35 |
| 81C84A-35 | 7C194-35 |
| 71A38-35 | 7C291/2A-35C |
| 71C44-35 | 7C264-35C |
| 81C67-45 | 7C167A-45C |
| 81C68-45 | 7C168A-45C |
| 81C71-45 | 7C187-45C |
| 81C74-45 | 7C164-45C |
| 81C78-45 | 7C186-45C |
| 81C81A-45 | 7C197-45 |
| 81C84A-45 | 7C194-45 |
| 71C44-45 | 7C264-45C |
| 71C46-45 | 7C254-45C |
| 81C67-55W | 7C167A-45M |
| 81C68-55W | 7C168A-45M+ |
| 81C71-55 | 7C187-45C |
| 81C78-55 | 7C186-55C |
| 81C86-70 | 7C192-45C+ |
| 27C256A-150C | 27C256-150C+ |

| FUJITSU | CYPRESS |
|---|---|
| 27C128-170C | 27C128-150C+ |
| 27C256A-170C | 27C256-150C+ |
| 27C128-200C | 27C128-200C+ |
| 27C256A-200C | 27C256-200C+ |
| 27C128-250C | 27C128-200C+ |
| 7238RA-20-W | 7C245A-18M |
| 7238RA-25-W | 7C245A-25M |

| HARRIS | CYPRESS |
|---|---|
| PREFIX: HM | PREFIX: CY |
| PREFIX: HPL | PREFIX: CY |
| SUFFIX: 8 | SUFFIX: B |
| PREFIX: 1 | SUFFIX: D |
| PREFIX: 9 | SUFFIX: F |
| PREFIX: 4 | SUFFIX: L |
| PREFIX: 3 | SUFFIX: P |
| 16LC8-9 | PALC16L8-40M |
| 16RC8-9 | PALC16R8-40M |
| 16LC8-8 | PALC16L8-40M |
| 16RC8-8 | PALC16R8-40M |
| 16LC8-5 | PALC16L8L-35C |
| 16RC8-5 | PALC16R8L-35C |
| 16RC6-9 | PALC16R6-40M |
| 16RC6-8 | PALC16R6-40M |
| 16RC6-5 | PALC16R6L-35C |
| 16RC4-9 | PALC16R4-40M |
| 16RC4-8 | PALC16R4-40M |
| 16RC4-5 | PALC16R4L-35C |
| 6-7681-5 | 7C281A-45C |
| 6-7681A-5 | 7C281A-45C |
| 6-76161-2 | 7C291A-50M |
| 6-76161-5 | 7C291A-50C |
| 6-76161A-2 | 7C291A-50M |
| 6-76161A-5 | 7C291A-50C |
| 6-76161B-5 | 7C291A-35C |
| 76641-2 | 7C264-55M |
| 76641-5 | 7C264-55C |
| 76641A-5 | 7C264-45C |
| 7681-2 | 7C282A-45M |
| 7681-5 | 7C282A-45C |
| 7681A-5 | 7C282A-45C |

| HARRIS | CYPRESS |
|---|---|
| 76161-2 | 7C292A-50M |
| 76161A-2 | 7C292A-50M |
| 76161A-5 | 7C292A-50C |
| 76161B-5 | 7C292A-35C |

| HITACHI | CYPRESS |
|---|---|
| PREFIX: HM | PREFIX: CY |
| PREFIX: HN | PREFIX: CY |
| PREFIX: HN48 | PREFIX: CY |
| SUFFIX: CG | SUFFIX: L |
| SUFFIX: G | SUFFIX: D |
| SUFFIX: P | SUFFIX: P |
| 25089 | 7C282-45C |
| 25089S | 7C282-45C |
| 25169S | 7C292-50C |
| 27128G-25C | 27C128-200C+ |
| 27256G-15 | 27C256A-150C |
| 27256G-17 | 27C256A-150C |
| 27256G-20 | 27C256A-200C |
| 4847 | 2147-55C |
| 4847-2 | 2147-45C |
| 4847-3 | 2147-55C |
| 6116ALS-12 | 6116A-55C* |
| 6116AS-12 | 6116A-55C+ |
| 6147 | 7C147-45C* |
| 6147-3 | 7C147-45C* |
| 6147H-35 | 7C147-35C+ |
| 6147H-45 | 7C147-45C+ |
| 6147H-55 | 7C147-45C+ |
| 6147HL-35 | 7C147-35C* |
| 6147HL-45 | 7C147-45C* |
| 6148 | 7C148-45C |
| 6148H-35 | 21L48-35C |
| 6148H-45 | 7C148-45C+ |
| 6148H-55 | 7C14845C+ |
| 6148HL-35 | 21L48-35C* |
| 6148HL-45 | 7C148-45C* |
| 6148L | 7C148-45C* |
| 6167-6 | 7C167A-45C+ |
| 6167-8 | 7C167A-45C+ |
| 6167H-55 | 7C167A-45C |

| HITACHI | CYPRESS |
|---|---|
| 6167H-70 | 7C167A-45C |
| 6167L-6 | 7C167A-45C* |
| 6167L-8 | 7C167A-45C* |
| 6168H-45 | 7C168A-45C+ |
| 6168HL-45 | 7C168A-45C* |
| 6207P-35 | 7C197-35 |
| 6207P-45 | 7C197-45 |
| 6208P-35 | 7C194-35 |
| 6208P-45 | 7C194-45 |
| 62256 | 7C198* |
| 62256BLTM-xx | CY62256-70ZC |
| 62256BLRM-xx | CY62256-70RZC |
| 62256BLFP-xx | CY62256-70SC |
| 62256BLFP-xx | CY62256-70SNC |
| 62256BLP-xx | CY62256-70PC |
| 62256BLTM-xx | CY62256L-70ZC |
| 62256BLRM-xx | CY62256L-70RZC |
| 62256BLFP-xx | CY62256L-70SNC |
| 62256BLTM-xxSL | CY62256L-70ZC |
| 62256BLRM-xxSL | CY62256LL-70RC |
| 62256BLFP-xxSL | CY62256LL-70SNC |
| 6264AFP-XX | 6264-70SC |
| 6264AFP-XX | 6264-70SNC |
| 6267-35 | 7C167A-35C+ |
| 6267-45 | 7C167A-45C |
| 6268-25 | 7C168A-25C |
| 6268-35 | 7C168A-35C |
| 628128ALFP-xx | 62128-70SC |
| 628128ALT-xx | 62128-70ZC |
| 628128ALFP-xxL | 62128-70RZC |
| 628128ALT-xxL | 62128L-70SC |
| 628128ALT-xxL | 62128L-70ZC |
| 628128ALP-xxSL | 62128LL-70SC |
| 628128LT-xxSL | 62128LL-70ZC |
| 62832H | 7C199+ |
| 62832 | 7C199 |
| 6288-35 | 7C164-35C |
| 62932 | 7C188 |
| 62W256LT-xx | CY62256-70ZC |
| 62W256LFP-xxT | CY62256V-70SC |
| 62W256LFP-xxT | CY62256V-70SNC |
| 62W256LT-xx | CY62256VL-70ZC |

| HITACHI | CYPRESS |
|---|---|
| 62W256LFP-xxT | CY62256VL-70SC |
| 62W256LFP-xxT | CY62256VL-70SNC |
| 62W256LT-xxSL | CY62256VLL-70ZC |
| 62W256LT-xxSLT | CY62256VLL-70SNC |
| 6707-20 | 7C197-20C |
| 6707-25 | 7C197-25C |
| 6707A-15 | 7C197-15C |
| 6707A-20 | 7C197-20C |
| 6707A-25 | 7C197-25C |
| 6708-20 | 7C194-20C |
| 6708-25 | 7C194-25C |
| 6708A-15 | 7C194-15C |
| 6708A-20 | 7C194-20C |
| 6708A-25 | 7C194-25C |
| 6709-20 | 7C195-20C |
| 6709-25 | 7C195-25C |
| 6709A-15 | 7C195-15C |
| 6709A-20 | 7C195-20C |
| 6709A-25 | 7C195-25C |
| 6716-25 | 7C128A-25C |
| 6716-30 | 7C128A-25C |
| 6787-30 | 7C187-25C |
| 6788-25 | 7C164-25C |
| 6788-30 | 7C164-25C |

| HYUNDAI | CYPRESS |
|---|---|
| PREFIX: HY | PREFIX: CY |
| 62256AT1-xx | CY62256-70ZC |
| 62256AR1-xx | CY62256-70RZC |
| 62256AJ-xx | CY62256-70SC |
| 62256AJ-xx | CY62256-70SNC |
| 62256AP-xx | CY62256-70PC |
| 62256ALT1-xx | CY62256L-70ZC |
| 62256ALR1-xx | CY62256L-70RZC |
| 62256ALJ-xx | CY62256L-70SNC |
| 62256ALLT1-xx | CY62256L-70ZC |
| 62256ALLR1-xx | CY62256LL-70RC |
| 62256ALLJ-xx | CY62256LL-70SNC |
| 62V256AT1-xx | CY62256-70ZC |
| 62V256AR1-xx | CY62256-70RZC |
| 62V256AJ-xx | CY62256V-70SC |

| HYUNDAI | CYPRESS |
|---|---|
| 62V256AJ-xx | CY62256V-70SNC |
| 62V256ALT1-xx | CY62256VL-70ZC |
| 62V256ALR1-xx | CY62256VL-70RZC |
| 62V256ALJ-xx | CY62256VL-70SC |
| 62V256ALJ-xx | CY62256VL-70SNC |
| 62V256ALLT1-xx | CY62256VLL-70ZC |
| 62V256ALLR1-xx | CY62256VLL-70RZC |
| 62V256ALLJ-xx | CY62256VLL-70SNC |
| 628100AG-xx | 62128-70SC |
| 628100AT1-xx | 62128-70ZC |
| 628100AR1-xx | 62128-70RZC |
| 628100ALG-xx | 62128L-70SC |
| 628100ALT1-xx | 62128L-70ZC |
| 628100ALLG-xx | 62128LL-70SC |
| 628100ALLT1-xx | 62128LL-70ZC |
| 62V8100ALG-xx | 62128VL-70SC |
| 62V8100ALT1-xx | 62128VL-70ZC |
| 62V8100ALG-xx | 62128VLL-70SC |
| 62V8100ALT1-xx | 62128VLL-70ZC |
| 6264AJ-xx | 6264-70SC |
| 6264AJ-xx | 6264-70SNC |

| ICS | CYPRESS |
|---|---|
| ICS9159C-02 | CY2254SC-1 |
| AV9159-01 | CY2255SC-1 |
| ICS9159-05 | CY2260 |

| ICT | CYPRESS |
|---|---|
| 27CX256-35C | CY27H256-35C |
| 27CX256-45C | CY27C256-45C |
| 27CX256-55C | CY27C256-55C |

| ICW | CYPRESS |
|---|---|
| W84C60-202 | CY2254SC-1 |
| W84C60-402 | CY2254SC-1 |
| W48C60-203 | CY2255SC-1 |
| W84C60-404 | CY2260 |

| IDT | CYPRESS |
|---|---|
| PREFIX: IDT | PREFIX: CY |
| PREFIX: IDT | PREFIX: CYM |
| SUFFIX: B | SUFFIX: B |
| SUFFIX: D | SUFFIX: D |
| SUFFIX: F | SUFFIX: F |
| SUFFIX: L | SUFFIX: L |
| SUFFIX: P | SUFFIX: P |
| SUFFIX: PF | SUFFIX: A |
| 39C01CB | 7C901-32M+ |
| 39C01CC | 2901CC+ |
| 39C01CM | 2901CM+ |
| 39C01DB | 7C901-27M+ |
| 39C01DC | 7C901-23C+ |
| 39C09A | 7C909-40C+ |
| 39C09AB | 7C909-40M+ |
| 39C10B | 7C910-50C- |
| 39C10BB | 7C910-51M |
| 39C11A | 7C911-40C+ |
| 39C11AB | 7C911-40M+ |
| 6116SA25 | 7C128A-25C |
| 6116SA35 | 7C128A-35C |
| 6116SA35 | 6116A-35C |
| 6116SA35B | 7C128A-35MB |
| 6116SA35B | 6116A-35MB |
| 6116SA45 | 7C128A-45C |
| 6116SA45 | 6116A-45C |
| 6116SA45B | 7C128A-45MB |
| 6116SA45B | 6116A-45MB |
| 6116SA55B | 7C128A-55MB |
| 6116SA55B | 6116A-55MB |
| 61298SA15 | 7C196-15 |
| 61298SA25 | 7C196-25C |
| 61298SA25B | 7C196-25MB |
| 61298SA35 | 7C196-35C |
| 61298SA35B | 7C196-35MB |
| 61298SA45 | 7C196-45C |
| 61298SA45B | 7C196-45MB |
| 6167SA15 | 7C167A-15C |
| 6167SA20 | 7C167A-20C |
| 6167SA20B | 7C167A-20B |
| 6167SA25 | 7C167A-25C |

| IDT | CYPRESS |
|---|---|
| 6167SA25B | 7C167A-25M |
| 6167SA35 | 7C167A-35C |
| 6167SA35B | 7C167A-35MB |
| 6167SA45B | 7C167A-45MB |
| 6168SA15 | 7C168A-15C |
| 6168SA20 | 7C168A-20C |
| 6168SA20B | 7C168A-20B |
| 6168SA25 | 7C168A-25C |
| 6168SA25B | 7C168A-25MB |
| 6168SA35 | 7C168A-35C |
| 6168SA35B | 7C168A-35MB |
| 6168SA45B | 7C168A-45MB |
| 6197SA15 | 7C170A-15C |
| 6197SA15 | 7C170A-20C |
| 6197SA25 | 7C170A-25C |
| 6197SA35 | 7C170A-35C |
| 6197SA35B | 7C170A-35MB |
| 6197SA45B | 7C170A-45MB |
| 6197SA55 | 7C170A-45C |
| 6197SA55B | 7C170A-45MB |
| 6198SA15 | 7C166-15C |
| 6198SA20 | 7C166-20C |
| 6198SA20B | 7C166-A20MB |
| 6198SA25 | 7C166-25C |
| 6198SA25B | 7C166-A25MB |
| 6198SA30B | 7C166A-25MB |
| 61B298S12 | 7C195-12C |
| 61B298S15 | 7C195-15C |
| 61B298S20 | 7C195-20C |
| 61B298S15B | 7C195-15MB |
| 61B298S20B | 7C195-20MB |
| 7005S35 | 7C144-25C |
| 7005S35 | 7C144-35C |
| 7005S45B | 7C144-35MB |
| 7006S15 | 7C006-15C |
| 7006S17 | 7C006-15C |
| 7006S20 | 7C006-15C |
| 7006S25 | 7006S25C |
| 7006S35 | 7006S35C |
| 7006S55 | 7C006-55C |
| 7006S70 | 7C024-15C |
| 7015S25 | 7C145-25C |

| IDT | CYPRESS |
|---|---|
| 7015S35 | 7C145-35C |
| 7016S25 | 7C016-25C |
| 7016S35 | 7C016-35C |
| 7024S15 | 7C024-15C |
| 7024S17 | 7C024-15C |
| 7024S20 | 7C024-15C |
| 7024S25 | 7C024-25C |
| 7024S35 | 7C024-35C |
| 7024555 | 7C024-55C |
| 7024555 | 7C025-55C |
| 7025S15 | 7C025-15C |
| 7025217 | 7C025-15C |
| 7025S20 | 7C025-15C |
| 7025S25 | 7C025-25C |
| 7025S35 | 7C025-35C |
| 7133S25 | 7C133-25C |
| 7133S35 | 7C133-35C |
| 7143S25 | 7C143-25C |
| 7143S35 | 7C143-35C |
| 71V256-15 | 7C1399-15C |
| 71V256-20 | 7C1399-20C |
| 71V256-25 | 7C1399-25C |
| 71024-15 | 7C109A-15C |
| 71024-20 | 7C109A-20C |
| 71024-20 | 7C109-20C |
| 71024-25 | 7C109-25C |
| 71028-15 | 7C106A-15C |
| 71028-20 | 7C106A-20C |
| 71028-25 | 7C106A-25C |
| 71256SA15 | 7C199-15C |
| 71256SA20 | 7C199-20C |
| 71256SA20B | 7C199-20MB |
| 71256SA25 | 7C198-25C |
| 71256SA30 | 7C198-25C |
| 71256SA30B | 7C198-25MB |
| 71256SA35 | 7C198-35C |
| 71256SA35B | 7C198-35MB |
| 71256SA45 | 7C198-45C |
| 71256SA45B | 7C198-45MB |
| 71257SA25 | 7C197-25C |
| 71257SA25B | 7C197-25MB |
| 71257SA35 | 7C197-35C |

| IDT | CYPRESS |
|---|---|
| 71257SA35B | 7C197-35MB |
| 71257SA45 | 7C197-45C |
| 71257SA45B | 7C197-45MB |
| 71257SA55 | 7C197-45C |
| 71258SA25 | 7C194-25C |
| 71258SA25B | 7C194-25MB |
| 71258SA35 | 7C194-35C |
| 71258SA35B | 7C194-35MB |
| 71258SA45 | 7C194-45C |
| 71258SA45B | 7C194-45MB |
| 71281SA25 | 7C191-25C |
| 71281SA25B | 7C191-25MB |
| 71281SA35 | 7C191-35C |
| 71281SA35B | 7C191-35MB |
| 71281SA45 | 7C191-45C |
| 71281SA45B | 7C191-45MB |
| 71282SA25 | 7C192-25C |
| 71282SA25B | 7C192-25MB |
| 71282SA35 | 7C192-35C |
| 71282SA35B | 7C192-35MB |
| 71282SA45 | 7C192-45C |
| 7130LA25 | 7C130-25C |
| 7130LA25J | 7C131-25JC |
| 7130LA30 | 7C130-30C |
| 7130LA30J | 7C131-30JC |
| 7130LA35 | 7C130-35C |
| 7130LA35B | 7C130-35MB |
| 7130LA35J | 7C131-35JC |
| 7130LA35LB | 7C130-35LMB |
| 7130LA45 | 7C130-45C |
| 7130LA45B | 7C131-45MB |
| 7130LA45J | 7C131-45JC |
| 7130LA45LB | 7C130-45LMB |
| 7130LA55 | 7C130-55C |
| 7130LA55B | 7C131-55MB |
| 7130LA55J | 7C131-55JC |
| 7130LA55L52B | 7C130-55LMB |
| 7130LA70 | 7C130-55C |
| 7130LA70B | 7C131-55MB |
| 7130LA70J | 7C131-55JC |
| 7130LA70LB | 7C130-55LMB |
| 7130LA90LB | 7C131-55LMB |

| IDT | CYPRESS |
|---|---|
| 7130SA25 | 7C130-25C |
| 7130SA25J | 7C131-25JC |
| 7130SA30 | 7C130-25C |
| 7130SA30J | 7C131-30JC |
| 7130SA35 | 7C130-35C |
| 7130SA35B | 7C130-35MB |
| 7130SA35J | 7C131-35JC |
| 7130SA35LB | 7C131-35LMB |
| 7130SA45 | 7C130-45C |
| 7130SA45B | 7C130-45MB |
| 7130SA45J | 7C131-45JC |
| 7130SA45LB | 7C131-45LMB |
| 7130SA55 | 7C130-55C |
| 7130SA55B | 7C130-55M |
| 7130SA55J | 7C131-55JC |
| 7130SA55LB | 7C131-55LMB |
| 7130SA70 | 7C130-55C |
| 7130SA70B | 7C130-55MB |
| 7130SA70J | 7C131-55JC |
| 7130SA70LB | 7C131-55LMB |
| 7130SA90 | 7C130-55C |
| 7130SA90B | 7C130-55MB |
| 7130SA90J | 7C131-55JC |
| 7130SA90LB | 7C131-55LMB |
| 7130SA100 | 7C130-55C |
| 7130SA100B | 7C130-55MB |
| 7130SA100LB | 7C131-55LMB |
| 71321LA25 | 7C136-25C |
| 71321LA30 | 7C136-30C |
| 71321LA35 | 7C136-35C |
| 71321LA35B | 7C136-35MB |
| 71321LA45 | 7C136-45C |
| 71321LA45B | 7C136-45MB |
| 71321LA55 | 7C136-55C |
| 71321LA55B | 7C136-55MB |
| 71321LA70 | 7C136-55C |
| 71321LA70B | 7C136-55MB |
| 71321LA90 | 7C136-55C |
| 71321LA90B | 7C136-55MB |
| 71321SA25 | 7C136-25C |
| 71321SA30 | 7C136-30C |
| 71321SA35 | 7C136-35C |

| IDT | CYPRESS |
|---|---|
| 71321SA35B | 7C136-35MB |
| 71321SA45 | 7C136-45C |
| 71321SA45B | 7C136-45MB |
| 71321SA55 | 7C136-55C |
| 71321SA55B | 7C136-55MB |
| 71321SA70 | 7C136-55C |
| 71321SA70B | 7C136-55MB |
| 71321SA90 | 7C136-55C |
| 71321SA90B | 7C136-55MB |
| 713256-20 | 7C1399-20C |
| 713256-25 | 7C1399-25C |
| 7132LA25 | 7C132-25C |
| 7132LA30 | 7C132-30C |
| 7132LA35 | 7C132-35C |
| 7132LA35B | 7C132-35MB |
| 7132LA45 | 7C132-45C |
| 7132LA45B | 7C132-45MB |
| 7132LA55 | 7C132-55C* |
| 7132LA55B | 7C132-55MB |
| 7132LA70 | 7C132-55C* |
| 7132LA70B | 7C132-55M* |
| 7132LA90 | 7C132-55C* |
| 7132LA90B | 7C132-55M* |
| 7132LA100 | 7C132-55C* |
| 7132LA100B | 7C132-55M* |
| 7132LA120B | 7C132-55M* |
| 7132SA25 | 7C132-25C |
| 7132SA30 | 7C132-30C |
| 7132SA35 | 7C132-35C |
| 7132SA35B | 7C132-35MB |
| 7132SA45 | 7C132-45C |
| 7132SA45B | 7C132-45MB |
| 7132SA55 | 7C132-55C+ |
| 7132SA55B | 7C132-55MB |
| 7132SA70 | 7C132-55C+ |
| 7132SA70B | 7C132-55M+ |
| 7132SA90 | 7C132-55C+ |
| 7132SA90B | 7C132-55M+ |
| 7132SA100 | 7C132-55C+ |
| 7132SA100B | 7C132-55M+ |
| 7132SA120B | 7C132-55M+ |
| 71342S35 | 7C1342-25C |

| IDT | CYPRESS |
|---|---|
| 71342S35 | 7C1342-35C |
| 71342S45B | 7C1342-35MB |
| 7134S35 | 7C134-25C |
| 7134S35 | 7C134-35C |
| 7134S35J52 | 7C135-25JC |
| 7134S35J52 | 7C135-35JC |
| 7134S35L52 | 7C135-25LC |
| 7134S35L52 | 7C135-35LC |
| 7134S45B | 7C134-35MB |
| 7134S45L52B | 7C135-35LMB |
| 7140LA25 | 7C140-25C |
| 7140LA25J | 7C141-25JC |
| 7140LA30 | 7C140-30C |
| 7140LA30J | 7C141-30JC |
| 7140LA30L52 | 7C141-30LC |
| 7140LA35 | 7C140-35C |
| 7140LA35B | 7C140-35MB |
| 7140LA35J | 7C141-35JC |
| 7140LA35LB | 7C141-35LMB |
| 7140LA45 | 7C140-45C |
| 7140LA45B | 7C140-45MB |
| 7140LA45J | 7C141-45JC |
| 7140LA45LB | 7C141-45LMB |
| 7140LA55 | 7C140-55C |
| 7140LA55B | 7C140-55MB |
| 7140LA55J52 | 7C141-55JC |
| 7140LA55LB | 7C141-55LMB |
| 7140LA70 | 7C140-55C |
| 7140LA70B | 7C140-55MB |
| 7140LA70J | 7C141-55JC |
| 7140LA70LB | 7C141-55LMB |
| 7140LA90J | 7C141-55JC |
| 7140LA90LB | 7C141-55LMB |
| 7140SA25 | 7C140-25C |
| 7140SA25J | 7C141-25JC |
| 7140SA30 | 7C140-30C |
| 7140SA30J | 7C141-30JC |
| 7140SA35 | 7C140-35C |
| 7140SA35B | 7C140-35MB |
| 7140SA35J | 7C141-35JC |
| 7140SA35LB | 7C141-35LMB |
| 7140SA45 | 7C140-45C |

| IDT | CYPRESS |
|---|---|
| 7140SA45B | 7C140-45MB |
| 7140SA45J | 7C141-45JC |
| 7140SA45LB | 7C141-45LMB |
| 7140SA55 | 7C140-55C |
| 7140SA55B | 7C140-55MB |
| 7140SA55J | 7C141-55JC |
| 7140SA55LB | 7C141-55LMB |
| 7140SA70 | 7C140-55C |
| 7140SA70B | 7C140-55MB |
| 7140SA70J | 7C141-55JC |
| 7140SA70LB | 7C141-55LMB |
| 7140SA90 | 7C140-55C |
| 7140SA90B | 7C140-55MB |
| 7140SA90J | 7C141-55JC |
| 7140SA90LB | 7C141-55LMB |
| 7140SA100 | 7C140-55C |
| 7140SA100B | 7C140-55MB |
| 7140SA100L | 7C141-55C |
| 7140SA100LB | 7C141-55MB |
| 71420-9 | 7C178-8.5 |
| 71420-10 | 7C178-9.5 |
| 71420-12 | 7C178-12 |
| 71421LA25 | 7C146-25C |
| 71421LA30 | 7C146-30C |
| 71421LA35 | 7C146-35C |
| 71421LA35B | 7C146-35MB |
| 71421LA45 | 7C146-45C |
| 71421LA45B | 7C146-45MB |
| 71421LA55 | 7C146-55C |
| 71421LA55B | 7C146-55MB |
| 71421LA70 | 7C146-55C |
| 71421LA70B | 7C146-55MB |
| 71421LA90 | 7C146-55C |
| 71421LA90B | 7C146-55MB |
| 71421SA25 | 7C146-25C |
| 71421SA30 | 7C146-30C |
| 71421SA35 | 7C146-35C |
| 71421SA35B | 7C146-35MB |
| 71421SA45 | 7C146-45C |
| 71421SA45B | 7C146-45MB |
| 71421SA55 | 7C146-55C |
| 71421SA55B | 7C146-55MB |

| IDT | CYPRESS |
|---|---|
| 71421SA70 | 7C146-55C |
| 71421SA70B | 7C146-55MB |
| 71421SA90 | 7C146-55C |
| 71421SA90B | 7C146-55MB |
| 7142LA25 | 7C142-25C |
| 7142LA30 | 7C142-30C |
| 7142LA35 | 7C142-35C |
| 7142LA35B | 7C142-35MB |
| 7142LA45 | 7C142-45C |
| 7142LA45B | 7C142-45MB |
| 7142LA55 | 7C142-55C |
| 7142LA55B | 7C142-55MB |
| 7142LA70 | 7C142-55C |
| 7142LA70B | 7C142-55MB |
| 7142SA25 | 7C142-25C |
| 7142SA30 | 7C142-30C |
| 7142SA35 | 7C142-35C |
| 7142SA35B | 7C142-35MB |
| 7142SA45 | 7C142-45C |
| 7142SA45B | 7C142-45MB |
| 7142SA55 | 7C142-55C |
| 7142SA55B | 7C142-55MB |
| 7142SA70 | 7C142-55C |
| 7142SA70B | 7C142-55MB |
| 71V432S7PF | CY7C1335-75AC |
| 71V432S8PF | CY7C1335-66AC |
| 71V432S10PF | CY7C1335-60AC |
| 7164SA20 | 7C185-20C |
| 7164SA20P | 7C186-20C |
| 7164SA25 | 7C185-25C |
| 7164SA25B | 7C185A-25MB |
| 7164SA25P | 7C186-25C |
| 7164SA25PB | 7C186A-25MB |
| 7164SA30 | 7C185-25C |
| 7164SA30B | 7C185A-25MB |
| 7164SA30P | 7C186-25C |
| 7164SA30PB | 7C186A-25MB |
| 7164SA35 | 7C185-35C |
| 7164SA35B | 7C185A-35MB |
| 7164SA35P | 7C186-35C |
| 7164SA35PB | 7C186A-35MB |
| 7164SA45B | 7C185A-45MB |

| IDT | CYPRESS |
|---|---|
| 7164SA45PB | 7C186A-45MB |
| 7164SA55B | 7C185A-55MB |
| 7164SA55BP | 7C185A-55MB |
| 71681SA25 | 7C171A-25C |
| 71681SA25B | 7C171A-25MB |
| 71681SA35 | 7C171A-35C |
| 71681SA35B | 7C171A-35MB |
| 71681SA45 | 7C171A-45C |
| 71681SA45B | 7C171A-45MB |
| 71681SA55B | 7C171A-45MB |
| 71681SA70B | 7C171A-45MB |
| 71681SA85B | 7C171A-45MB |
| 71682SA25 | 7C172A-25C |
| 71682SA25B | 7C172A-25MB |
| 71682SA35 | 7C172A-35C |
| 71682SA35B | 7C172A-35MB |
| 71682SA45 | 7C172A-45C |
| 71682SA45B | 7C172A-45MB |
| 71682SA100B | 7C172A-45MB |
| 7187SA15 | 7C187-15C |
| 7187SA20 | 7C187-20C |
| 7187SA25 | 7C187-25C |
| 7187SA25B | 7C187A-25MB |
| 7187SA30 | 7C187-25C |
| 7187SA30B | 7C187A-25MB |
| 7187SA35 | 7C187-35C |
| 7187SA35B | 7C187A-35MB |
| 7187SA45B | 7C187A-45MB |
| 7188SA15 | 7C164-15C |
| 7188SA20 | 7C164-20C |
| 7188SA20B | 7C164A-20MB |
| 7188SA25 | 7C164-25C |
| 7188SA25B | 7C164A-25MB |
| 7188SA30 | 7C164-25C |
| 7188SA35 | 7C164-35C |
| 7188SA35B | 7C164A-35MB |
| 71981S35 | 7C161-35C |
| 71981S35B | 7C161A-35M |
| 71981S45 | 7C161-45C |
| 71981S45B | 7C161A-45M |
| 71981S55B | 7C161A-45M |
| 71981S70B | 7C161A-45M |

| IDT | CYPRESS |
|---|---|
| 71981S85B | 7C161A-45M |
| 71982S35 | 7C162-35C |
| 71982S35B | 7C162A-35M |
| 71982S45B | 7C162A-45M |
| 7198S35 | 7C166-35C |
| 7198S35B | 7C166A-35M |
| 7198S45B | 7C166A-45M |
| 71B256A12 | 7C199-12C |
| 71B256S20 | 7C199-20C |
| 71B256S20B | 7C199-20MB |
| 71B256SA12 | 7C199-12C |
| 71B258S12 | 7C194-12C |
| 71B258S15 | 7C194-15C |
| 71B258S15B | 7C194-15MB |
| 71B258S20 | 7C194-20C |
| 71B258S20B | 7C194-20MB |
| 71B259 | 7C188 |
| 71B2595 | 7C188 |
| 7200LA15 | 7C419-15 |
| 7200LA20T | 7C419-20 |
| 7200LA25T | 7C419-25 |
| 7200LA30T | 7C419-30 |
| 7200LA35T | 7C419-30 |
| 7200LA40T | 7C419-40 |
| 7200LA50T | 7C419-50 |
| 7200LA65T | 7C419-65 |
| 7200LA80T | 7C419-65 |
| 7200SA15 | 7C419-15 |
| 7200SA20T | 7C419-20 |
| 7200SA25T | 7C419-25 |
| 7200SA30T | 7C419-30 |
| 7200SA35T | 7C419-30 |
| 7200SA40T | 7C419-40 |
| 7200SA50T | 7C419-50 |
| 7200SA65T | 7C419-65 |
| 7200SA80T | 7C419-65 |
| 7201LA15 | 7C421-15 |
| 7201LA20 | 7C420-20C |
| 7201LA20T | 7C421-20C |
| 7201LA25 | 7C420-25C |
| 7201LA25T | 7C421-25C |
| 7201LA30B | 7C420-30MB |

| IDT | CYPRESS |
|---|---|
| 7201LA30TB | 7C421-30MB |
| 7201LA35 | 7C420-30C |
| 7201LA35T | 7C421-30C |
| 7201LA40B | 7C420-40MB |
| 7201LA40TB | 7C421-40MB |
| 7201LA50 | 7C420-40C |
| 7201LA50B | 7C420-40MB |
| 7201LA50T | 7C421-40C |
| 7201LA50TB | 7C421-40MB |
| 7201LA65 | 7C420-65C |
| 7201LA65B | 7C420-65MB |
| 7201LA65T | 7C421-65C |
| 7201LA65TB | 7C421-65MB |
| 7201LA80 | 7C420-65C |
| 7201LA80B | 7C420-65MB |
| 7201LA120 | 7C420-65C |
| 7201LA120B | 7C420-65MB |
| 7201SA15 | 7C421-15 |
| 7201SA20 | 7C420-20C |
| 7201SA20T | 7C421-20C |
| 7201SA25 | 7C420-25C |
| 7201SA25T | 7C421-25C |
| 7201SA30B | 7C420-30MB |
| 7201SA30TB | 7C421-30MB |
| 7201SA35 | 7C420-30C |
| 7201SA35T | 7C421-30C |
| 7201SA40B | 7C420-40MB |
| 7201SA40TB | 7C421-40MB |
| 7201SA50 | 7C420-40C |
| 7201SA50B | 7C420-40MB |
| 7201SA50T | 7C421-40C |
| 7201SA50TB | 7C421-40MB |
| 7201SA65 | 7C420-65C |
| 7201SA65B | 7C420-65MB |
| 7201SA65T | 7C421-65C |
| 7201SA65TB | 7C421-65MB |
| 7201SA80 | 7C420-65C |
| 7201SA80B | 7C420-65MB |
| 7201SA120 | 7C420-65C |
| 7201SA120B | 7C420-65MB |
| 7202LA15 | 7C425-15 |
| 7202LA20 | 7C424-20C |

| IDT | CYPRESS |
|---|---|
| 7202LA20T | 7C425-20C |
| 7202LA25 | 7C424-25C |
| 7202LA25T | 7C425-25C |
| 7202LA30B | 7C424-30MB |
| 7202LA30TB | 7C425-30MB |
| 7202LA35 | 7C424-30C |
| 7202LA35T | 7C425-30C |
| 7202LA40B | 7C424-40MB |
| 7202LA40TB | 7C425-40MB |
| 7202LA50 | 7C424-40C |
| 7202LA50B | 7C424-40MB |
| 7202LA50T | 7C425-40C |
| 7202LA50TB | 7C425-40MB |
| 7202LA65 | 7C424-65C |
| 7202LA65B | 7C424-65MB |
| 7202LA65T | 7C425-65C |
| 7202LA65TB | 7C425-65MB |
| 7202LA80 | 7C424-65C |
| 7202LA80B | 7C424-65MB |
| 7202LA120 | 7C424-65C |
| 7202LA120B | 7C424-65MB |
| 7202SA15 | 7C425-15 |
| 7202SA20 | 7C424-20C |
| 7202SA20T | 7C425-20C |
| 7202SA25 | 7C424-25C |
| 7202SA25T | 7C425-25C |
| 7202SA30B | 7C424-30MB |
| 7202SA30TB | 7C425-30MB |
| 7202SA35 | 7C424-30C |
| 7202SA35T | 7C425-30C |
| 7202SA40B | 7C424-40MB |
| 7202SA40TB | 7C425-40MB |
| 7202SA50 | 7C424-40C |
| 7202SA50B | 7C424-40MB |
| 7202SA50T | 7C425-40C |
| 7202SA50TB | 7C425-40MB |
| 7202SA65 | 7C424-65C |
| 7202SA65B | 7C424-65MB |
| 7202SA65T | 7C425-65C |
| 7202SA65TB | 7C425-65MB |
| 7202SA80 | 7C424-65C |
| 7202SA80B | 7C424-65MB |

| IDT | CYPRESS |
|---|---|
| 7202SA120 | 7C424-65C |
| 7202SA120B | 7C424-65MB |
| 7203L20 | 7C428-20C |
| 7203L20T | 7C429-20C |
| 7203L25 | 7C428-25C |
| 7203L25B | 7C428-25MB |
| 7203L25T | 7C429-25C |
| 7203L25TB | 7C429-25MB |
| 7203L30 | 7C428-30C |
| 7203L30T | 7C429-30C |
| 7203L35B | 7C428-30MB |
| 7203L35TB | 7C429-30MB |
| 7203L40 | 7C428-40C |
| 7203L40T | 7C429-40C |
| 7203L55B | 7C428-40MB |
| 7203L55TB | 7C429-40MB |
| 7203L65 | 7C428-65C |
| 7203L65B | 7C428-65MB |
| 7203L65T | 7C429-65C |
| 7203L65TB | 7C429-65MB |
| 7203L80 | 7C428-65C |
| 7203L80B | 7C428-65MB |
| 7203L80T | 7C429-65C |
| 7203L80TB | 7C429-65MB |
| 7203S20 | 7C428-20C |
| 7203S20T | 7C429-20C |
| 7203S25 | 7C428-25C |
| 7203S25B | 7C428-25MB |
| 7203S25T | 7C429-25C |
| 7203S25TB | 7C429-25MB |
| 7203S30 | 7C428-30C |
| 7203S30T | 7C429-30C |
| 7203S35B | 7C428-30MB |
| 7203S35TB | 7C429-30MB |
| 7203S40 | 7C428-40C |
| 7203S40T | 7C429-40C |
| 7203S55B | 7C428-40MB |
| 7203S55TB | 7C429-40MB |
| 7203S65 | 7C428-65C |
| 7203S65B | 7C428-65MB |
| 7203S65T | 7C429-65C |
| 7203S65TB | 7C429-65MB |

| IDT | CYPRESS |
|---|---|
| 7203S80 | 7C428-65C |
| 7203S80B | 7C428-65MB |
| 7203S80T | 7C429-65C |
| 7203S80TB | 7C429-65MB |
| 7204S25 | 7C432-25C |
| 7204S25T | 7C433-25C |
| 7204L30 | 7C432-30C |
| 7204L30T | 7C433-30C |
| 7204L35B | 7C432-30MB |
| 7204L35TB | 7C433-30MB |
| 7204L40 | 7C432-40C |
| 7204L40T | 7C433-40C |
| 7204L55B | 7C432-40MB |
| 7204L55TB | 7C433-40MB |
| 7204L65 | 7C432-65C |
| 7204L65B | 7C432-65MB |
| 7204L65T | 7C433-65C |
| 7204L65TB | 7C433-65MB |
| 7204L80B | 7C432-65MB |
| 7204L80TB | 7C433-65MB |
| 7204S30 | 7C432-30C |
| 7204S30T | 7C433-30C |
| 7204S35B | 7C432-30MB |
| 7204S35TB | 7C433-30MB |
| 7204S40 | 7C432-40C |
| 7204S40T | 7C433-40C |
| 7204S55B | 7C432-40MB |
| 7204S55TB | 7C433-40MB |
| 7204S65 | 7C432-65C |
| 7204S65B | 7C432-65MB |
| 7204S65T | 7C433-65C |
| 7204S65TB | 7C433-65MB |
| 7204S80B | 7C432-65MB |
| 7204S80TB | 7C433-65MB |
| 7205L20 | 7C460-15C |
| 7205L25 | 7C460-25C |
| 7205L30B | 7C460-15MB |
| 7205L30B | 7C460-25MB |
| 7205L35 | 7C460-25C |
| 7205L50 | 7C460-40C |
| 7205L50B | 7C460-40MB |
| 7206-15 | 7C462-15 |

| IDT | CYPRESS |
|---|---|
| 7206-20 | 7C462-20 |
| 7206-25 | 7C462-25 |
| 7207L15 | 7C464-15 |
| 7207L20 | 7C464-15 |
| 7207L25 | 7C464-25 |
| 7207L35 | 7C464-25 |
| 7207L30 | 7C464-40 |
| 72201L15 | 7C4201-15 |
| 72201L25 | 7C4201-25 |
| 72201L35 | 7C4201-35 |
| 72205LB15 | 7C4205-15 |
| 72205LB25 | 7C4205-25 |
| 72205LB35 | 7C4205-35 |
| 72211L15 | 7C4211-15 |
| 72211L25 | 7C4211-25 |
| 72211L35 | 7C4211-35 |
| 72215LB15 | 7C4215-15 |
| 72215LB25 | 7C4215-25 |
| 72215LB35 | 7C4215-35 |
| 72221L15 | 7C4221-15 |
| 72221L25 | 7C4221-25 |
| 72221L35 | 7C4221-35 |
| 72225LB15 | 7C4225-15 |
| 72225LB25 | 7C4225-25 |
| 72225LB35 | 7C4225-35 |
| 72231L15 | 7C4231-15 |
| 72231L25 | 7C4231-25 |
| 72231L35 | 7C4231-35 |
| 72235LB15 | 7C4235-15 |
| 72235LB25 | 7C4235-25 |
| 72235LB35 | 7C4235-35 |
| 72241L15 | 7C4241-15 |
| 72241L25 | 7C4241-25 |
| 72241L35 | 7C4241-35 |
| 72245LB15 | 7C4245-15 |
| 72245LB25 | 7C4245-25 |
| 72245LB35 | 7C4245-35 |
| 72251L15J | 7C4251-15J |
| 72251L25J | 7C4251-25J |
| 72251L35J | 7C4251-35J |
| 72251L20J | 7C4251-15J |
| 72401L10 | 7C401-10C |

| IDT | CYPRESS |
|---|---|
| 72401L10B | 7C401-10MB |
| 72401L15 | 7C401-15C |
| 72401L15B | 7C401-15MB |
| 72401L25 | 7C401-25C |
| 72401L25B | 7C401-25MB |
| 72401L35 | 7C401-25C |
| 72401L35B | 7C401-25MB |
| 72401L45 | 7C401-25C |
| 72402L10 | 7C402-10C |
| 72402L10B | 7C402-10MB |
| 72402L15 | 7C402-15C |
| 72402L15B | 7C402-15MB |
| 72402L25 | 7C402-25C |
| 72402L25B | 7C402-25MB |
| 72402L35 | 7C402-25C |
| 72402L35B | 7C402-25MB |
| 72402L45 | 7C402-25C |
| 72403L10 | 7C403-10C |
| 72403L10B | 7C403-10MB |
| 72403L15 | 7C403-15C |
| 72403L15B | 7C403-15MB |
| 72403L25 | 7C403-25C |
| 72403L25B | 7C403-25MB |
| 72403L35 | 7C403-25C |
| 72403L35B | 7C403-25MB |
| 72403L45 | 7C403-25C |
| 72404L10 | 7C404-10C |
| 72404L10B | 7C404-10MB |
| 72404L15 | 7C404-15C |
| 72404L15B | 7C404-15MB |
| 72421L15 | 7C4421-15 |
| 72421L25 | 7C4421-25 |
| 72421L35 | 7C4421-35 |
| 72404L25 | 7C404-25C |
| 72404L25B | 7C404-25MB |
| 72404L35 | 7C404-25C |
| 72404L35B | 7C404-25MB |
| 72404L45 | 7C404-25C |
| 72801L15 | 7C4801-15 |
| 72801L20 | 7C4801-15 |
| 72801L25 | 7C4801-25 |
| 72801L35 | 7C4801-35 |

| IDT | CYPRESS |
|---|---|
| 72811L15 | 7C4811-15 |
| 72821L20 | 7C4811-15 |
| 72831L25 | 7C4811-25 |
| 72841L35 | 7C4811-35 |
| 7M4017S40C | 1830HD-35C |
| 7M4017S45C | 1830HD-45C |
| 7M4017S50C | 1830HD-45C |
| 7M4017S50CB | 1830HD-45MB |
| 7M4017S55C | 1830HD-55C |
| 7M4017S60C | 1830HD-55C |
| 7M4017S60CB | 1830HD-55MB |
| 7M4017S70C | 1830HD-55C |
| 7M4017S70CB | 1830HD-55MB |
| 7MP4031 | M1821PZ |
| 7MP4036Z | M1831PZ |
| 7MP4036M | M1831PM |
| 7MP4036Z | M1836PZ |
| 7MP4036M | M1836PM |
| 7MP4045Z | M1841PZ |
| 7MP4045M | M1841PM |
| 7MP4120Z | M1851PZ |
| 7MP4120M | M1851PM |
| 7MP6121S | M7450PM-33C |
| 7MP6122S | M7451PM-33C |
| 7MP6133S33 | M7427PB-20 |
| 7MP6134S33 | M7428PB-20 |
| 7MP6151S33 | M9230PB-20 |
| 7MP6152S33 | M9231PB-20 |
| 7MP6157S | M7432PB-12/15C |
| 7MP6183S | M7424PB-20C |
| 7MP6184S | M7425PB-20C |

| IMI | CYPRESS |
|---|---|
| IMISC478 | CY2254SC-1 |
| IMISC498 | CY2254SC-1 |

| INTEL | CYPRESS |
|---|---|
| PREFIX: 85C | PREFIX: CY |
| PREFIX: 85C | PREFIX: PLD |
| PREFIX: D | SUFFIX: D |
| PREFIX: L | SUFFIX: L |

| INTEL | CYPRESS |
|---|---|
| PREFIX: P | SUFFIX: P |
| SUFFIX: /B | SUFFIX: B |
| 1223-35 | 7C148-35C |
| 1223M-35 | 7C148-25M+ |
| 1400-35 | 7C167A-35C |
| 1400M-45 | 7C167A-45M |
| 1403-25 | 7C167A-25C |
| 1403-35 | 7C167A-35C+ |
| 1403LM-35 | 7C167A-35M* |
| 1403M-35 | 7C167A-35M+ |
| 1420-45 | 7C168A-35C |
| 1420M-55 | 7C168A-45M+ |
| 1423-25 | 7C168A-25C+ |
| 1423-35 | 7C168A-35C+ |
| 1423M-35 | 7C168A-35M* |
| 1433-30 | 7C128A-25C+ |
| 1433-35 | 7C128A-35C+ |
| 1433M-35 | 7C128A-35M+ |
| 22V10-10C | PALC22V10D-7C |
| 22V10-10C | PALC22V10D-10C |
| 22V10-10C | PAL22V10C-7C+ |
| 22V10-10C | PAL22V10C-10C+ |
| 22V10-15C | PALC22V10B-15C |
| 22V10-15C | PALC22V10D-15C |

| ISSI | CYPRESS |
|---|---|
| PREFIX: IS | PREFIX: CY |
| SUFFIX: CW | SUFFIX: W |
| SUFFIX: PL | SUFFIX: J |
| 27HC010-30C | 27H010-30C |
| 27HC010-45C | 27H010-45C |
| 27HC010-55C | 27H010-55C |
| 27HC010-70C | 27C010-70C |
| 61C1024 | 7C1009 |
| 61C1024-xxU | 62128-70SC |
| 61C1024-xxT | 62128-70ZC |
| 61C512 | 7C1512 |
| 61C632A-5TQ | CY7C1335-100AC |
| 61C632A-7TQ | CY7C1335-66AC |
| 61C632A-8TQ | CY7C1335-60AC |
| 62C256-45T | 62256-70ZC |

| ISSI | CYPRESS |
|---|---|
| 62C256-xxU | 62256-70SC |
| 62C256-xxU | 62256-70SNC |
| 62C256-45T | 62256L-70ZC |
| 62C256-xxU | 62256L-70SNC |
| 62C256-45T | 62256LL-70ZC |
| 62C256-xxU | 62256LL-70SNC |
| 62C512 | 7C1512 |
| 62C64-xxU | 6264-70SC |
| 62C64-xxU | 6264-70SNC |
| 62LV256-xxT | 62256V-70ZC |
| 62LV256-xxT | 62256VL-70ZC |
| 62LV256-xxT | 62256VLL-70ZC |

| LATTICE | CYPRESS |
|---|---|
| PREFIX: EE | PREFIX: CY |
| PREFIX: GAL | PREFIX: CY |
| PREFIX: ST | PREFIX: CY |
| SUFFIX: B | SUFFIX: B |
| SUFFIX: D | SUFFIX: D |
| SUFFIX: L | SUFFIX: L |
| SUFFIX: P | SUFFIX: P |
| GAL16V8A-10LJ | PALCE16V8-10JC |
| GAL16V8A-10LP | PALCE16V8-10PC |
| GAL16V8A-15LJ | PALCE16V8-15JC |
| GAL16V8A-15LP | PALCE16V8-15PC |
| GAL16V8A-15QJ | PALCE16V8L-15JC |
| GAL16V8A-15QP | PALCE16V8L-15PC |
| GAL16V8A-25LJ | PALCE16V8-25JC |
| GAL16V8A-25LP | PALCE16V8-25PC |
| GAL16V8A-25QJ | PALCE16V8L-25JC |
| GAL16V8A-25QP | PALCE16V8L-25PC |
| GAL16V8B-7LJ | PALCE16V8-7JC |
| GAL16V8B-7LP | PALCE16V8-7PC |
| GAL16V8B-10LJ | PALCE16V8-10JC |
| GAL16V8B-10LJI | PALCE16V8-10JI |
| GAL16V8B-10LP | PALCE16V8-10PC |
| GAL16V8B-10LPI | PALCE16V8-10PI |
| GAL16V8B-15LJI | PALCE16V8-15JI |
| GAL16V8B-15LPI | PALCE16V8-15PI |
| GAL16V8B-25LJI | PALCE16V8-25JI |
| GAL16V8B-25LPI | PALCE16V8-25PI |

| LATTICE | CYPRESS |
|---|---|
| GAL16V8C-5LJ | PALCE16V8-5JC |
| GAL20V8A | PALCE20V8 |
| GAL20V8B | PALCE20V8 |
| GAL22V10B-7LJ | PALCE22V10-7JC |
| GAL22V10B-7LP | PALCE22V10-7PC |
| GAL22V10B-10LJ | PALCE22V10-10JC |
| GAL22V10B-10LP | PALCE22V10-10PC |
| GAL22V10B-15LD/883 | PALCE22V10-15DMB |
| GAL22V10B-15LJ | PALCE22V10-15JC |
| GAL22V10B-15LJI | PALCE22V10-15JI |
| GAL22V10B-15LP | PALCE22V10-15PC |
| GAL22V10B-15LPI | PALCE22V10-15PI |
| GAL22V10B-15LR/883 | PALCE22V10-15LMB |
| GAL22V10B-20LJI | PALCE22V10-15JI |
| GAL22V10B-20LD/883 | PALCE22V10-15DMB |
| GAL22V10B-20LPI | PALCE22V10-15PI |
| GAL22V10B-20LR/883 | PALCE22V10-15LMB |
| GAL22V10B-25LD/883 | PALCE22V10-25DMB |
| GAL22V10B-25LJ | PALCE22V10-25JC |
| GAL22V10B-25LJI | PALCE22V10-25JI |
| GAL22V10B-25LP | PALCE22V10-25PC |
| GAL22V10B-25LPI | PALCE22V10-25PI |
| GAL22V10B-25LR/883 | PALCE22V10-25LMB |
| GAL22V10B-30LD/883 | PALCE22V10-25DMB |
| GAL22V10B-30LR/883 | PALCE22V10-25LMB |
| GAL22V10C-5LJ | PALCE22V10-5JC |
| GAL22V10C-7LJ | PALCE22V10-7JC |
| GAL22V10C-7PC | PALCE22V10-7PC |

| MACRONIX | CYPRESS |
|---|---|
| PREFIX: MX | PREFIX: CY |
| SUFFIX: P | SUFFIX: P |
| SUFFIX: Q | SUFFIX: J |
| SUFFIX: D | SUFFIX: W |
| SUFFIX: T | SUFFIX: Z |
| 27C1000-45C | 27H010-45C |

| MACRONIX | CYPRESS |
|---|---|
| 27C1000-55C | 27C010-55C |
| 27C1000-70C | 27C010-70C |
| 27C1000-90C | 27C010-90C |
| 27C1000-120C | 27C010-120C |
| 27C1000-150C | 27C010-150C |
| 27C1000-200C | 27C010-200C |
| 27C256-45C | 27C256A-45C |
| 27C256-55C | 27C256A-55C |
| 27C256-70C | 27C256A-70C |
| 27C256-90C | 27C256A-90C |
| 27C256-120C | 27C256A-120C |
| 27C256-150C | 27C256A-150C |
| 27C256-200C | 27C256A-200C |
| 27C512-45C | 27H512-45C |
| 27C512-55C | 27H512-55C |
| 27C512-70C | 27C512-70C |
| 27C512-90C | 27C512-90C |
| 27C512-120C | 27C512-120C |
| 27C512-150C | 27C512-150C |
| 27C512-200C | 27C512-200C |

| MICROCHIP | CYPRESS |
|---|---|
| SUFFIX: J | SUFFIX: W |
| SUFFIX: P | SUFFIX: P |
| SUFFIX: L | SUFFIX: J |
| 27C64-12 | 27C64-120C |
| 27C64-15 | 27C64-150C |
| 27C64-17 | 27C64-150C |
| 27C64-20 | 27C64-200C |
| 27C64-25 | 27C64-200C |
| 27C128-12 | CY27C128-120C |
| 27C128-15 | CY27C128-150C |
| 27C128-17 | CY27C128-150C |
| 27C128-20 | CY27C128-200C |
| 27C128-25 | CY27C128-200C |
| 27C256-10 | CY27C256A-90C |
| 27C256-12 | CY27C256A-120C |
| 27C256-15 | CY27C256A-150C |
| 27C256-20 | CY27C256A-200C |
| 27C512-10 | 27C512-90C |
| 27C512-12 | 27C512-120C |

| MICROCHIP | CYPRESS |
|---|---|
| 27C512-15 | 27C512-150C |
| 27C512-20 | 27C512-200C |
| 27C512-90 | 27C512-90C |
| 27HC256-55 | CY27C256A-55C |
| 27HC256-70 | CY27C256A-70C |
| 27HC256-90 | CY27C256A-90C |

| MICROLINEAR | CYPRESS |
|---|---|
| SUFFIX: CQ | SUFFIX: JC |
| 4663 | 7C4663 |

| MICRON** | CYPRESS |
|---|---|
| PREFIX: MT | PREFIX: CY |
| 5C2568-12 | 7C199-12 |
| 5C2565-12 | 7C195-12 |
| 5C2564-12 | 7C194-12 |
| 5C2561-12 | 7C197-12 |
| 58LC64K18-10 | 7C1031-10 |
| 58LC64K18-9 | 7C1031-8.5 |
| 58LC64K18B2 | 7C1331 |
| 5C1001-15C | 7C107A-15C |
| 5C1001-20C | 7C107A-20C |
| 5C1001-25C | 7C107A-25C |
| 5C1008-20C | 7C109-20C |
| 5C1008-25C | 7C109-25C |
| 5C1008-12C | 7C109A-12C |
| 5C1008-15C | 7C109A-15C |
| 5C1008-20C | 7C109A-20C |
| 5C1601-15 | 7C167A-15C |
| 5C1601-20C | 7C167A-20C |
| 5C1601-25C | 7C167A-25C |
| 5C1601-30 | 7C167A-25C |
| 5C1601-35C | 7C167A-35C |
| 5C1604-15 | 7C168A-15C |
| 5C1604-20C | 7C168A-20C |
| 5C1604-25C | 7C168A-25C |
| 5C1604-30 | 7C168A-25C |
| 5C1604-35C | 7C168A-35C |
| 5C1605-15 | 7C170A-15C |
| 5C1605-20C | 7C170A-20C |
| 5C1605-25C | 7C170A-25C |

| MICRON** | CYPRESS |
|---|---|
| 5C1605-30 | 7C170A-25C |
| 5C1605-35C | 7C170A-35C |
| 5C1608-15 | 7C128A-15C |
| 5C1608-20C | 7C128A-20C |
| 5C1608-25C | 7C128A-25C |
| 5C1608-35C | 7C128A-35C |
| 5C2561-15 | 7C197-15 |
| 5C2561-20 | 7C197-20 |
| 5C2561-25 | 7C197-25C |
| 5C2561-30 | 7C197-25C |
| 5C2561-35 | 7C197-35C |
| 5C2561-45 | 7C197-45C |
| 5C2564-15 | 7C194-15 |
| 5C2564-20 | 7C194-20 |
| 5C2564-25 | 7C194-25C |
| 5C2564-30 | 7C194-25C |
| 5C2564-35 | 7C194-35C |
| 5C2564-45 | 7C194-45C |
| 5C2565-15 | 7C195-15 |
| 5C2565-20 | 7C195-20 |
| 5C2565-25 | 7C195-25C |
| 5C2565-30 | 7C195-25C |
| 5C2565-35 | 7C195-35C |
| 5C2565-45 | 7C195-45C |
| 5C2568-15 | 7C199-15 |
| 5C2568-20 | 7C199-20 |
| 5C2568-25 | 7C199-25C |
| 5C2568-30 | 7C199-25C |
| 5C2568-35 | 7C199-35C |
| 5C2568-45 | 7C199-45C |
| 5C2889-20C | 7C188-20C |
| 5C2889-25C | 7C188-25C |
| 5C6404-15 | 7C164-15C |
| 5C6404-20 | 7C164-20C |
| 5C6404-25 | 7C164-25C |
| 5C6404-30 | 7C164-25C |
| 5C6404-35 | 7C164-35C |
| 5C6405-15 | 7C166-15C |
| 5C6405-20C | 7C166-20C |
| 5C6405-25C | 7C166-25C |
| 5C6405-30 | 7C166-25C |
| 5C6405-35C | 7C166-35C |

| MICRON** | CYPRESS |
|---|---|
| 5C6408-15 | 7C185-15C |
| 5C6408-20C | 7C185-20C |
| 5C6408-25C | 7C185-25C |
| 5C6408-30 | 7C185-25C |
| 5C6408-35C | 7C185-35C |
| 5LC2568-15 | 7C1399-15 |
| 5LC2568-20C | 7C1399-20C |
| 5LC2568-25C | 7C1399-25C |
| 58LC32K32D8L-8.5 | 7C1335-117AC |
| 58LC32K32D8L-10 | 7C1335-100AC |
| 58LC32K32D8L-15 | 7C1335-66AC |
| 58LC32K32G1L-G-8.5 | 7C1337-117AC |
| 58LC32K32G1L-G-10 | 7C1337-100AC |
| 85C1664-30C | 1620HD-30C |
| 85C8128-25 | M1420PD-25C |
| 85C8128-35 | M1420PD-35C |
| 85C8128-45C | 1423PD-45C |

| MITSUBISHI | CYPRESS |
|---|---|
| PREFIX: M5L | PREFIX: CY |
| PREFIX: M5M | PREFIX: CY |
| SUFFIX: AP | SUFFIX: L |
| SUFFIX: FP | SUFFIX: F |
| SUFFIX: K | SUFFIX: D |
| SUFFIX: P | SUFFIX: P |
| 8S1632Z | M1821PZ |
| 8S1632M | M1821PM |
| 8S6432Z | M1831PZ |
| 8S6432M | M1831PM |
| 8S25632Z | M1841PZ |
| 8S25632M | M1841PM |
| 4S12832Z | M1836PZ |
| 4S12832M | M1836PM |
| 21C67P-35 | 7C167A-35C |
| 21C67P-45 | 7C167A-45C |
| 21C67P-55 | 7C167A-45C |
| 21C68P-35 | 7C168A-35C |
| 21C68P-45 | 7C168A-45C |
| 21C68P-55 | 7C168A-45C |
| 27C256-85 | 27C256A-70C |

| MITSUBISHI | CYPRESS |
|---|---|
| 27C256-100 | 27C256A-90C |
| 27C256-120 | 27C256A-120C |
| 27C256-150 | 27C256A-150C |
| 27C256-170 | 27C256A-150C |
| 5165L-70 | 7C186-55C+ |
| 5165L-100 | 7C186-55C+ |
| 5165L-120 | 7C186-55C+ |
| 5165P-70 | 7C186-55C+ |
| 5165P-100 | 7C186-55C+ |
| 5165P-120 | 7C186-55C+ |
| 5178P-45 | 7C186-45C+ |
| 5178P-55 | 7C186-55C+ |
| 5187P-25 | 7C187-25C |
| 5187P-35 | 7C187-35C |
| 5187P-45 | 7C187-45C |
| 5187P-55 | 7C187-45C |
| 5188P-25 | 7C164-25C |
| 51008BFP-xxL | 62128-70SC |
| 51008BVP-xxL | 62128-70ZC |
| 51008BRV-xxL | 62128-70RZC |
| 51008BFP-xxL | 62128L-70SC |
| 51008BVP-xxL | 62128L-70ZC |
| 51T08AFP-xxSL | 62128LL-70SC |
| 51T08AVP-xxSL | 62128LL-70ZC |
| 51008BFP-xxVL | 62128VL-70SC |
| 51008BVP-xxVL | 62128VL-xxVL |
| 51T08AFP-xxVSL | 62128VLL-70SC |
| 51T08AVP-xxVSL | 62128VLL-70ZC |
| 5188P-35 | 7C164-35C |
| 5188P-45 | 7C164-45C |
| 5188P-55 | 7C164-45C |
| 5256CVP-xx | CY62256-70ZC |
| 5256CRV-xx | CY62256-70RZC |
| 5256CFP-xx | CY62256-70SC |
| 5256CFP-xx | CY62256-70SNC |
| 5256CP-xx | CY62256-70PC |
| 5256CVP-xxL | CY62256L-70ZC |
| 5256CRV-xxL | CY62256L-70RZC |
| 5256CFP-xxL | CY62256L-70SNC |
| 5256CVP-xxLL | CY62256L-70ZC |
| 5256CRV-xxLL | CY62256LL-70RC |

| MITSUBISHI | CYPRESS |
|---|---|
| 5256CFP-xxLL | CY62256LL-70SNC |
| 5256CVP-xxVLL | CY62256VL-70ZC |
| 5256CRV-xxVLL | CY62256VL-70RZC |
| 5256CFP-xxVLL | CY62256VL-70SC |
| 5256CFP-xxVLL | CY62256VL-70SNC |
| 5256CVP-xxVXL | CY62256VLL-70ZC |
| 5256CRV-xxVXL | CY62256VLL-70RZC |
| 5256CFP-xxVXL | CY62256VLL-70SNC |
| 5257J-35 | 7C197-35C |
| 5257J-45 | 7C197-45C |
| 5257P-35 | 7C197-35C |
| 5257P-45 | 7C197-45C |
| 5258J-45 | 7C194-45C |
| 5258P-35 | 7C194-35C |
| 5258P-45 | 7C194-45C |
| 52B79P/J | 7C188 |
| 5V1132FP-6 | 7C1335-100AC |
| 5V1132FP-7 | 7C1335-75AC |
| 5V1132FP-8 | 7C1335-66AC |
| 5V1132FP-10 | 7C1335-60AC |
| 5V1132FP-7L | 7C1335L-75AC |
| 5V1132FP-8L | 7C1335L-66AC |
| 5V1132FP-10L | 7C1335L-60AC |
| 5V1132AFP-6 | 7C1335-100AC |
| 5V1132AFP-7 | 7C1335-75AC |
| 5V1132AFP-8 | 7C1335-66AC |
| 5V278 | 7C1399 |

| MMI/AMD | CYPRESS |
|---|---|
| SUFFIX: 883B | SUFFIX: B |
| SUFFIX: F | SUFFIX: F |
| SUFFIX: J | SUFFIX: D |
| SUFFIX: L | SUFFIX: L |
| SUFFIX: N | SUFFIX: P |
| SUFFIX: SHRP | SUFFIX: B |
| PALC22V10/A | PALC22V10-35C |
| PAL20R8M | PLDC20G10-40M |
| PAL20R8C | PLDC20G10-35C |

| MMI/AMD | CYPRESS |
|---|---|
| PAL20R8AM | PLDC20G10-30M |
| PAL20R8AC | PLDC20G10-25C |
| PAL20R8A-2M | PLDC20G10-40M |
| PAL20R8A-2C | PLDC20G10-35C |
| PAL20R6M | PLDC20G10-40M |
| PAL20R6C | PLDC20G10-35C |
| PAL20R6AM | PLDC20G10-30M |
| PAL20R6AC | PLDC20G10-25C |
| PAL20R6A-2M | PLDC20G10-40M |
| PAL20R6A-2C | PLDC20G10-35C |
| PAL20R4M | PLDC20G10-40M |
| PAL20R4C | PLDC20G10-35C |
| PAL20R4AM | PLDC20G10-30M |
| PAL20R4AC | PLDC20G10-25C |
| PAL20R4A-2M | PLDC20G10-40M |
| PAL20R4A-2C | PLDC20G10-35C |
| PAL20L8M | PLDC20G10-40M |
| PAL20L8C | PLDC20G10-35C |
| PAL20L8AM | PLDC20G10-30M |
| PAL20L8AC | PLDC20G10-25C |
| PAL20L8A-2M | PLDC20G10-40M |
| PAL20L8A-2C | PLDC20G10-35C |
| PAL20L2M | PLDC20G10-40M |
| PAL20L2C | PLDC20G10-35C |
| PAL20L10M | PLDC20G10-40M |
| PAL20L10C | PLDC20G10-35C |
| PAL20L10AM | PLDC20G10-30M |
| PAL20L10AC | PLDC20G10-35C |
| PAL18L4M | PLDC20G10-40M |
| PAL18L4C | PLDC20G10-35C |
| PAL16R8M | PALC16R8-40M |
| PAL16R8D-4C | PALC1648L-25C |
| PAL16R8C | PALC16R8-35C |
| PAL16R8BM | PALC16R8-20M |
| PAL16R8B-4M | PALC16R8-40M |
| PAL16R8B-4C | PALC16R8L-35C |
| PAL16R8B-2M | PALC16R8-30M |
| PAL16R8B-2C | PALC16R8-25C |
| PAL16R8AM | PALC16R8-30M |
| PAL16R8AC | PALC16R8-25C |
| PAL16R8A-4M | PALC16R8-40M |
| PAL16R8A-4C | PALC16R8L-35C |

| MMI/AMD | CYPRESS |
|---|---|
| PAL16R8A-2M | PALC16R8-40M |
| PAL16R8A-2C | PALC16R8-35C |
| PAL16R6M | PALC16R6-40M |
| PAL16R6D-4C | PALC16R6L-25C |
| PAL16R6C | PALC16R6-35C |
| PAL16R6BM | PALC16R6-20M |
| PAL16R6B-4M | PALC16R6-40M |
| PAL16R6B-4C | PALC16R6L-35C |
| PAL16R6B-2M | PALC16R6-30M |
| PAL16R6B-2C | PALC16R6-25C |
| PAL16R6AM | PALC16R6-30M |
| PAL16R6AC | PALC16R6-25C |
| PAL16R6A-4M | PALC16R6-40M |
| PAL16R6A-4C | PALC16R6L-35C |
| PAL16R6A-2M | PALC16R6-40M |
| PAL16R6A-2C | PALC16R6-35C |
| PAL16R4M | PALC16R4-40M |
| PAL16R4D-4C | PALC16R4L-25C |
| PAL16R4C | PALC16R4-35C |
| PAL16R4BM | PALC16R4-20M |
| PAL16R4B-4M | PALC16R4-40M |
| PAL16R4B-4C | PALC16R4L-35C |
| PAL16R4B-2M | PALC16R4-30M |
| PAL16R4B-2C | PALC16R4-25C |
| PAL16R4AM | PALC16R4-30M |
| PAL16R4AC | PALC16R4-25C |
| PAL16R4A-4M | PALC16R4-40M |
| PAL16R4A-4C | PALC16R4L-35C |
| PAL16R4A-2M | PALC16R4-40M |
| PAL16R4A-2C | PALC16R4-35C |
| PAL16L8M | PALC16L8-40M |
| PAL16L8D-4M | PALC16L8-30M |
| PAL16L8D-4C | PALC16L8L-25C |
| PAL16L8C | PALC16L8-35C |
| PAL16L8BM | PALC16L8-20M |
| PAL16L8B-4M | PALC16L8-40M |
| PAL16L8B-4C | PALC16L8L-35C |
| PAL16L8B-2M | PALC16L8-30M |
| PAL16L8B-2C | PALC16L8-35C |
| PAL16L8AM | PALC16L8-30M |
| PAL16L8AC | PALC16L8-25C |
| PAL16L8A-4M | PALC16L8-40M |

| MMI/AMD | CYPRESS |
|---|---|
| PAL16L8A-4C | PALC16L8L-35C |
| PAL16L8A-2M | PALC16L8-40M |
| PAL16L8A-2C | PALC16L8-35C |
| PAL16L6M | PLDC20G10-40M |
| PAL16L6C | PLDC20G10-35C |
| PAL14L8M | PLDC20G10-40M |
| PAL14L8C | PLDC20G10-35C |
| PAL12L10M | PLDC20G10-40M |
| PAL12L10C | PLDC20G10-35C |

| MOSEL | CYPRESS |
|---|---|
| PREFIX: MS | PREFIX: CY |
| 6264L-xxFC | 6264-70SC |
| 6264L-xxFC | 6264-70SNC |
| 62256L-70FC | 62256-70SC |
| 62256L-70FC | 62256-70SNC |
| 62256L-70FC | 62256L-70SNC |
| 62256LL-70FC | 62256LL-70SNC |

| MOTOROLA | CYPRESS |
|---|---|
| PREFIX: MCM | PREFIX: CY |
| SUFFIX: BXAJC | SUFFIX: MB |
| SUFFIX: P | SUFFIX: P |
| SUFFIX: S | SUFFIX: D |
| SUFFIX: Z | SUFFIX: L |
| 1423-45 | 7C168A-45C+ |
| 2016H-45 | 6116A-45C |
| 2018-35 | 7C128A-35C |
| 2167H-35 | 7C167A-35C |
| 2167H-45 | 7C167A-45C |
| 6164-45 | 7C186-45C |
| 6168-35 | 7C168A-35C+ |
| 6205D-20 | 7C188-20C |
| 6205D-25 | 7C188-25C |
| 6206D-12 | 7C199-12C |
| 6206D-15 | 7C199-15 |
| 6206D-20 | 7C199-20 |
| 6206D-25 | 7C199-25 |
| 6206D-35 | 7C199-35C |
| 6207C-15 | 7C197-15 |
| 6207C-20 | 7C197-20 |

| MOTOROLA | CYPRESS |
|---|---|
| 6207C-25 | 7C197-25 |
| 6207C-35 | 7C197-35 |
| 6208C-12 | 7C194-12C |
| 6208C-15 | 7C194-15C |
| 6208C-20 | 7C194-20 |
| 6208C-25 | 7C194-25 |
| 6208C-35 | 7C194-35 |
| 6209C-12 | 7C195-12C |
| 6209-15 | 7C195-15C |
| 6209C-20 | 7C195-20C |
| 6209C-25 | 7C195-25C |
| 6226-20 | 7C109-20C |
| 6226-25 | 7C109-25C |
| 6229B | 7C1006 |
| 6264-15C | 7C185-15C |
| 6264-25 | 7C185-25C |
| 6264-25 | 7C186-25C |
| 6264-30 | 7C185-25C |
| 6264-30 | 7C186-25C |
| 6264-35 | 7C185-35C |
| 6264-35 | 7C186-35C |
| 6264-45 | 7C186-45C |
| 6265C-25 | 7C182-25C |
| 6268P25 | 7C168A-25C |
| 6268P35 | 7C168A-35C |
| 6268P40 | 7C168A-40C |
| 6268P45 | 7C168A-45C |
| 6269P20 | 7C169A-20C |
| 6269P25 | 7C169A-25C |
| 6269P35 | 7C169A-35C |
| 6270-20 | 7C170A-20C |
| 6270-25 | 7C170A-25C |
| 6270-35 | 7C170A-35C |
| 6287-15 | 7C187-15C |
| 6287-20 | 7C187-20C |
| 6287-25 | 7C187-25C |
| 6287-35 | 7C187-35C |
| 6288-15 | 7C164-15C |
| 6288-25 | 7C164-25C |
| 6288-30 | 7C164-25C |
| 6288-35 | 7C164-35C |
| 6290-15 | 7C166-15C |

# Product Line Cross Reference

| MOTOROLA | CYPRESS |
|---|---|
| 6290-20 | 7C166-20C |
| 6290-25 | 7C166-25C |
| 6290-35 | 7C166-35C |
| 62V06D-20 | 7C1399-20C |
| 62V06D-25 | 7C1399-25C |
| 6306D | 7C1399 |
| 6726A-12 | 7C109A-12C |
| 6726A-15 | 7C109A-15C |
| 6726A-20 | 7C109A-20C |
| 67H518-9 | 7C178-8.5 |
| 67H518-12 | 7C178-12 |
| 67H518-9 | 7C1031-8.5 |
| 67H518-12 | 7C1031-12 |

| NATIONAL | CYPRESS |
|---|---|
| PREFIX: DM | PREFIX: CY |
| PREFIX: GAL | PREFIX: None |
| PREIFX: IDM | PREFIX: CY |
| PREFIX: NM | PREFIX: CY |
| PREFIX: NM | PREFIX: CY |
| SUFFIX: A | SUFFIX: Z |
| SUFFIX: J | SUFFIX: D |
| SUFFIX: N | SUFFIX: P |
| SUFFIX: Q | SUFFIX: W |
| SUFFIX: V | SUFFIX: J |
| 18L4C | PLDC20G10-35C |
| 18L4M | PLDC20G10-40M |
| 20L2M | PLDC20G10-40M |
| 2147H | 2147-C |
| 2147H | 7C147-C |
| 2148H | 7C148-C |
| 2148H | 2148-C |
| 2148H | 21L48-C |
| 27C010-120C | 27C010-120C |
| 27C010-150C | 27C010-150C |
| 27C010-200C | 27C010-200C |
| 27C64-100C | 27C64-90C |
| 27C64-120C | 27C64-120C |
| 27C64-150C | 27C64-150C |
| 27C64-200C | 27C64-200C |
| 27C128-12C | 27C128-120C+ |

| NATIONAL | CYPRESS |
|---|---|
| 27C128-15C | 27C128-150C+ |
| 27C128-20C | 27C128-200C+ |
| 27C256-100 | 27C256A-90C |
| 27C256-120 | 27C256A-120C |
| 27C256-150 | 27C256-150C+ |
| 27C256-200 | 27C256-200C+ |
| 27C512-120C | 27C512-120C |
| 27C512-150C | 27C512-150C |
| 27C512-200C | 27C512-200C |
| 27P010-70C | 27C010-70C |
| 27P010-90C | 27C010-90C |
| 27P010-100C | 27C010-90C |
| 77LS181 | 7C282A-45M |
| 77S181 | 7C282A-45M |
| 77S181A | 7C282A-45M |
| 77S281 | 7C281A-45M |
| 77S281A | 7C281A-45M |
| 77S401 | 7C401-10M |
| 77S401A | 7C401-10M |
| 77S402 | 7C402-10M |
| 77S402A | 7C402-10M |
| 77SR181 | 7C235A-40M |
| 77SR476 | 7C225A-40M- |
| 77SR476B | 7C225A-40M- |
| 85S07A | 7C128-45C+ |
| 87LS181 | 7C282A-45C |
| 87S181 | 7C282A-45C |
| 87S281 | 7C281A-45C |
| 87S281A | 7C281A-45C |
| 87S401 | 7C401-10C |
| 87S401A | 7C401-15C |
| 87S402 | 7C402-10C |
| 87S402A | 7C402-15C |
| 87SR181 | 7C235-40C |
| 87SR476 | 7C225A-40C |
| 87SR476B | 7C225A-30C |
| 93L422A | 7C122-C |
| C27C53-55 | 27C256-55C |
| C27C53-70 | 27C256-70C |
| DP892CV | 7B8392-PC |
| DP8892CN | 7B8392-JC |
| GAL22V10-15C | PALCE22V10-15C |

| NATIONAL | CYPRESS |
|---|---|
| GAL22V10-20I | PALCE22V10-15I |
| GAL22V10-20M | PALCE22V10-15M |
| GAL22V10-25C | PALCE22V10-25C |
| GAL22V10-30I | PALCE22V10-25I |
| GAL22V10-30M | PALCE22V10-25M |
| NMF512X9-15 | 7C421-15 |
| NMF512X9-25 | 7C421-25 |
| NMF2048X9-20 | 7C429-20 |
| NMF4096X9A-25 | 7C433-25 |
| PAL164A2M | PALC16R4-40M |
| PAL16L8A2C | PALC16L8-35C |
| PAL16L8A2M | PALC16L8-40M |
| PAL16L8AC | PALC16L8-25C |
| PAL16L8AM | PALC16L8-30M |
| PAL16L8B2C | PALC16L8-25C |
| PAL16L8B2M | PALC16L8-30M |
| PAL16L8B4C | PALC16L8L-35C |
| PAL16L8B4M | PALC16L8-40M |
| PAL16L8BM | PALC16L8-20M |
| PAL16L8C | PALC16L8-35C |
| PAL16L8M | PALC16L8-40M |
| PAL16R4A2C | PALC16R4-35C |
| PAL16R4AC | PALC16R4-25C |
| PAL16R4AM | PALC16R4-30M |
| PAL16R4B2C | PALC16R4-25C |
| PAL16R4B2M | PALC16R4-30M |
| PAL16R4B4C | PALC16R4L-35C |
| PAL16R4B4M | PALC16R4-40M |
| PAL16R4BM | PALC16R4-20M |
| PAL16R4C | PALC16R4-35C |
| PAL16R4M | PALC16R4-40M |
| PAL16R6A2C | PALC16R6-35C |
| PAL16R6A2M | PALC16R6-40M |
| PAL16R6AC | PALC16R6-25C |
| PAL16R6AM | PALC16R6-30M |
| PAL16R6B2C | PALC16R6-25C |
| PAL16R6B2M | PALC16R6-30M |
| PAL16R6B4C | PALC16R6L-35C |
| PAL16R6B4M | PALC16R6-40M |
| PAL16R6BM | PALC16R6-20M |
| PAL16R6C | PALC16R6-35C |
| PAL16R6M | PALC16R6-40M |

| NATIONAL | CYPRESS |
|---|---|
| PAL16R8A2C | PALC16R8-35C |
| PAL16R8A2M | PALC16R8-40M |
| PAL16R8AC | PALC16R8-25C |
| PAL16R8AM | PALC16R8-30M |
| PAL16R8B2C | PALC16R8-25C |
| PAL16R8B2M | PALC16R8-30M |
| PAL16R8B4C | PALC16R8L-35C |
| PAL16R8B4M | PALC16R8-40M |
| PAL16R8BM | PALC16R8-20M |
| PAL16R8C | PALC16R8-35C |
| PAL16R8M | PALC16R8-40M |
| PAL20L2C | PLDC20G10-35C |
| PAL20L8AC | PLDC20G10-25C |
| PAL20L8AM | PLDC20G10-30M |
| PAL20L8BC | PLDC20G10-25C |
| PAL20L8BM | PLDC20G10-30M |
| PAL20L8C | PLDC20G10-35C |
| PAL20L8M | PLDC20G10-40M |
| PAL20L10B2C | PLDC20G10-25C |
| PAL20L10B2M | PLDC20G10-30M |
| PAL20L10C | PLDC20G10-35C |
| PAL20L10M | PLDC20G10-40M |
| PAL20R4AC | PLDC20G10-25C |
| PAL20R4AM | PLDC20G10-30M |
| PAL20R4BC | PLDC20G10-25C |
| PAL20R4BM | PLDC20G10-30M |
| PAL20R4C | PLDC20G10-35C |
| PAL20R4M | PLDC20G10-40M |
| PAL20R6AC | PLDC20G10-25C |
| PAL20R6AM | PLDC20G10-30M |
| PAL20R6BC | PLDC20G10-25C |
| PAL20R6BM | PLDC20G10-30M |
| PAL20R6C | PLDC20G10-35C |
| PAL20R6M | PLDC20G10-40M |
| PAL20R8AC | PLDC20G10-25C |
| PAL20R8AM | PLDC20G10-30M |
| PAL20R8BC | PLDC20G10-25C |
| PAL20R8BM | PLDC20G10-30M |
| PAL20R8C | PLDC20G10-35C |
| PAL20R8M | PLDC20G10-40M |

| NEC | CYPRESS |
|---|---|
| PREFIX: uPD | PREFIX: CY |
| SUFFIX: C | SUFFIX: P |
| SUFFIX: D | SUFFIX: D |
| SUFFIX: K | SUFFIX: L |
| SUFFIX: L | SUFFIX: F |
| 2147A | 7C147-C |
| 2149 | 2149-C |
| 2149 | 7C149-C |
| 2167-2 | 7C167A-C |
| 27HC65-25 | 7C263/4-25C |
| 27HC65-35 | 7C263/4-35C |
| 27HC65-45 | 7C263/4-45C |
| 431000AGW-xxL | 62128-70SC |
| 431000GZ-xxL | 62128-70ZC |
| 431000GZM-xxL | 62128-70RZC |
| 431000GW-xxL | 62128L-70SC |
| 431000GZ-xxL | 62128L-70ZC |
| 431000AGW-xxLL | 62128LL-70SC |
| 431000GZ-xxLL | 62128LL-70ZC |
| 4311-45 | 7C167A-45C |
| 4311-55 | 7C167A-45C |
| 4312-32L | 7C133-5L |
| 43254C-35 | 7C194-35 |
| 43254C-45 | 7C194-45 |
| 43256BGX-xxL-EJA | CY62256-70ZC |
| 43256BGX-xxL-EKA | CY62256-70RZC |
| 43256BGU-xxL | CY62256-70SC |
| 43256BGU-xxL | CY62256-70SNC |
| 43256BCZ-xx | CY62256-70PC |
| 43256BGX-xxL-EJA | CY62256L-70ZC |
| 43256BGX-xxL-EKA | CY62256L-70RZC |
| 43256BGU-xxL | CY62256L-70SNC |
| 43256BGX-xxLL-EJA | CY62256L-70ZC |
| 43256BGX-xxL-EKA | CY62256LL-70RC |
| 43256BGU-xxLL | CY62256LL-70SNC |
| 43256BGX-Bxx-EJA | CY62256-70ZC |
| 43256BGX-Bxx-EKA | CY62256-70RZC |
| 43256BGU-Bxx | CY62256V-70SC |
| 43256BGU-Bxx | CY62256V-70SNC |

| NEC | CYPRESS |
|---|---|
| 4361 | 7C187-C |
| 4362 | 7C164-C |
| 4363 | 7C166-C |
| 43259-20 | 7C188-20 |
| 43259-25 | 7C188-25 |
| 431001-20 | 7C107A-20C |
| 431001-25 | 7C107A-25C |
| 431004-20 | 7C106A-20C |
| 431004-25 | 7C106A-25C |
| 431008-15 | 7C109A-15 |
| 431008-20 | 7C109A-20 |
| 431008-20 | 7C109-20 |
| 431232LGF-A8 | 7C1335-66AC |
| 431232LGF-A10 | 7C1335-60AC |

| OKI | CYPRESS |
|---|---|
| PREFIX: MSM | PREFIX: CY |
| 27128A-12C | 27C128-120C+ |
| 27128A-15C | 27C128-150C+ |
| 27128A-20C | 27C128-200C+ |
| 27256-100 | 27C256A-90C |
| 27256-120 | 27C256A-120C |
| 27256-150 | 27C256A-150C |
| 27256-200 | 27C256A-200C |
| 27256H-55 | 27C256A-55C |
| 27256H-70 | 27C256A-70C |

| PARADIGM | CYPRESS |
|---|---|
| PREFIX: PDM | PREFIX: CY |
| 41251L | 7C191-C |
| 41251LB | 7C191-MB* |
| 41251S | 7C191-C |
| 41251SB | 7C191-MB |
| 41252L | 7C192-C |
| 41252LB | 7C192-MB* |
| 41252S | 7C192-C |
| 41252SB | 7C192-MB |
| 41256L | 7C199/8-C* |
| 41256LB | 7C199/8-MB* |
| 41256S | 7C199/8-C |
| 41256SB | 7C199/8-MB |

| PARADIGM | CYPRESS |
|---|---|
| 41258L | 7C194-C* |
| 41258LB | 7C194-B* |
| 41258S | 7C194-C |
| 41258SB | 7C194-B |

| PERFORMANCE | CYPRESS |
|---|---|
| PREFIX: P | PREFIX: CY |
| SUFFIX: L | SUFFIX: L |
| SUFFIX: S | SUFFIX: S |
| 41256-35 | 7C199-35 |
| 41256-45 | 7C199-45 |
| 4C1256-25 | 7C199-25 |
| 4C1256-35 | 7C199-35 |
| 4C1256-45 | 7C198-45 |
| 4C1257-25 | 7C197-25 |
| 4C1257-35 | 7C197-35 |
| 4C1257-45 | 7C197-45 |
| 4C1258-25 | 7C194-25 |
| 4C1258-35 | 7C194-35 |
| 4C1258-45 | 7C194-45 |
| 4C150-12C | 7C150-12C |
| 4C150-15C | 7C150-15C |
| 4C150-15M | 7C150-15M |
| 4C150-20C | 7C150-15C |
| 4C150-20M | 7C150-15M |
| 4C150-25C | 7C150-25C |
| 4C150-25M | 7C150-25M |
| 4C150-35M | 7C150-35M |
| 4C164DW-20C | 7C186-20C |
| 4C164DW-25C | 7C186-25C |
| 4C164DW-25M | 7C186A-25M |
| 4C164DW-35C | 7C186-35C |
| 4C164DW-35M | 7C186A-35M |
| 4C164DW-55C | 7C186-55C |
| 4C164P-20C | 7C185-20C |
| 4C164P-25C | 7C185-25C |
| 4C164P-25M | 7C185A-25M |
| 4C164P-35C | 7C185-35C |
| 4C164P-35M | 7C185A-35M |
| 4C164P-45M | 7C185A-45M |
| 4C1681-25C | 7C171A-25C |

| PERFORMANCE | CYPRESS |
|---|---|
| 4C1681-35C | 7C171A-35C |
| 4C1681-35M | 7C171A-35M |
| 4C1681-45C | 7C171A-45C |
| 4C1681-45M | 7C171A-45M |
| 4C1682-25C | 7C172A-25C |
| 4C1682-35C | 7C172A-35C |
| 4C1682-35M | 7C172A-35M |
| 4C1682-45C | 7C172A-45C |
| 4C1682-45M | 7C172A-45M |
| 4C169-25C | 7C169A-25C |
| 4C169-30C | 7C169A-25C |
| 4C169-35C | 7C169A-35C |
| 4C169-35M | 7C169A-35M |
| 4C169-45M | 7C169A-45M |
| 4C187-20C | 7C187-20C |
| 4C187-25C | 7C187-25C |
| 4C187-25M | 7C187A-25M |
| 4C187-35M | 7C187A-35M |
| 4C188-20C | 7C164-20C |
| 4C188-25C | 7C164-25C |
| 4C188-25M | 7C164A-25M |
| 4C188-35C | 7C164-35C |
| 4C188-35M | 7C164A-35M |
| 4C188-45M | 7C164A-45M |
| 4C198-20C | 7C166-20C |
| 4C198-25C | 7C166-25C |
| 4C198-25M | 7C166A-25M |
| 4C198-35C | 7C166-35C |
| 4C198-35M | 7C166A-35M |
| 4C198-45M | 7C166A-45M |
| 4C1981-20C | 7C161-20C |
| 4C1981-25C | 7C161-25C |
| 4C1981-25M | 7C161A-25M |
| 4C1981-35C | 7C161-35C |
| 4C1981-35M | 7C161A-35M |
| 4C1982-20C | 7C162-20C |
| 4C1982-25C | 7C162-25C |
| 4C1982-25M | 7C162A-25M |
| 4C1982-35C | 7C162-35C |
| 4C1982-35M | 7C162A-35M |
| 93U422-35C | 7C122-15C |
| 93U422-35C | 7C122-25C |

| PERFORMANCE | CYPRESS |
|---|---|
| 93U422-35C | 7C122-35C |
| 93U422-35M | 7C122-25M |
| 93U422-35M | 7C122-35M |

| PHILIPS-SIGNETICS | CYPRESS |
|---|---|
| SUFFIX: G | SUFFIX: L |
| SUFFIX: N | SUFFIX: P |
| SUFFIX: R | SUFFIX: F |
| SUFFIX: F | SUFFIX: W |
| SUFFIX: A | SUFFIX: J |
| 27C256-12 | 27C256A-120C |
| 27C256-15 | 27C256A-150C |
| 27C256-17 | 27C256A-150C |
| 27C256-20 | 27C256A-200C |
| 27C256-90 | 27C256A-90C |
| 27HC641-45C | 7C263/4-45C |
| 27HC641-55C | 7C263/4-55C |
| N74S189 | 74S189C |
| N82HS321 | 7C243/4-45C |
| N82HS321A | 7C243/4-35C |
| N82HS321B | 7C243/4-30C |
| N82HS321C | 7C243/4-25C |
| N82HS641 | 7C263/4-55C |
| N82HS641A | 7C263/4-45C |
| N82HS641B | 7C263/4-35C |
| N82HS641C | 7C263/4-25C |
| N82LHS191-3 | 7C291A-35C |
| N82LHS191-6 | 7C292A-35C |
| N82S181 | 7C281/2A-45C |
| N82S181A | 7C281/2A-45C |
| N82S181C | 7C281/2A-30C |
| N82S191-3 | 7C291A-50C |
| N82S191-6 | 7C292A-50C |
| N82S191A-3 | 7C291A-50C |
| N82S191A-6 | 7C292A-50C |
| N82S191C-3 | 7C291A-35C |
| N82S191C-6 | 7C292A-35C |
| NE83Q92A | 7B8392-PC |
| NE83Q92N | 7B8392-JC |
| S82HS641 | 7C263/4-55M |
| S82LS181 | 7C282A-45M |

| PHILIPS-SIGNETICS | CYPRESS |
|---|---|
| S82S181 | 7C282A-45M |
| S82S181A | 7C282A-45M |
| S82S191-3 | 7C291A-50M |
| S82S191-6 | 7C292A-50M |
| S82S191A-3 | 7C291A-50M |
| S82S191A-6 | 7C292A-50M |
| S82S191B-3 | 7C291A-50M |
| S82S191B-6 | 7C292A-50M |

| PMC-SIERRA | CYPRESS |
|---|---|
| PREFIX: PM | PREFIX: CY |
| PREFIX: NC | SUFFIX: NC |
| 5346 | 7C955 |

| QUALITY SEMI | CYPRESS |
|---|---|
| Q7201 | 7C421 |
| Q7202 | 7C425 |
| Q7203 | 7C429 |
| Q7204 | 7C433 |
| Q72211 | 7C4211 |
| Q72221 | 7C4221 |
| Q72231 | 7C4251 |
| Q72241 | 7C4241 |
| Q72215 | 7C4215 |
| Q72225 | 7C4225 |
| Q7025A | 7C025 |

| SAMSUNG | CYPRESS |
|---|---|
| PREFIX: KM | PREFIX: CY |
| 18V87-8 | 7C1031-8.5 |
| 61257A-25 | 7C197-25C |
| 61257A-35 | 7C197-35C |
| 61257A-45 | 7C197-45C |
| 62256CLTG-xxL | 62256-70ZC |
| 62256CLRG-xxL | 62256-70RZC |
| 62256CLG-xxL | 62256-70SC |
| 62256CLG-xxL | 62256-70SNC |
| 62256CLP-xxL | 62256-70PC |
| 62256CLTG-xx | 62256L-70ZC |
| 62256CLRG-xx | 62256L-70RZC |

| SAMSUNG | CYPRESS |
|---|---|
| 62256CLG-xx | 62256L-70SNC |
| 62256CLTG-xxL | 62256L-70ZC |
| 62256CLRG-xxL | 62256LL-70RC |
| 62256CLG-xxL | 62256LL-70SNC |
| 62256CLTG-xxLV | 62256-70ZC |
| 62256CLRG-xxLV | 62256-70RZC |
| 62256CLG-xxLV | 62256V-70SC |
| 62256CLG-xxLV | 62256V-70SNC |
| 62256CLTG-xxLV | 62256VL-70ZC |
| 62256CLRG-xxLV | 62256VL-70RZC |
| 62256CLG-xxLV | 62256VL-70SC |
| 62256CLG-xxLV | 62256VL-70SNC |
| 62256CLTG-xxLV | 62256VLL-70ZC |
| 62256CLRG-xxLV | 62256VLL-70RZC |
| 62256CLG-xxLV | 62256VLL-70SNC |
| 6264BLG-xx | 6264-70SC |
| 6264BLG-xx | 6264-70SNC |
| 64257A-25 | 7C194-25C |
| 64257A-35 | 7C194-35C |
| 64257A-45 | 7C194-45C |
| 64258B-15 | 7C194-15C |
| 64258B-20 | 7C194-20C |
| 64259B-15 | 7C196-15C |
| 64259B-20 | 7C196-20C |
| 641001-20 | 7C106A-20C |
| 681000BLG-xx | 62128-70SC |
| 681000BLT-xx | 62128-70ZC |
| 681000BLR-xx | 62128-70RZC |
| 681000BLG-xx | 62128L-70SC |
| 681000BLT-ss | 62128L-70ZC |
| 681000BLG-xxL | 62128LL-70SC |
| 681000BL6-xxL | 62128LL-70ZC |
| 68U1000BLG-xx | 62128VL-70SC |
| 681U000BLT-ss | 62128VL-70ZC |
| 681U000BLG-xxL | 62128VLL-70SC |
| 681U000BL6-xxL | 62128VLL-70ZC |
| 681001-20 | 7C109A-20C |
| 681002-15 | 7C1009-15C |
| 681002-20 | 7C1009-20C |
| 68257-12 | 7C199-12C |
| 68257-15 | 7C199-15C |
| 68257C | 7C1399 |

| SAMSUNG | CYPRESS |
|---|---|
| 68512AL | 7C1512 |
| 718B514-8 | 7C178-8.5 |
| 732V592 | 7C1335-L |
| 75C01A-15 | 7C421-15 |
| 75C01A-20 | 7C421-20C |
| 75C01A-25 | 7C421-25C |
| 75C01A-35 | 7C421-30C |
| 75C01A-50 | 7C421-40C |
| 75C01A-80 | 7C421-65C |
| 75C01AP-20 | 7C420-20C |
| 75C01AP-25 | 7C420-25C |
| 75C01AP-35 | 7C420-35C |
| 75C01AP-50 | 7C420-50C |
| 75C01AP-80 | 7C420-80C |
| 75C02A-15 | 7C425-15 |
| 75C02A-20 | 7C425-20C |
| 75C02A-25 | 7C425-25C |
| 75C02A-35 | 7C425-30C |
| 75C02A-50 | 7C425-40C |
| 75C02A-80 | 7C425-65C |
| 75C02AP-20 | 7C424-20C |
| 75C02AP-25 | 7C424-25C |
| 75C02AP-35 | 7C424-30C |
| 75C02AP-50 | 7C424-40C |
| 75C02AP-80 | 7C424-65C |
| 75C03A-15 | 7C429-15C |
| 75C03A-20 | 7C429-20C |
| 75C03A-25 | 7C429-25C |
| 75C03A-35 | 7C429-30C |
| 75C03A-50 | 7C429-40C |
| 75C03A-80 | 7C429-65C |
| 75C03AP-20 | 7C428-20C |
| 75C03AP-25 | 7C428-25C |
| 75C03AP-35 | 7C428-30C |
| 75C03AP-50 | 7C428-40C |
| 75C03AP-80 | 7C428-65C |
| 75C102A-20 | 7C425-20C |
| 75C102A-25 | 7C425-25C |
| 75C102A-35 | 7C425-25C |
| 75C102A-80 | 7C425-65C |

| SGS-THOMSON | CYPRESS |
|---|---|
| PREFIX: M | PREFIX: CY |
| SUFFIX: F1 | SUFFIX: W |
| SUFFIX: B1 | SUFFIX: P |
| SUFFIX: C1 | SUFFIX: J |
| SUFFIX: N1 | SUFFIX: Z |
| 27256-150 | 27C256A-150C |
| 27256-170 | 27C256A-150C |
| 27256-200 | 27C256A-200C |
| 27C64A-12 | 27C64-120C |
| 27C64A-15 | 27C64-150C |
| 27C64A-20 | 27C64-200C |
| 27C64A-25 | 27C64-200C |
| 27C64A-30 | 27C64-200C |
| 27C128A-12 | 27C128-120C+ |
| 27C128A-15 | 27C128-150C+ |
| 27C128A-20 | 27C128-200C+ |
| 27C256B-80 | 27C256A-70C |
| 27C256B-90 | 27C256A-90C |
| 27C256B-100 | 27C256A-90C |
| 27C256B-120 | 27C256A-120C |
| 27C512-10 | 27C512-90C |
| 27C512-12 | 27C512-120C |
| 27C512-15 | 27C512-150C |
| 27C512-20 | 27C512-200C |
| 27C512-25 | 27C512-200C |
| 27C512-80 | 27C512-70C |
| 27C512-90 | 27C512-90C |
| 27C1001-60X | 27H010-55C |
| 27C1001-70 | 27C010-70C |
| 27C1001-90 | 27C010-90C |
| 27C1001-120 | 27C010-120C |
| 27C1001-150 | 27C010-150C |
| 27C1001-200 | 27C010-200C |
| 629032-E1 | 7C188 |

| SHARP | CYPRESS |
|---|---|
| PREFIX: LH | PREFIX: CY |
| 5168N-xx | 6264-70SC |
| 5168N-xx | 6264-70SNC |
| 52251-35 | 7C197-35C |
| 52251-45 | 7C197-45C |

| SHARP | CYPRESS |
|---|---|
| 52252-35 | 7C194-35C |
| 52252-45 | 7C194-45C |
| 52254D-25 | 7C199-25C |
| 52259 | 7C188 |
| 52B256T-xx | CY62256-70ZC |
| 52B256TR-xx | CY62256-70RZC |
| 52B256N-xx | CY62256-70SC |
| 52B256N-xx | CY62256-70SNC |
| 52B256T-xxL | CY62256L-70ZC |
| 52B256TR-xxL | CY62256L-70RZC |
| 52B256N-xxLL | CY62256L-70SNC |
| 52B256T-xxLL | CY62256L-70ZC |
| 52B256TR-xxLL | CY62256LL-70RC |
| 52B256N-xxLL | CY62256LL-70SNC |
| 5481-15 | 7C408A-15C |
| 5481-25 | 7C408A-25C |
| 5481-35 | 7C408A-35C |
| 5491-15 | 7C409A-15C |
| 5491-25 | 7C409A-25C |
| 5491-35 | 7C409A-35C |
| 5496-20 | 7C420-20C |
| 5496-35 | 7C420-30C |
| 5496-50 | 7C420-40C |
| 5496D-15 | 7C421-15 |
| 5496D-20 | 7C421-20C |
| 5496D-35 | 7C421-30C |
| 5496D-50 | 7C421-40C |
| 5497-20 | 7C424-20C |
| 5497-35 | 7C424-30C |
| 5497-50 | 7C424-40C |
| 5497D-15 | 7C425-15 |
| 5497D-20 | 7C425-20C |
| 5497D-35 | 7C425-30C |
| 5497D-50 | 7C425-40C |

| SMOS | CYPRESS |
|---|---|
| PREFIX: SRM | PREFIX: CY |
| 20100SLMxx | 62128-70SC |
| 20100SLTMxx | 62128-70ZC |
| 20256LTMxx | 62256-70ZC |
| 20256LRMxx | 62256-70RZC |

| SMOS | CYPRESS |
|---|---|
| 20256LMxx | 62256-70SC |
| 20256LMxx | 62256-70SNC |
| 20256LTMxx | 62256L-70ZC |
| 20256LRMxx | 62256L-70RZC |
| 20256LMxx | 62256L-70SNC |

| SONY | CYPRESS |
|---|---|
| PREFIX: CXK | PREFIX: CY |
| 51256P-35 | 7C197-35 |
| 51256P-45 | 7C197-45 |
| 55464-20 | 7C194-20C |
| 55464-25 | 7C194-25C |
| 581001M-xxL | 62128-70SC |
| 581100TM-xxL | 62128-70ZC |
| 581100YM-xxL | 62128-70RZC |
| 581001M-xxL | 62128L-70SC |
| 581100TM-xxL | 62128L-70ZC |
| 581001M-xxLL | 62128LL-70SC |
| 581100TM-xxLL | 62128LL-70ZC |
| 581100TM-xxLL | 62128LL-70SC |
| 581100TM-xxL | 62128V-70ZC |
| 581000M-xxLB | 62128VL-70SC |
| 581100TM-xxL | 62128VL-70ZC |
| 581000M-xxLB | 62128VLL-70SC |
| 5811005M-xxLL | 62128VLL-70ZC |
| 58257ATM-xxL | CY62256-70ZC |
| 58257AYM-xxL | CY62256-70RZC |
| 58257AM-xxL | CY62256-70SC |
| 58257AM-xxL | CY62256-70SNC |
| 58257AP-xxL | CY62256L-70ZC |
| 58257ATM-xxL | CY62256L-70RZC |
| 58257AYM-xxL | CY62256L-70SNC |
| 58257AM-xxLL | CY62256L-70ZC |
| 58257AYM-xxLL | CY62256LL-70RC |
| 58257AM-xxLL | CY62256LL-70SNC |
| 58258A-15 | 7C199-15C |
| 58258A-20 | 7C199-20C |
| 58258A-25 | 7C199-25C |
| 5864BM-xxL | 6264-70SC |
| 5864BM-xxL | 6264-70SNC |
| 59288-20C | 7C188-20C |

| SONY | CYPRESS |
|---|---|
| 59288-25C | 7C188-25C |

| TDK/SSI | CYPRESS |
|---|---|
| SSI78Q8392-CP | CY7B8392-PC |
| SSI78Q8392-28CH | CY7B8392-JC |

| TI | CYPRESS |
|---|---|
| PREFIX: JBP | PREFIX: CY |
| PREFIX: PAL | SUFFIX: P |
| PREFIX: SM | PREFIX: CY |
| PREFIX: SMJ | PREFIX: CY |
| PREFIX: SN | PREFIX: CY |
| PREFIX: TBP | PREFIX: CY |
| PREFIX: TIB | PREFIX: CY |
| PREFIX: TMS | PREFIX: CY |
| SUFFIX: F | SUFFIX: F |
| SUFFIX: J | SUFFIX: L |
| SUFFIX: N | SUFFIX: D |
| SUFFIX: DD | SUFFIX: Z |
| SUFFIX: N | SUFFIX: P |
| 22V10AC | PALC22V10-25C |
| 22V10AM | PALC22V10-30M |
| 27C010-120 | 27C010-120C |
| 27C010-150 | 27C010-150C |
| 27C010-200 | 27C010-200C |
| 27C128-12 | 27C128-120 |
| 27C512-100 | 27C512-90C |
| 27C512-120 | 27C512-120C |
| 27C512-150 | 27C512-150C |
| 27C512-200 | 27C512-200C |
| 27C/PC128-15 | 27C128-150 |
| 27C/PC128-20 | 27C128-200 |
| 27C256-10 | 27C256A-90C |
| 27C256-12 | 27C256A-120C |
| 27C256-15 | 27C256A-150C |
| 27C256-17 | 27C256A-150C |
| 27C256-20 | 27C256A-200C |
| 27PC010-120 | 27C010-120C |
| 27PC010-150 | 27C010-150C |
| 27PC010-200 | 27C010-200C |
| 27PC256-10 | 27C256A-90C |

| TI | CYPRESS |
|---|---|
| 27PC256-12 | 27C256A-120C |
| 27PC256-15 | 27C256A-150C |
| 27PC256-17 | 27C256A-150C |
| 27PC256-20 | 27C256A-200C |
| 27PC512-100 | 27C512-90C |
| 27PC512-120 | 27C512-120C |
| 27PC512-150 | 27C512-150C |
| 27PC512-200 | 27C512-200C |
| 22V10AC | PALC22V10-25C |
| 22V10AM | PALC22V10-30M |
| PAL16R4-25C | PALC16R4-25C |
| PAL16R4-30M | PALC16R4-30M |
| PAL16R4A-2C | PALC16R4-25C |
| PAL16R4A-2M | PALC16R4-40M |
| PAL16R4AC | PALC16R4-25C |
| PAL16R4AM | PALC16R4-30M |
| PAL16R6-20M | PALC16R6-20M |
| PAL16R6-25C | PALC16R6-25C |
| PAL16R6-30M | PALC16R6-30M |
| PAL16R6A-2C | PALC16R6-25C |
| PAL16R6A-2M | PALC16R6-40M |
| PAL16R6AC | PALC16R6-25C |
| PAL16R6AM | PALC16R6-30M |
| PAL16R8-25C | PALC16R8-25C |
| PAL16R8-30M | PALC16R8-30M |
| PAL16R8A-2C | PALC16R8-25C |
| PAL16R8A-2M | PALC16R8-40M |
| PAL16R8AC | PALC16R8-25C |
| PAL16R8AM | PALC16R8-30M |
| PAL20L8A-2C | PLDC20G10-25C |
| PAL20L8A-2M | PLDC20G10-30M |
| PAL20L8AC | PLDC20G10-25C |
| PAL20L8AM | PLDC20G10-30M |
| PAL20L10A-2C | PLDC20G10-25C |
| PAL20L10A-2M | PLDC20G10-30M |
| PAL20L10AC | PLDC20G10-35C |
| PAL20L10AM | PLDC20G10-30M |
| PAL20R4A-2C | PLDC20G10-25C |
| PAL20R4A-2M | PLDC20G10-30M |
| PAL20R4AC | PLDC20G10-25C |
| PAL20R4AM | PLDC20G10-30M |
| PAL20R6A-2C | PLDC20G10-25C |

| TI | CYPRESS |
|---|---|
| PAL20R6A-2M | PLDC20G10-30M |
| PAL20R6AC | PLDC20G10-25C |
| PAL20R6AM | PLDC20G10-30M |
| PAL20R8A-2C | PLDC20G10-25C |
| PAL20R8A-2M | PLDC20G10-30M |
| PAL20R8AC | PLDC20G10-25C |
| PAL20R8AM | PLDC20G10-30M |
| PAL22V10-7C | PALCE22V10-7C |
| PAL22V10-15C | PALC22V10B-15C |
| PAL22V10-20M | PALC22V10B-20M |
| PAL22V10AC | PALC22V10-25C |
| PAL22V10AC | PALC22V10L-25C |
| PAL22V10AM | PALC22V10-25MB |
| PAL22V10AM | PALC22V10-30MB |
| PAL22V10C | PALC22V10-35C |
| PAL22V10C | PALC22V10L-35C |
| PAL20R6AM | PLDC20G10-30M |
| PAL20R8A-2C | PLDC20G10-25C |
| PAL20R8A-2M | PLDC20G10-30M |
| PAL20R8AC | PLDC20G10-25C |
| PAL20R8AM | PLDC20G10-30M |
| PAL22V10-7C | PALCE22V10-7C |
| PAL22V10-15C | PALC22V10B-15C |
| PAL22V10-20M | PALC22V10B-20M |
| PAL22V10AC | PALC22V10-25C |
| PAL22V10AC | PALC22V10L-25C |
| PAL22V10AM | PALC22V10-25MB |
| PAL22V10AM | PALC22V10-30MB |
| PAL22V10C | PALC22V10-35C |
| PAL22V10C | PALC22V10L-35C |
| SN74ACT7201LA15 | 7C421-15 |
| SN74ACT7201LA25 | 7C421-25 |
| SN74ACT7202LA15 | 7C425-15 |
| SN74ACT7202LA25 | 7C425-25 |
| SN74ACT7203L15 | 7C429-15 |

| TI | CYPRESS |
|---|---|
| SN74ACT7203L25 | 7C429-25 |
| SN74ACT7204L15 | 7C433-15 |
| SN74ACT7204L25 | 7C433-25 |
| SN74ACT72211L-RJ | 7C4211-JC |
| SN74ACT72221L-RJ | 7C4221-JC |
| SN74ACT72231L-RJ | 7C4231-JC |
| SN74ACT72241L-RJ | 7C4241-JC |
| SN74ACT7884-FN | 7C4245-JC |
| SN74ACT7882-FN | 7C4235-JC |
| SN74ACT7881-FN | 7C4225-JC |
| SN74ACT7811-FN | 7C4225-JC |

| TOSHIBA | CYPRESS |
|---|---|
| PREFIX: P | SUFFIX: P |
| PREFIX: TC | PREFIX: CY |
| PREFIX: TMM | PREFIX: CY |
| SUFFIX: D | SUFFIX: D |
| 2015A | 7C128A-55C+ |
| 2018-25 | 7C128A-25C |
| 2018-35 | 7C128A-35C |
| 2018-45 | 7C128A-45C |
| 2018-55 | 7C128A-55C+ |
| 2018AP-35 | 7C128A-35C |
| 2018AP-45 | 7C128A-45C |
| 2068-25 | 7C168A-25C |
| 2068-35 | 7C168A-35C |
| 2068-45 | 7C168A-45C |
| 2068-55 | 7C168A-45C |
| 2069-35 | 7C169A-35C |
| 2078-35 | 7C170A-35C |
| 2078-45 | 7C170A-45C |
| 2078-55 | 7C170A-45C |
| 2088-35 | 7C186-35C |
| 315 | 2147-55C |
| 55257-10 | 7C199-55C |
| 55101BFL-xx | 62128-70SC |
| 551001BFTL-xx | 62128-70ZC |
| 55101BTRL-xs | 62128-70RZC |
| 551001BFL-xx | 62128L-70SC |

| TOSHIBA | CYPRESS |
|---|---|
| 551001BFTL-xx | 62128L-70ZC |
| 551001BFL-xxL | 62128LL-70SC |
| 551001BFTL-xxL | 62128LL-70ZC |
| 551001BFL-xxV | 62128VL-70SC |
| 551001BFTL-xxV | 62128VL70ZC |
| 551001BLF-xxV | 62128VLL-70SC |
| 551001BFTL-xxV | 62128VLL-70ZC |
| 55257-70 | 7C199-55C |
| 55257-85 | 7C199-55C |
| 55257CFTL-xx | 62256-70ZC |
| 55257CTRL-xx | 62256-70RZC |
| 55257CFL-xx | 62256-70SC |
| 55257CFL-xx | 62256-70SNC |
| 55257CPL-xx | 62256-70PC |
| 55257CFTL-xxL | 62256L-70ZC |
| 55257CTRL-xxL | 62256L-70RZC |
| 55257CFL-xxL | 62256L-70SNC |
| 55257CFTL-xxL | 62256L-70ZC |
| 55257CTRL-xxL | 62256LL-70RC |
| 55257CFL-xxL | 62256LL-70SNC |
| 55257BFTL-xxLV | 62256-70ZC |
| 55257BTRL-xxLV | 62256-70RZC |
| 55257BFL-xxLV | 62256V-70SC |
| 55257BFL-xxLV | 62256V-70SNC |
| 55257BFTL-xxLV | 62256VL-70ZC |
| 55257BTRL-xxLV | 62256VL-70RZC |
| 55257BFL-xxLV | 62256VL-70SC |
| 55257BFL-xxLV | 62256VL-70SNC |
| 55257BFTL-xxLV | 62256VLL-70ZC |
| 55257BTRL-xxLV | 62256VLL-70RZC |
| 55257BFL-xxLV | 62256VLL-70SNC |
| 55328-15 | 7C199-15C |
| 55328-20 | 7C199-20C |
| 55328-25 | 7C199-25C |
| 55328-35 | 7C199-35C |
| 55328-25 | 7C199-25C |
| 55328-35 | 7C199-35C |
| 55329P/J | 7C188 |
| 55399-20 | 7C188-20C |
| 55399-25 | 7C188-25C |
| 55416-35 | 7C164-35C |
| 55417-25 | 7C166-25C |

| TOSHIBA | CYPRESS |
|---|---|
| 55417-35 | 7C166-35C |
| 55417P/J-15 | 7C166-15C |
| 55417P/J-20 | 7C166-20C |
| 55417P/J-25 | 7C166-25C |
| 55417P/J-35 | 7C166-35C |
| 55464-12 | 7C194-12C |
| 55464-15 | 7C194-15C |
| 5561P/J | 7C187-C |
| 5562 | 7C187-C |
| 55v63 | 7C185-C |
| 55464-20 | 7C194-20C |
| 5588P/J | 7C185-C |
| 5589P/J-25 | 7C182-25C |
| 55B328-12 | 7C199-12C |
| 55B328-15 | 7C199-15C |
| 55B464-12 | 7C194-12C |
| 55B465-12 | 7C196-12C |
| 57C256A-120 | 27C256A-120C |
| 57C256A-150 | 27C256A-150C |
| 55464-25 | 7C194-25C |
| 55464-35 | 7C194-35C |
| 55464-25 | 7C194-25C |
| 55464-35 | 7C194-35C |
| 55465-12 | 7C196-12C |
| 55465-15 | 7C196-15C |
| 55465-20 | 7C196-20C |
| 55465-25 | 7C196-25C |
| 57C256A-200 | 27C256A-200C |
| 55465-25 | 7C195-25C |
| 55465-35 | 7C196-35C |
| 55465-35 | 7C195-35C |

| WSI | CYPRESS |
|---|---|
| PREFIX: W | PREFIX: CY |
| PREFIX: WS | PREFIX: CY |
| SUFFIX: C | SUFFIX: Q |
| SUFFIX: D/T | SUFFIX: W |
| SUFFIX: P/S | SUFFIX: P |
| 57C43C-55C | 7C243/4-55C |
| 57C43C-45C | 7C243/4-45C |
| 57C43C-35C | 7C243/4-35C |

| WSI | CYPRESS |
|---|---|
| 57C43C-25C | 7C243/4-25C |
| 57C45-25 | 7C245A-25C |
| 57C45-25M | 7C245A-25M |
| 57C45-35 | 7C245A-35C |
| 57C45-35M | 7C245A-35M |
| 57C45-45 | 7C245A-45C |
| 57C45-45M | 7C245A-45M |
| 57C49B-35 | 7C263/4-35C |
| 57C49B-45 | 7C263/4-45C |
| 57C49B-45M | 7C263/4-45C |
| 57C49B-55 | 7C263/4-55C |
| 57C49B-55M | 7C263/4-55C |
| 57C49B-70 | 7C263/4-55C |
| 57C49B-70M | 7C263/4-55C |
| 57C49C-25 | 7C263/4-25C |
| 57C49C-35 | 7C263/4-35C |
| 57C49C-45 | 7C263/4-45C |
| 57C49C-45M | 7C263/4-45C |
| 57C49C-55 | 7C263/4-55C |
| 57C49C-55M | 7C263/4-55C |
| 57C49C-70 | 7C263/4-55C |
| 57C49C-70M | 7C263/4-55C |
| 57C64F-55 | 7C266-55C |
| 57C51C-45 | 7C251/4-45C |
| 57C51C-45M | 7C251/4-45M |
| 57C51C-55 | 7C251/4-55C |
| 57C51C-55M | 7C251/4-55M |
| 57C51C-70 | 7C251/4-55M |
| 57C51C-70M | 7C251/4-55M |
| 57C71C-35 | 7C271A-35C |
| 57C71C-45 | 7C271A-45C |
| 57C71C-55 | 7C271A-55C |
| 57C71C-55M | 7C271A-55M |
| 57C71C-70 | 7C271A-55M |
| 57C71C-70M | 7C271A-55M |
| 57C128F-55C | 27C128-55C+ |
| 57C128F-70C | 27C128-70C+ |
| 57C128FB-45 | 27C128-45C |
| 57C128FB-55 | 27C128-55C |
| 57C128FB-70 | 27C128-70C |
| 57C191B-35 | 7C292A-35C |
| 57C191B-35M | 7C292A-35M |
| 57C191B-45 | 7C292A-35C |
| 57C191B-45M | 7C292A-35M |
| 57C191B-50M | 7C292A-50M |
| 57C191B-55 | 7C292A-50C |
| 57C191B-55M | 7C292A-50M |
| 57C191C-25 | 7C292A-25C |
| 57C191C-35 | 7C292A-35C |
| 57C191C-45 | 7C292A-35C |
| 57C191C-45M | 7C292A-35M |
| 57C191C-55 | 7C292A-50C |
| 57C191C-55M | 7C292A-50M |
| 57C256F-35 | 27H256-35C |
| 57C256F-45 | 27C256A-45C |
| 57C256F-55 | 27C256A-55C |
| 57C256F-55M | 27C256-55M |
| 57C256F-70 | 27C256A-70C |
| 57C256F-70M | 27C256A-70M |
| 57C256F-90 | 27C256A-90C |
| 57C291B-35 | 7C291A-35C |
| 57C291B-35M | 7C291A-35M |
| 57C291B-45 | 7C291A-35C |
| 57C291B-45M | 7C291A-35M |
| 57C291B-50M | 7C291A-50M |
| 57C291B-55 | 7C291A-50C |
| 57C291B-55M | 7C291A-50M |
| 57C291C-25 | 7C291A-25C |
| 57C291C-35 | 7C291A-35C |
| 57C291C-45 | 7C291A-35C |
| 57C291C-45M | 7C291A-35M |
| 57C291C-55 | 7C291A-50C |
| 57C291C-55M | 7C291A-50M |

CYPRESS

# Cypress Military Products

For a complete listing of all Cypress Military products, please refer to the Military Product Selector Guide (published twice yearly). This information is also available on the Cypress CD-ROM (published quarterly) and on our website at http://www.cypress.com.

CYPRESS

*GENERAL INFORMATION* 1

**SMALL PLDS** 2

*CPLDS* 3

*DEVELOPMENT TOOLS* 4

*APPLICATION NOTES* 5

*QUALITY* 6

*PACKAGES* 7

**Table of Contents** **Page Number**

## Small PLDs (Programmable Logic Devices)

Introduction to Cypress PLDs .......... 2-1

| Device | Description |
|---|---|
| PAL®C20 Series | Reprogrammable CMOS PALC 16L8, 16R8, 16R6, 16R4 .......... 2-7 |
| PALCE16V8 | Flash Erasable, Reprogrammable CMOS PAL® Device .......... 2-8 |
| PALCE20V8 | Flash Erasable, Reprogrammable CMOS PAL® Device .......... 2-18 |
| PLDC20G10B | CMOS Generic 24-Pin Reprogrammable Logic Device .......... 2-29 |
| PLDC20G10 | CMOS Generic 24-Pin Reprogrammable Logic Device .......... 2-29 |
| PLDC20RA10 | Reprogrammable Asynchronous CMOS Logic Device .......... 2-38 |
| PALCE22V10 | Flash Erasable, Reprogrammable CMOS PAL® Device .......... 2-48 |
| PALC22V10 | Reprogrammable CMOS PAL® Device .......... 2-58 |
| PALC22V10B | Reprogrammable CMOS PAL® Device .......... 2-60 |
| PALC22V10D | Flash Erasable, Reprogrammable CMOS PAL® Device .......... 2-62 |
| CY7C330 | CMOS Programmable Synchronous State Machine .......... 2-72 |
| CY7C331 | Asynchronous Registered EPLD .......... 2-73 |
| CY7C335 | Universal Synchronous EPLD .......... 2-87 |

CYPRESS

# Introduction to Cypress PLDs

## Cypress PLD Family Features

Cypress Semiconductor's PLD family offers the user a wide range of programmable logic solutions that incorporate leading-edge circuit design techniques as well as diverse process technology capabilities. This allows Cypress PLD users to select PLDs that best suit the needs of their particular high-performance system, regardless of whether speed, power consumption, density, or device flexibility are the critical requirements imposed by the system.

Cypress offers enhanced-performance industry-standard 20- and 24-pin device architectures as well as proprietary 28-pin application-tailored architectures. The range of technologies offered includes leading-edge 0.65-micron CMOS EPROM for high speed, low power, and high density, 0.65-micron FLASH technology for high speed, low power and electrical alterability.

The reprogrammable memory cells used by Cypress serve the same purpose as the fuse used in most bipolar PLD devices. Before programming, the AND gates or product terms are connected via the reprogrammable memory cell to both the true and complement inputs. When the reprogrammable memory cell is programmed, the inputs from a gate or product term are disconnected. Programming alters the transistor threshold of each cell so that no conduction can occur, which is equivalent to disconnecting the input from the gate or product term. This is similar to "blowing" the fuses of BiCMOS or bipolar fusible devices, which disconnects the input gate from the product term. Selective programming of each of these reprogrammable memory cells enables the specific logic function to be implemented by the user.

The programmability of Cypress's PLDs allows the users to customize every device in a number of ways to implement their unique logic requirements. Using PLDs in place of SSI or MSI components results in more effective utilization of board space, reduced cost and increased reliability. The flexibility afforded by these PLDs allows the designer to quickly and effectively implement a number of logic functions ranging from random logic gate replacement to complex combinatorial logic functions.

The PLD family implements the familiar "sum of products" logic by using a programmable AND array whose output terms feed a fixed OR array. The sum of these can be expressed in a Boolean transfer function and is limited only by the number of product terms available in the AND-OR array. A variety of different sizes and architectures are available. This allows for more efficient logic optimization by matching input, output, and product terms to the desired application.

## PLD Notation

To reduce confusion and to have an orderly way of representing the complex logic networks, logic diagrams are provided for the various part types. In order to be useful, Cypress logic diagrams employ a common logic convention that is easy to use. *Figure 1* shows the adopted convention. In part (a), an "x" represents an unprogrammed EPROM cell or intact fuse link that is used to perform the logical AND operation upon the input terms. The convention adopted does not imply that the input terms are connected on the common line that is indicated. A further extension of this convention is shown in part (b), which shows the implementation of a simple transfer function. The normal logic representation of the transfer function logic convention is shown in part (c).

## PLD Circuit Configurations

Cypress PLDs have several different output configurations that cover a wide spectrum of applications. The available output configurations offer the user the benefits of both lower package counts and reduced costs when used. This approach allows designers to select PLDs that best fit their applications. An example of some of the configurations that are available are listed below.

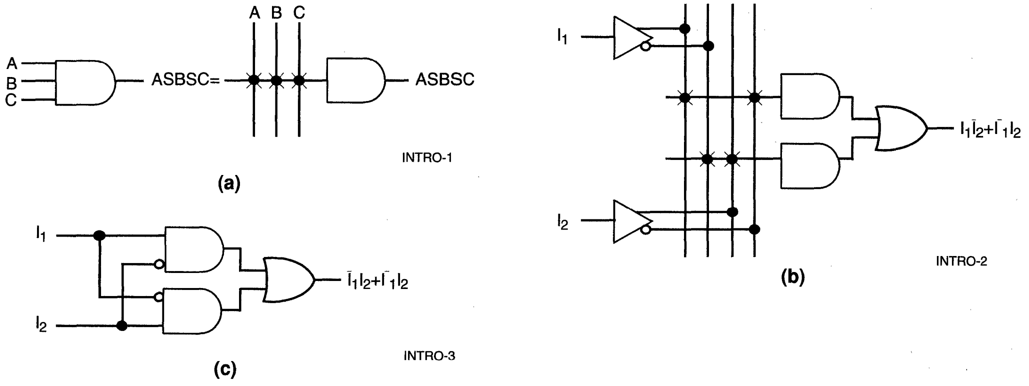


**Figure 1. Logic Diagram Conventions**

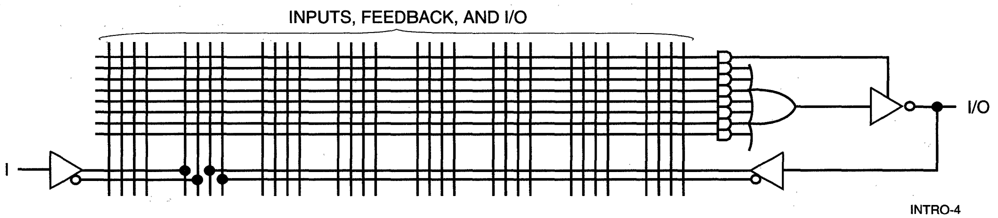


**Figure 2. Programmable I/O**

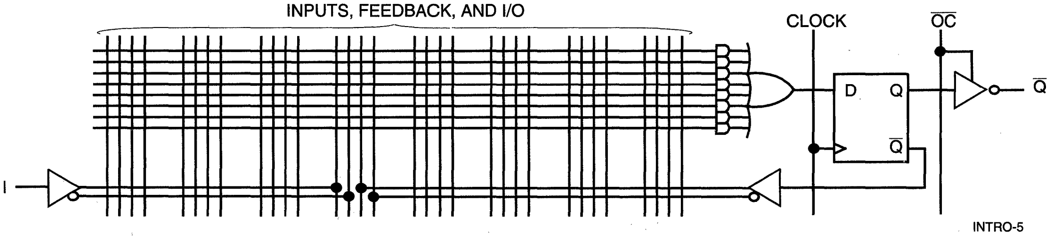


**Figure 3. Registered Outputs with Feedback**

## Programmable I/O

*Figure 2* illustrates the programmable I/O offered in the Cypress PLD family that allows product terms to directly control the outputs of the device. One product term is used to directly control the three-state output buffer, which then gates the summation of the remaining terms to the output pin. The output of this summation can be fed back into the PLD as an input to the array. This programmable I/O feature allows the PLD to drive the output pin when the three-state output is enabled or, when the three-state output is disabled, the I/O pin can be used as an input to the array.

## Registered Outputs with Feedback

*Figure 3* illustrates the registered outputs offered on a number of the Cypress PLDs which allow any of these circuits to function as a state sequencer. The summation of the product terms is stored in the D-type output flip-flop on the rising edge of the system clock. The Q output of the flip-flop can then be gated to the output pin by enabling the three-state output buffer. The output of the flip-flop can also be fed back into the array as an input term. The output feedback feature allows the PLD to remember and then alter its function based upon that state. This circuit can be used to execute such functions as counting, skip, shift, and branch.

## Programmable Macrocell

The programmable macrocell, illustrated in *Figure 4*, provides the capability of defining the architecture of each output individually. Each of the potential outputs may be specified to be "registered" or "combinatorial." Polarity of each output may also be individually selected allowing complete flexibility of output configuration. Further configurability is provided through "array" configurable "output enable" for each potential output. This feature allows the outputs to be reconfigured as inputs on an individual basis or alternately used as a bidirectional I/O controlled by the programmable array (see *Figure 5*).

## Buried Register Feedback

The CY7C331 and CY7C335 PLDs provide registers that may be "buried" or "hidden" by electing feedback of the register output. These buried registers, which are useful in state machines, may be implemented without sacrificing the use of the associated device pin as an input. In previous PLDs, when the feedback path was activated, the input pin-path to the logic array was blocked. The proprietary CY7C335 reprogrammable synchronous state machine macrocell illustrates the shared input multiplexer, which provides an alternative input path for the I/O pin associated with a buried macrocell register (*Figure 6*). Each pair of macrocells shares an input multiplexer, and as long as alternate macrocells are buried, up to six of the twelve output registers can be buried without the loss of any I/O pins as inputs. The CY7C335 also contains four dedicated hidden macrocells with no external output that are used as additional state registers for creating high-performance state machines (*Figure 7*).

## Asynchronous Register Control

Cypress also offers PLDs that may be used in asynchronous systems in which register clock, set, and reset are controlled by the outputs of the product term array. The clock signal is created by the processing of external inputs and/or internal feedback by the logic of the product term array, which is then routed to the register clock. The register set and reset are similarly controlled by product term outputs and can be triggered at any time independent of the register clock in response to external and/or feedback inputs processed by the logic array. The proprietary CY7C331 Asynchronous Registered PLD, for which the I/O macrocell is illustrated in *Figure 8*, is an example of such a device. The register clock, set, and reset functions of the CY7C331 are all controlled by product terms and are dependent only on input signal timing and combinatorial delay through the device logic array to enable their respective functions.

## Input Register Cell

Other Cypress PLDs provide input register cells to capture short duration inputs that would not otherwise be present at the inputs long enough to allow the device to respond. The proprietary CY7C335 Reprogrammable Synchronous State Machine provides these input register cells (*Figure 9*). The clock for the input register may be provided from one of two external clock input pins selectable by a configuration bit, C4, dedicated for this purpose for each input register. This choice of input register clock allows signals to be captured and processed from two independent system sources, each controlled by its own independent clock. These input register cells are provided within I/O macrocells, as well as for dedicated input pins.

Document #: 38-00165-B

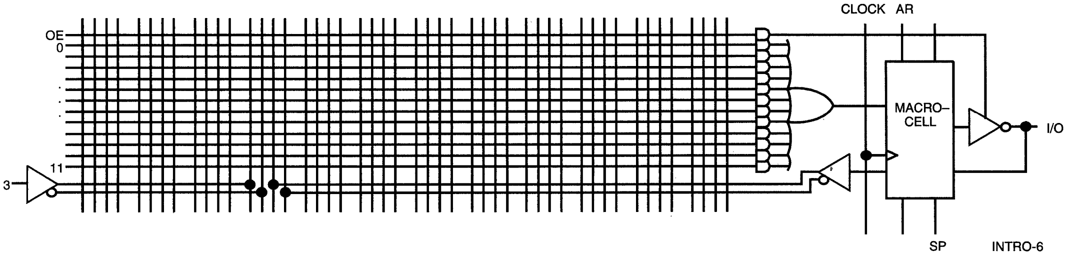


**Figure 4. Programmable Macrocell**

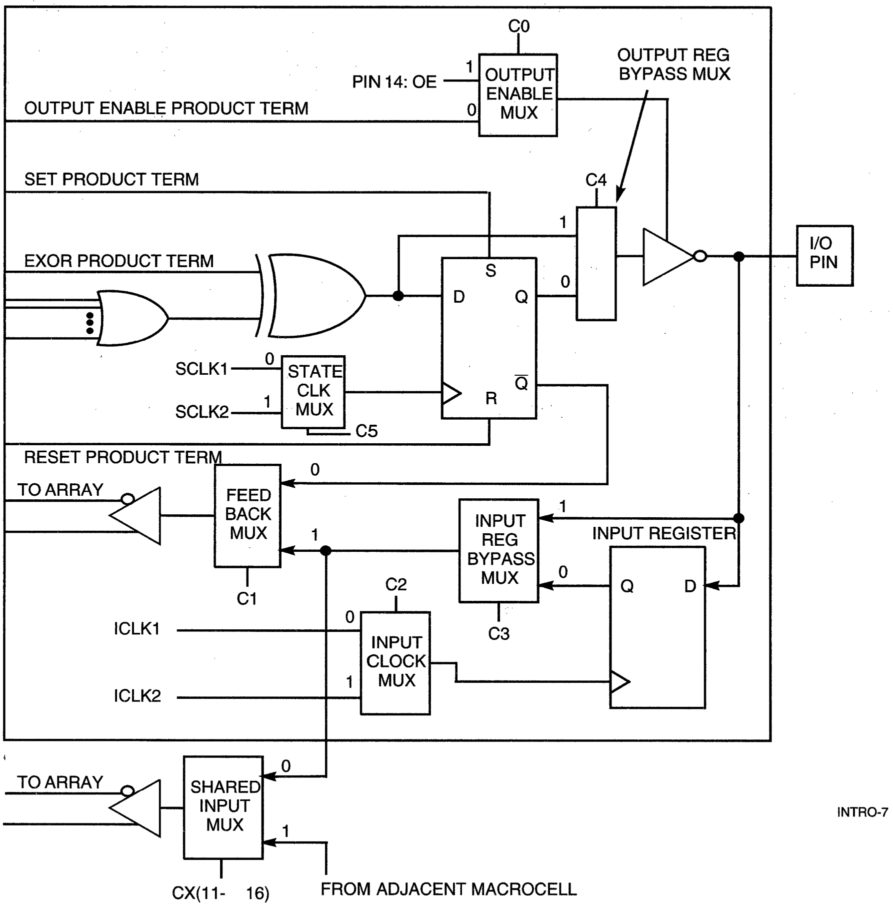
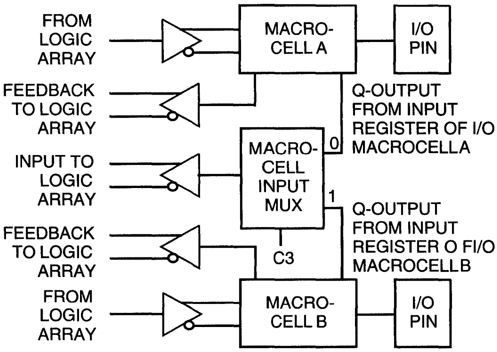
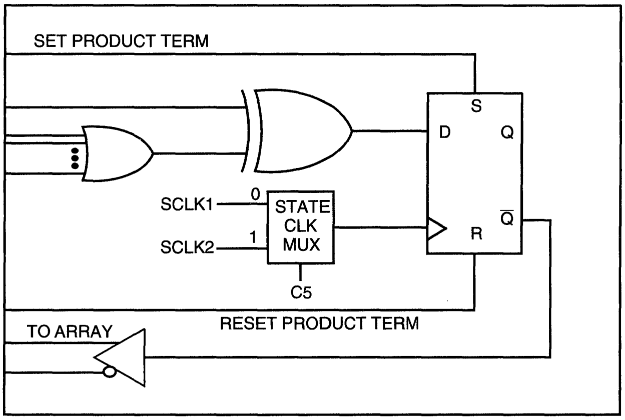


**Figure 5. CY7C335 I/O Macrocell**



INTRO-8

**Figure 6. CY7C335 I/O Macrocell Pair Shared Input MUX**



INTRO-9

**Figure 7. CY7C335 Hidden Macrocell**

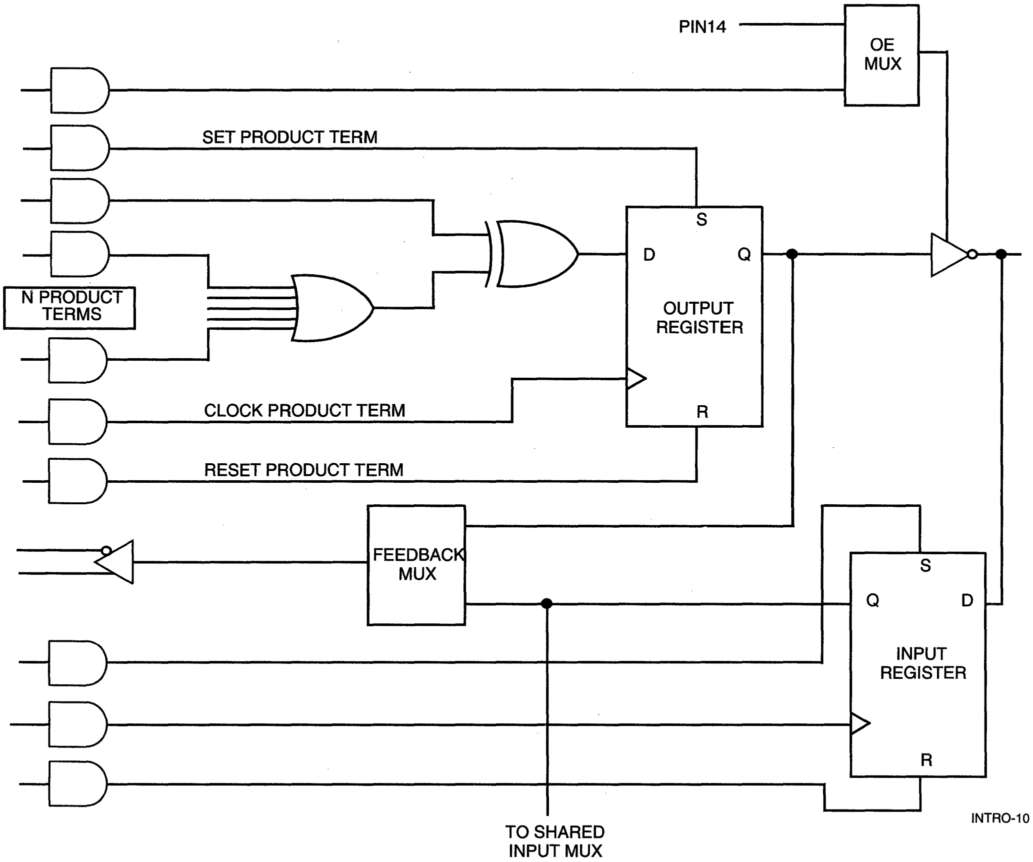


**Figure 8. CY7C331 Registered Asynchronous Macrocell**

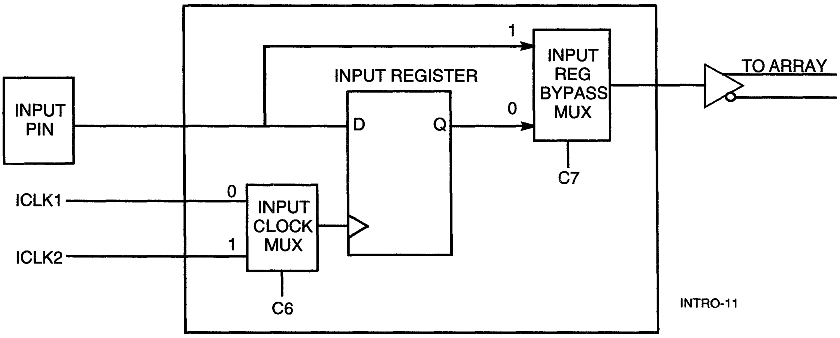


**Figure 9. CY7C335 Input Macrocell**

**This is an abbreviated datasheet. Contact a Cypress representative for complete specifications. For new designs, please refer to the PALCE16V8.**

CYPRESS

# Reprogrammable CMOS PAL®C 16L8, 16R8, 16R6, 16R4

## Features

- CMOS EPROM technology for reprogrammability
- High performance at quarter power
  - — $t_{PD}$ = 25 ns
  - — $t_S$ = 20 ns
  - — $t_{CO}$ = 15 ns
  - — $I_{CC}$ = 45 mA
- High performance at military temperature
  - — $t_{PD}$ = 20 ns
  - — $t_S$ = 20 ns
  - — $t_{CO}$ = 15 ns
  - — $I_{CC}$ = 70 mA
- Commercial and military temperature range
- High reliability
  - — Proven EPROM technology
  - — >1500V input protection from electrostatic discharge
  - — 100% AC and DC tested
  - — 10% power supply tolerances
  - — High noise immunity
  - — Security feature prevents pattern duplication
  - — 100% programming and functional testing

## Functional Description

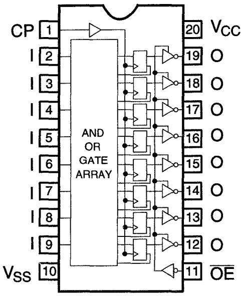
Cypress PALC20 Series devices are high- speed electrically programmable and UV- erasable logic devices produced in a proprietary N-well CMOS EPROM process. These devices utilize a sum-of-products (AND-OR) structure providing users with the ability to program custom logic functions serving unique requirements.

PALs are offered in 20-pin plastic and ceramic DIP, plastic SOJ, and ceramic LCC packages. The ceramic package can be equipped with an erasure window; when exposed to UV light, the PAL is erased and can then be reprogrammed.

Before programming, AND gates or product terms are connected via EPROM cells to both true and complement inputs. Programming an EPROM cell disconnects an input term from a product term. Selective programming of these cells allows a specific logic function to be implemented in a PALC device. PALC devices are supplied in four functional configurations designated 16R8, 16R6, 16R4, and 16L8. These 8 devices have potentially 16 inputs and 8 outputs configurable by the user. Output configurations of 8 registers, 8 combinatorial, 6 registers and 2 combinatorial as well as 4 registers and 4 combinatorial are provided by the 4 functional variations of the product family.

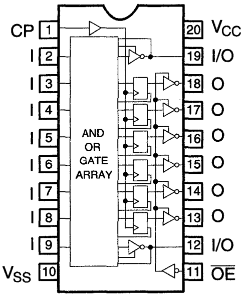
## Logic Symbols and DIP and SOJ Pinouts

16R8



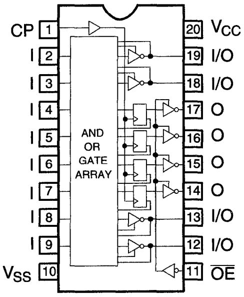
C20–1

16R6



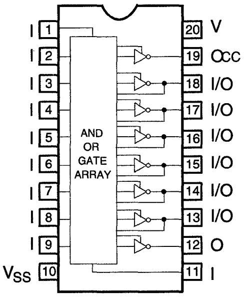
C20–2

16R4



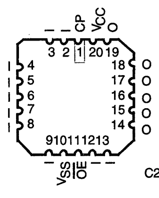
C20–3

16L8

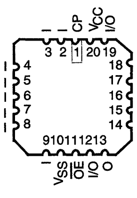


C20–4

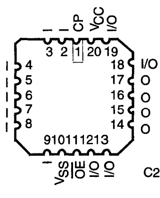
## LCC Pinouts



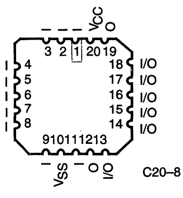
C20–5



C20–6



C20–7



C20–8

PAL is a registered trademark of Advanced Micro Devices.

# PALCE16V8

## Flash Erasable, Reprogrammable CMOS PAL® Device

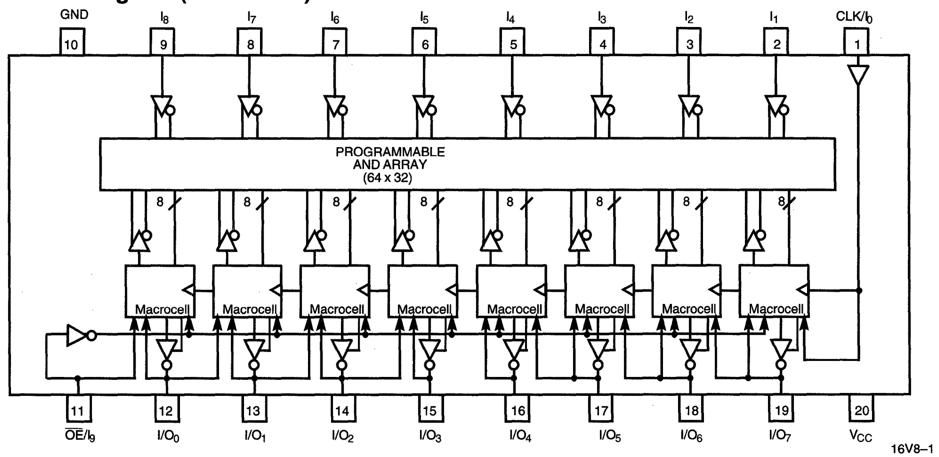
### Features

- Active pull-up on data input pins
- Low power version (16V8L)
  - —55 mA max. commercial (10, 15, 25 ns)
  - —65 mA max. industrial (10, 15, 25 ns)
  - —65 mA military (15 and 25 ns)
- Standard version has low power
  - —90 mA max. commercial (10, 15, 25 ns)
  - —115 mA max. commercial (7 ns)
  - —130 mA max. military/industrial (10, 15, 25 ns)
- CMOS Flash technology for electrical erasability and reprogrammability
- PCI compliant
- User-programmable macrocell
  - —Output polarity control
  - —Individually selectable for registered or combinatorial operation
- Up to 16 input terms and 8 outputs
- QSOP packaging available
  - —7.5 ns com'l version
    5 ns $t_{CO}$
    5 ns $t_S$
    7.5 ns $t_{PD}$
    125-MHz state machine
  - —10 ns military/industrial versions
    7 ns $t_{CO}$
    10 ns $t_S$
    10 ns $t_{PD}$
    62-MHz state machine
- High reliability
  - —Proven Flash technology
  - —100% programming and functional testing

### Functional Description

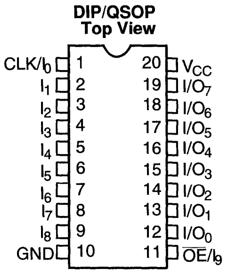
The Cypress PALCE16V8 is a CMOS Flash Electrical Erasable second-generation programmable array logic device. It is implemented with the familiar sum-of-product (AND-OR) logic structure and the programmable macrocell.

### Logic Block Diagram (PDIP/CDIP)

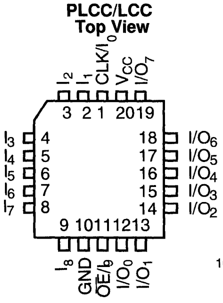


PAL is a registered trademark of Advanced Micro Devices.

## Pin Configuration



16V8–2



16V8–3

## Selection Guide

| Generic Part Number | $t_{PD}$ ns |  | $t_S$ ns |  | $t_{CO}$ ns |  | $I_{CC}$ mA |  |
|---|---|---|---|---|---|---|---|---|
|  | Com'l/Ind | Mil | Com'l/Ind | Mil | Com'l/Ind | Mil | Com'l | Mil/Ind |
| PALCE16V8-5 | 5 |  | 3 |  | 4 |  | 115 |  |
| PALCE16V8-7 | 7.5 |  | 7 |  | 5 |  | 115 |  |
| PALCE16V8-10 | 10 | 10 | 10 | 10 | 7 | 10 | 90 | 130 |
| PALCE16V8-15 | 15 | 15 | 12 | 12 | 10 | 10 | 90 | 130 |
| PALCE16V8-25 | 25 | 25 | 15 | 20 | 12 | 12 | 90 | 130 |
| PALCE16V8L-15 | 15 | 15 | 12 | 12 | 10 | 12 | 55 | 65 |
| PALCE16V8L-25 | 25 | 25 | 15 | 20 | 12 | 20 | 55 | 65 |

Shaded area contains preliminary information.

## Functional Description (continued)

The PALCE16V8 is executed in a 20-pin 300-mil molded DIP, a 300-mil cerdip, a 20-lead square ceramic leadless chip carrier, a 20-lead square plastic leaded chip carrier and a 20-lead, quarter-size outline. The device provides up to 16 inputs and 8 outputs. The PALCE16V8 can be electrically erased and reprogrammed. The programmable macrocell enables the device to function as a superset to the familiar 20-pin PLDs such as 16L8, 16R8, 16R6, and 16R4.

The PALCE16V8 features 8 product terms per output and 32 input terms into the AND array. The first product term in a macrocell can be used either as an internal output enable control or as a data product term.

There are a total of 18 architecture bits in the PALCE16V8 macrocell; two are global bits that apply to all macrocells and 16 that apply locally, two bits per macrocell. The architecture bits determine whether the macrocell functions as a register or combinatorial with inverting or noninverting output. The output enable control can come from an external pin or internally from a product term. The output can also be permanently enabled, functioning as a dedicated output or permanently disabled, functioning as a dedicated input. Feedback paths are selectable from either the input/output pin associated with the macrocell, the input/output pin associated with an adjacent pin, or from the macrocell register itself.

### Power-Up Reset

All registers in the PALCE16V8 power-up to a logic LOW for predictable system initialization. For each register, the associated output pin will be HIGH due to active-LOW outputs.

### Electronic Signature

An electronic signature word is provided in the PALCE16V8 that consists of 64 bits of programmable memory that can contain user-defined data.

### Security Bit

A security bit is provided that defeats the readback of the internal programmed pattern when the bit is programmed.

### Low Power

The Cypress PALCE16V8 provides low-power operation through the use of CMOS technology, and increased testability with Flash reprogrammability.

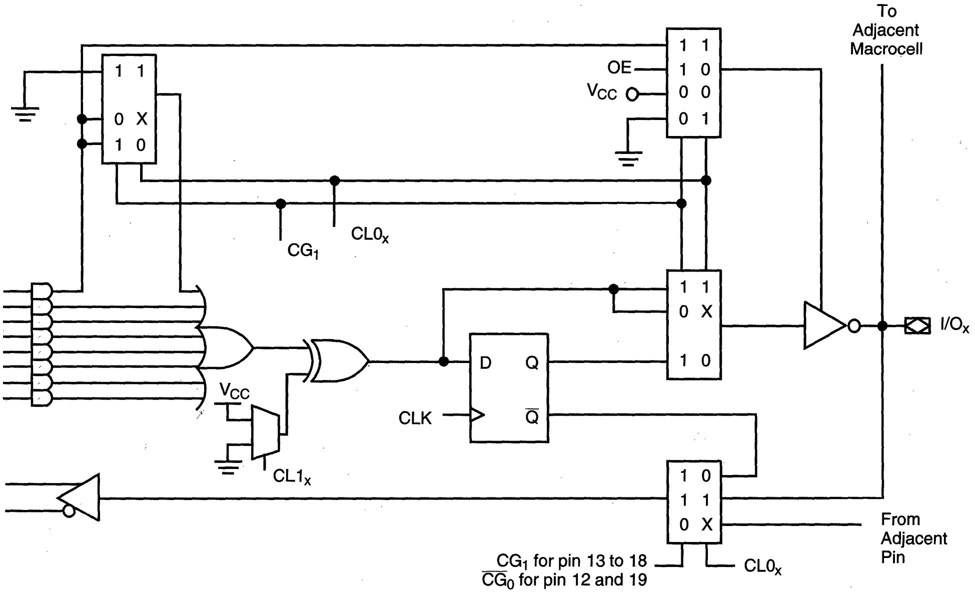
### Product Term Disable

Product Term Disable (PTD) fuses are included for each product term. The PTD fuses allow each product term to be individually disabled.

## Configuration Table

| $CG_0$ | $CG_1$ | $CL0_x$ | Cell Configuration | Devices Emulated |
|---|---|---|---|---|
| 0 | 1 | 0 | Registered Output | Registered Med PALs |
| 0 | 1 | 1 | Combinatorial I/O | Registered Med PALs |
| 1 | 0 | 0 | Combinatorial Output | Small PALs |
| 1 | 0 | 1 | Input | Small PALs |
| 1 | 1 | 1 | Combinatorial I/O | 16L8 only |

## Macrocell



16V8–4

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | −65°C to +150°C |
| Ambient Temperature with Power Applied | −55°C to +125°C |
| Supply Voltage to Ground Potential (Pin 24 to Pin 12) | −0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | −0.5V to +7.0V |
| DC Input Voltage | −0.5V to +7.0V |
| Output Current into Outputs (LOW) | 24 mA |
| DC Programming Voltage | 12.5V |
| Latch-Up Current | >200 mA |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ±5% |
| Military[1] | −55°C to +125°C | 5V ±10% |
| Industrial | −40°C to +85°C | 5V ±10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | Com'l | 2.4 | | V |
| | | | $I_{OH}$ = −2 mA | Mil/Ind | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 24 mA | Com'l | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | Mil/Ind | | | |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[3] | | | 2.0 | | V |
| $V_{IL}$[4] | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[3] | | | −0.5 | 0.8 | V |
| $I_{IH}$ | Input or I/O HIGH Leakage Current | 3.5V ≤ $V_{IN}$ ≤ $V_{CC}$ | | | | 10 | μA |
| $I_{IL}$[5] | Input or I/O LOW Leakage Current | 0V ≤ $V_{IN}$ ≤ $V_{IN}$ (Max.) | | | | −100 | μA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[6, 7] | | | −30 | −150 | mA |
| $I_{CC}$ | Operating Power Supply Current | | 5, 7 ns | Com'l | | 115 | mA |
| | | | 10, 15, 25 ns | | | 90 | mA |
| | | | 15L, 25L ns | | | 55 | mA |
| | | | 10, 15, 25 ns | Mil/Ind | | 130 | mA |
| | | | 15L, 25L ns | Mil. | | 65 | mA |
| | | | 15L, 25L ns | Ind. | | 65 | mA |

## Capacitance[7]

| Parameter | Description | Test Conditions | Typ. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0V @ f = 1 MHz | 5 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V @ f = 1 MHz | 5 | pF |

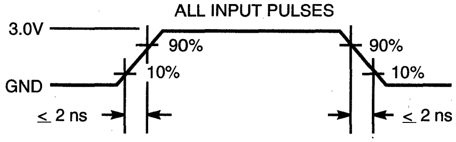
## Endurance Characteristics[7]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | | Cycles |

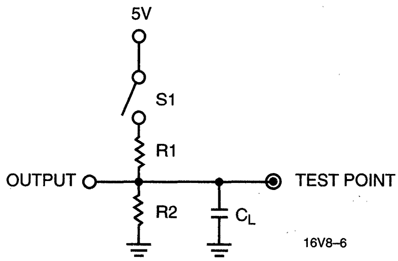
**Notes:**

1. $T_A$ is the "instant on" case temperature.
2. See the last page of this specification for Group A subgroup testing information.
3. These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.
4. $V_{IL}$ (Min.) is equal to -3.0V for pulse durations less than 20 ns.
5. The leakage current is due to the internal pull-up resistor on all pins.
6. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
7. Tested initially and after any design or process changes that may affect these parameters.

## AC Test Loads and Waveforms



16V8–5



16V8–6

| Specification | $S_1$ | $C_L$ | Commercial | | Military | | Measured Output Value |
|---|---|---|---|---|---|---|---|
| | | | $R_1$ | $R_2$ | $R_1$ | $R_2$ | |
| $t_{PD}$, $t_{CO}$ | Closed | 50 pF | 200Ω | 390Ω | 390Ω | 750Ω | 1.5V |
| $t_{PZX}$, $t_{EA}$ | Z ➧ H: Open<br>Z ➧ L: Closed | | | | | | 1.5V |
| $t_{PXZ}$, $t_{ER}$ | H ➧ Z: Open<br>L ➧ Z: Closed | 5 pF | | | | | H ➧ Z: $V_{OH}$ - 0.5V<br>L ➧ Z: $V_{OL}$ + 0.5V |

## Commercial and Industrial Switching Characteristics[2]

| Parameter | Description | 16V8-5 | | 16V8-7 | | 16V8-10 | | 16V8-15 | | 16V8-25 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8, 9] | 1 | 5 | 3 | 7.5 | 3 | 10 | 3 | 15 | 3 | 25 | ns |
| $t_{PZX}$ | $\overline{\text{OE}}$ to Output Enable | 1 | 6 | | 6 | | 10 | | 15 | | 20 | ns |
| $t_{PXZ}$ | $\overline{\text{OE}}$ to Output Disable | 1 | 5 | | 6 | | 10 | | 15 | | 20 | ns |
| $t_{EA}$ | Input to Output Enable Delay[7] | 1 | 6 | | 9 | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[7, 10] | 1 | 5 | | 9 | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8,9] | 1 | 4 | 2 | 5 | 2 | 7 | 2 | 10 | 2 | 12 | ns |
| $t_S$ | Input or Feedback Set-Up Time | 3 | | 5 | | 7.5 | | 12 | | 15 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 7 | | 10 | | 14.5 | | 22 | | 27 | | ns |
| $t_{WH}$ | Clock Width HIGH[7] | 3 | | 4 | | 6 | | 8 | | 12 | | ns |
| $t_{WL}$ | Clock Width LOW[7] | 3 | | 4 | | 6 | | 8 | | 12 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$)[7, 11] | 143 | | 100 | | 69 | | 45.5 | | 37 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[7, 12] | 166 | | 125 | | 83 | | 62.5 | | 41.6 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[7, 13] | 166 | | 125 | | 74 | | 50 | | 40 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[7, 14] | | 3 | | 3 | | 6 | | 8 | | 10 | ns |
| $t_{PR}$ | Power-Up Reset Time[7] | 1 | | 1 | | 1 | | 1 | | 1 | | µs |

Shaded area contains preliminary information.

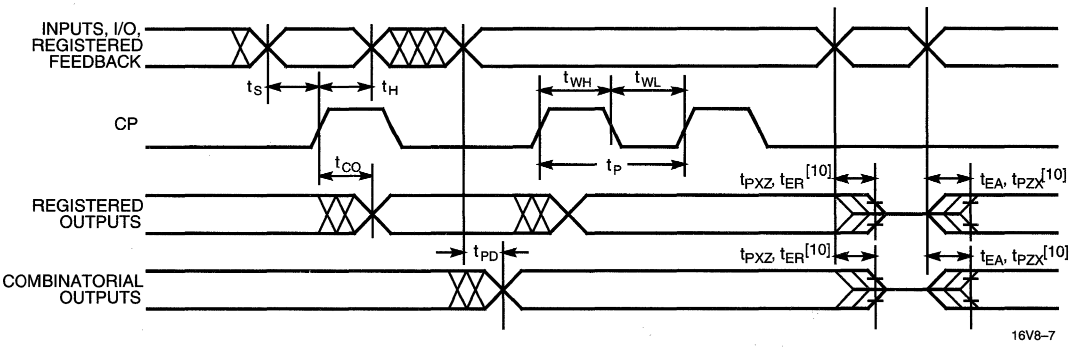
**Notes:**

8. Min. times are tested initially and after any design or process changes that may affect these parameters.
9. This specification is guaranteed for all device outputs changing state in a given access cycle.
10. This parameter is measured as the time after $\overline{\text{OE}}$ pin or internal disable input disables or enables the output pin. This delay is measured to the point at which a previous HIGH level has fallen to 0.5 volts below $V_{OH}$ min. or a previous LOW level has risen to 0.5 volts above $V_{OL}$ max.
11. This specification indicates the guaranteed maximum frequency at which a state machine configuration with external feedback can operate.
12. This specification indicates the guaranteed maximum frequency at which the device can operate in data path mode.
13. This specification indicates the guaranteed maximum frequency at which a state machine configuration with internal only feedback can operate.
14. This parameter is calculated from the clock period at $f_{MAX}$ internal ($1/f_{MAX3}$) as measured (see Note 7 above) minus $t_S$.

## Military Switching Characteristics[7]

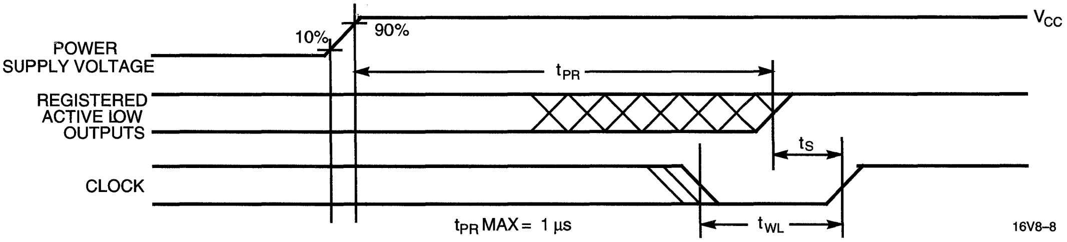
| Parameter | Description | 16V8-10 | | 16V8-15 | | 16V8-25 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8, 9] | 3 | 10 | 3 | 15 | 3 | 25 | ns |
| $t_{PZX}$ | $\overline{OE}$ to Output Enable | | 10 | | 15 | | 20 | ns |
| $t_{PXZ}$ | $\overline{OE}$ to Output Disable | | 10 | | 15 | | 20 | ns |
| $t_{EA}$ | Input to Output Enable Delay[7] | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[7, 10] | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8, 9] | 2 | 7 | 2 | 10 | 2 | 12 | ns |
| $t_S$ | Input or Feedback Set-Up Time | 10 | | 12 | | 15 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 17 | | 22 | | 27 | | ns |
| $t_{WH}$ | Clock Width HIGH[7] | 6 | | 8 | | 12 | | ns |
| $t_{WL}$ | Clock Width LOW[7] | 6 | | 8 | | 12 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$[7, 11] | 58 | | 45.5 | | 37 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[7, 12] | 83 | | 62.5 | | 41.6 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[7, 13] | 62.5 | | 50 | | 40 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[7, 14] | | 6 | | 8 | | 10 | ns |
| $t_{PR}$ | Power-Up Reset Time[7] | 1 | | 1 | | 1 | | µs |

## Switching Waveform



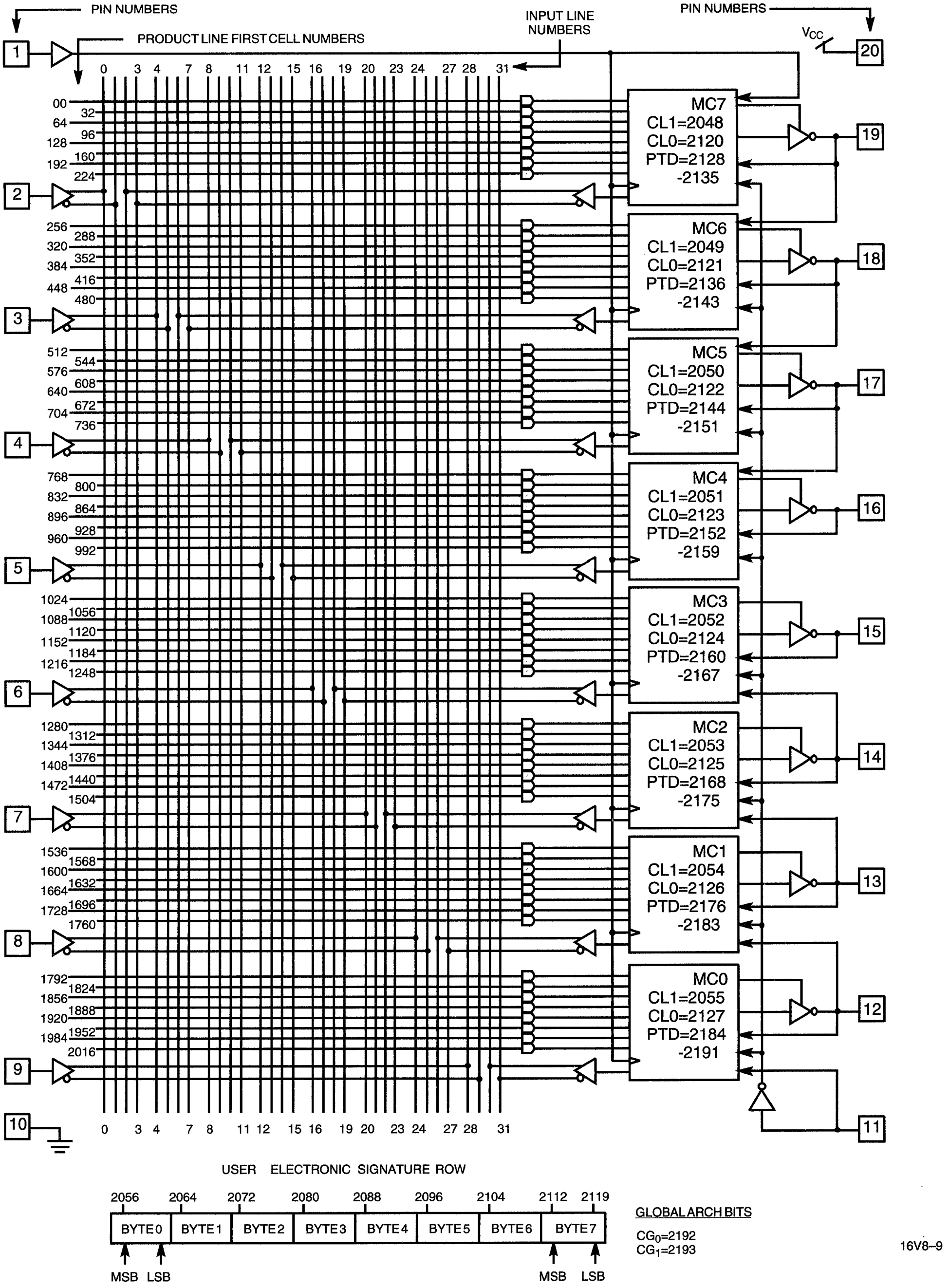
16V8–7

## Power-Up Reset Waveform



16V8–8

# Functional Logic Diagram for PALCE16V8



16V8–9

## Ordering Information

| $I_{CC}$ (mA) | $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 115 | 5 | 3 | 4 | PALCE16V8-5JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| 115 | 7.5 | 5 | 5 | PALCE16V8-7JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE16V8-7PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| 90 | 10 | 7.5 | 7 | PALCE16V8-10QC | Q5 | 20-Lead Quarter-Size Outline | |
| | | | | PALCE16V8-10JC | J61 | 20-Lead Plastic Leaded Chip Carrier | |
| | | | | PALCE16V8-10PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| 130 | 10 | 7.5 | 7 | PALCE16V8-10JI | J61 | 20-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE16V8-10PI | P5 | 20-Lead (300-Mil) Molded DIP | |
| 130 | 10 | 10 | 7 | PALCE16V8-10DMB | D6 | 20-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE16V8-10LMB | L61 | 20-Pin Square Leadless Chip Carrier | |
| 90 | 15 | 12 | 10 | PALCE16V8-15JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE16V8-15PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| 130 | 15 | 12 | 10 | PALCE16V8-15JI | J61 | 20-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE16V8-15PI | P5 | 20-Lead (300-Mil) Molded DIP | |
| | | | | PALCE16V8-15DMB | D6 | 20-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE16V8-15LMB | L61 | 20-Pin Square Leadless Chip Carrier | |
| 90 | 25 | 15 | 12 | PALCE16V8-25JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE16V8-25PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| 130 | 25 | 15 | 12 | PALCE16V8-25JI | J61 | 20-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE16V8-25PI | P5 | 20-Lead (300-Mil) Molded DIP | |
| | | | | PALCE16V8-25DMB | D6 | 20-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE16V8-25LMB | L61 | 20-Pin Square Leadless Chip Carrier | |
| 55 | 10 | 7.5 | 7 | PALCE16V8L-10JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE16V8L-10PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| 65 | 10 | 10 | 7 | PALCE16V8L-10JI | J61 | 20-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE16V8L-10PI | P5 | 20-Lead (300-Mil) Molded DIP | |
| 55 | 15 | 12 | 10 | PALCE16V8L-15JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE16V8L-15PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| | | | | PALCE16V8L-15QC | Q5 | 20-Lead Quarter-Size Outline | |
| 65 | 15 | 12 | 10 | PALCE16V8L-15JI | J61 | 20-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE16V8L-15PI | P5 | 20-Lead (300-Mil) Molded DIP | |
| | | | | PALCE16V8L-15QI | Q5 | 20-Lead Quarter-Size Outline | |
| 65 | 15 | 12 | 10 | PALCE16V8L-15DMB | D6 | 20-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE16V8L-15LMB | L61 | 20-Pin Square Leadless Chip Carrier | |
| 55 | 25 | 15 | 12 | PALCE16V8L-25JC | J61 | 20-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE16V8L-25PC | P5 | 20-Lead (300-Mil) Molded DIP | |
| | | | | PALCE16V8L-25QC | Q5 | 20-Lead Quarter-Size Outline | |
| 65 | 25 | 15 | 12 | PALCE16V8L-25JI | J61 | 20-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE16V8L-25PI | P5 | 20-Lead (300-Mil) Molded DIP | |
| | | | | PALCE16V8L-25QI | Q5 | 20-Lead Quarter-Size Outline | |
| | | | | PALCE16V8L-25DMB | D6 | 20-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE16V8L-25LMB | L61 | 20-Pin Square Leadless Chip Carrier | |

Shaded area contains preliminary information.

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing
## DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

## Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{S}$ | 9, 10, 11 |
| $t_{H}$ | 9, 10, 11 |

Document #: 38-00364-D

CYPRESS

# PALCE20V8

## Flash Erasable, Reprogrammable CMOS PAL® Device

### Features

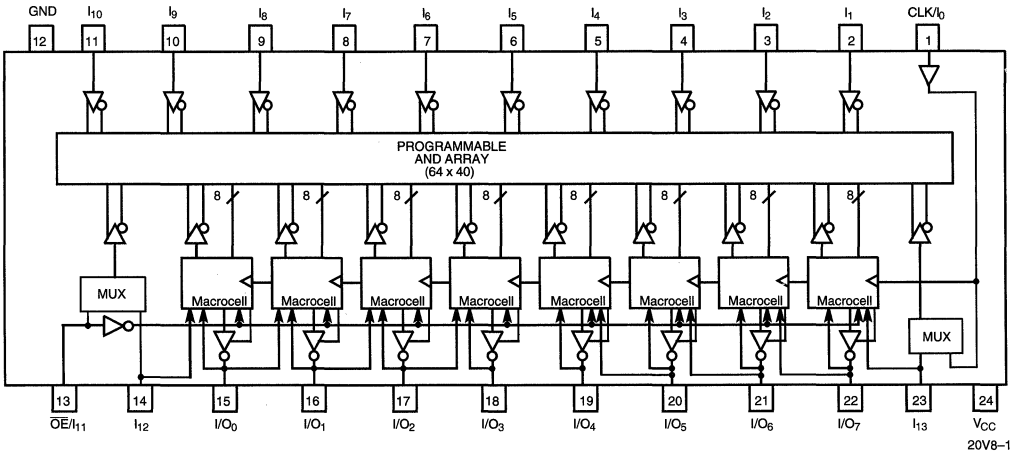
- Active pull-up on data input pins
- Low power version (20V8L)
  - —55 mA max. commercial (15, 25 ns)
  - —65 mA max. military/industrial (15, 25 ns)
- Standard version has low power
  - —90 mA max. commercial (15, 25 ns)
  - —115 mA max. commercial (10 ns)
  - —130 mA max. military/industrial (15, 25 ns)
- CMOS Flash technology for electrical erasability and reprogrammability
- User-programmable macrocell
  - —Output polarity control
  - —Individually selectable for registered or combinatorial operation
- QSOP package available
  - —10, 15, and 25 ns com'l version
  - —15, and 25 ns military/industrial versions
- High reliability
  - —Proven Flash technology
  - —100% programming and functional testing

### Functional Description

The Cypress PALCE20V8 is a CMOS Flash Erasable second-generation programmable array logic device. It is implemented with the familiar sum-of-product (AND-OR) logic structure and the programmable macrocell.

The PALCE20V8 is executed in a 24-pin 300-mil molded DIP, a 300-mil cerdip, a 28-lead square ceramic leadless chip carrier, a 28-lead square plastic leaded chip carrier, and a 24-lead quarter size outline. The device provides up to 20 inputs and 8 outputs. The PALCE20V8 can be electrically erased and reprogrammed. The programmable macrocell enables the device to function as a superset to the familiar 24-pin PLDs such as 20L8, 20R8, 20R6, 20R4.

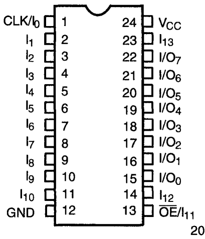
### Logic Block Diagram (PDIP/CDIP/QSOP)



PAL is a registered trademark of Advanced Micro Devices, Inc.

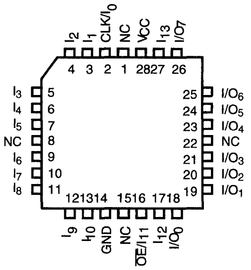
## Pin Configuration

DIP/QSOP
Top View



20V8-2

PLCC/LCC
Top View



20V8-3

## Selection Guide

| Generic Part Number | $t_{PD}$ ns Com'l/Ind | $t_{PD}$ ns Mil | $t_S$ ns Com'l/Ind | $t_S$ ns Mil | $t_{CO}$ ns Com'l/Ind | $t_{CO}$ ns Mil | $I_{CC}$ mA Com'l | $I_{CC}$ mA Mil/Ind |
|---|---|---|---|---|---|---|---|---|
| PALCE20V8–5 | 5 | | 3 | | 4 | | 115 | |
| PALCE20V8–7 | 7.5 | | 7 | | 5 | | 115 | |
| PALCE20V8–10 | 10 | 10 | 10 | 10 | 7 | 10 | 115 | 130 |
| PALCE20V8–15 | 15 | 15 | 12 | 12 | 10 | 12 | 90 | 130 |
| PALCE20V8–25 | 25 | 25 | 15 | 20 | 12 | 20 | 90 | 130 |
| PALCE20V8L–15 | 15 | 15 | 12 | 12 | 10 | 12 | 55 | 65 |
| PALCE20V8L–25 | 25 | 25 | 15 | 20 | 12 | 20 | 55 | 65 |

Shaded area contains preliminary information.

## Functional Description (continued)

The PALCE20V8 features 8 product terms per output and 40 input terms into the AND array. The first product term in a macrocell can be used either as an internal output enable control or as a data product term.

There are a total of 18 architecture bits in the PALCE20V8 macrocell; two are global bits that apply to all macrocells and 16 that apply locally, two bits per macrocell. The architecture bits determine whether the macrocell functions as a register or combinatorial with inverting or noninverting output. The output enable control can come from an external pin or internally from a product term. The output can also be permanently enabled, functioning as a dedicated output or permanently disabled, functioning as a dedicated input. Feedback paths are selectable from either the input/output pin associated with the macrocell, the input/output pin associated with an adjacent pin, or from the macrocell register itself.

### Power-Up Reset

All registers in the PALCE20V8 power-up to a logic LOW for predictable system initialization. For each register, the associated output pin will be HIGH due to active-LOW outputs.

### Electronic Signature

An electronic signature word is provided in the PALCE20V8 that consists of 64 bits of programmable memory that can contain user-defined data.

### Security Bit

A security bit is provided that defeats the readback of the internal programmed pattern when the bit is programmed.

### Low Power

The Cypress PALCE20V8 provides low-power operation through the use of CMOS technology, and increased testability with Flash reprogrammability.

### Product Term Disable

Product Term Disable (PTD) fuses are included for each product term. The PTD fuses allow each product term to be individually disabled.

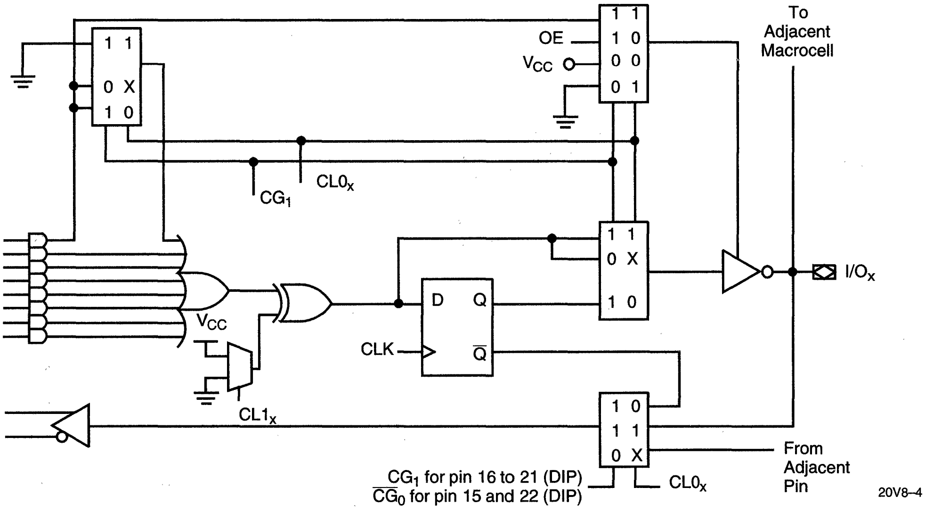
### Input and I/O Pin Pull-Ups

The PALCE20V8 input and I/O pins have built-in active pull-ups that will float unused inputs and I/Os to an active HIGH state (logical 1). All unused inputs and three-stated I/O pins should be connected to another active input, $V_{CC}$, or Ground to improve noise immunity and reduce $I_{CC}$.

## Configuration Table

| $CG_0$ | $CG_1$ | $CL0_x$ | Cell Configuration | Devices Emulated |
|---|---|---|---|---|
| 0 | 1 | 0 | Registered Output | Registered Med PALs |
| 0 | 1 | 1 | Combinatorial I/O | Registered Med PALs |
| 1 | 0 | 0 | Combinatorial Output | Small PALs |
| 1 | 0 | 1 | Input | Small PALs |
| 1 | 1 | 1 | Combinatorial I/O | 20L8 only |

## Macrocell



## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ....................................–65°C to +150°C

Ambient Temperature with Power Applied ................................................–55°C to +125°C

Supply Voltage to Ground Potential (Pin 24 to Pin 12)................................................–0.5V to +7.0V

DC Voltage Applied to Outputs in High Z State ....................................................–0.5V to +7.0V

DC Input Voltage ................................................–0.5V to +7.0V

Output Current into Outputs (LOW)..............................24 mA

DC Programming Voltage ..............................................12.5V

Latch-Up Current .....................................................>200 mA

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ±5% |
| Industrial | –40°C to +85°C | 5V ±10% |
| Military[1] | –55°C to +125°C | 5V ±10% |

**Note:**

[1] $T_A$ is the “instant on” case temperature.

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.,<br>$V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OH}$ = –3.2 mA | Com'l | 2.4 | | V |
| | | | $I_{OH}$ = –2 mA | Mil/Ind | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.,<br>$V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 24 mA | Com'l | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | Mil/Ind | | | |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[3] | | | 2.0 | | V |
| $V_{IL}$[4] | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[3] | | | –0.5 | 0.8 | V |
| $I_{IH}$ | Input or I/O HIGH Leakage Current | 3.5V ≤ $V_{IN}$ ≤ $V_{CC}$ | | | | 10 | μA |
| $I_{IL}$[5] | Input or I/O LOW Leakage Current | 0V ≤ $V_{IN}$ ≤ $V_{IN}$ (Max.) | | | | –100 | μA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[6,7] | | | –30 | –150 | mA |
| $I_{CC}$ | Operating Power Supply Current | $V_{CC}$ = Max.,<br>$V_{IL}$ = 0V, $V_{IH}$ = 3V,<br>Output Open,<br>f = 15 MHz<br>(counter) | 5, 7, 10 ns | Com'l | | 115 | mA |
| | | | 15, 25 ns | | | 90 | mA |
| | | | 15L, 25L ns | | | 55 | mA |
| | | | 10, 15, 25 ns | Mil/Ind | | 130 | mA |
| | | | 15L, 25L ns | Mil/Ind | | 65 | mA |

## Capacitance[7]

| Parameter | Description | Test Conditions | Typ. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0V @ f = 1 MHz | 5 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V @ f = 1 MHz | 5 | pF |

## Endurance Characteristics[7]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | | Cycles |

**Notes:**

[2] See the last page of this specification for Group A subgroup testing information.

[3] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

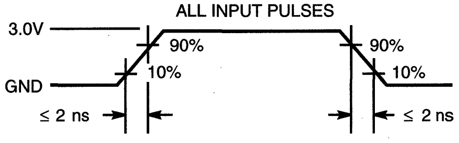
[4] $V_{IL}$ (Min.) is equal to –3.0V for pulse durations less than 20 ns.

[5] The leakage current is due to the internal pull-up resistor on all pins.

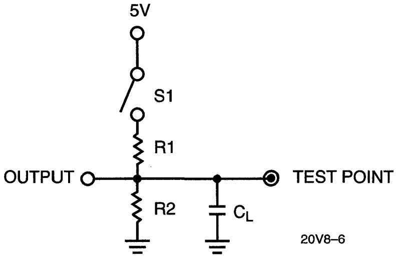
[6] Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[7] Tested initially and after any design or process changes that may affect these parameters.

## AC Test Loads and Waveforms



20V8–5



20V8–6

| Specification | $S_1$ | $C_L$ | Commercial | | Military | | Measured Output Value |
|---|---|---|---|---|---|---|---|
| | | | $R_1$ | $R_2$ | $R_1$ | $R_2$ | |
| $t_{PD}$, $t_{CO}$ | Closed | 50 pF | 200Ω | 390Ω | 390Ω | 750Ω | 1.5V |
| $t_{PZX}$, $t_{EA}$ | Z ▶ H: Open<br>Z ▶ L: Closed | | | | | | 1.5V |
| $t_{PXZ}$, $t_{ER}$ | H ▶ Z: Open<br>L ▶ Z: Closed | 5 pF | | | | | H ▶ Z: $V_{OH} - 0.5V$<br>L ▶ Z: $V_{OL} + 0.5V$ |

## Commercial and Industrial Switching Characteristics[2]

| Parameter | Description | 20V8-5 | | 20V8-7 | | 20V8-10 | | 20V8-15 | | 20V8-25 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8] | 1 | 5 | 1 | 7.5 | 1 | 10 | 1 | 15 | 1 | 25 | ns |
| $t_{PZX}$ | $\overline{OE}$ to Output Enable | | 5 | | 6 | | 10 | | 15 | | 20 | ns |
| $t_{PXZ}$ | $\overline{OE}$ to Output Disable | | 5 | | 6 | | 10 | | 15 | | 20 | ns |
| $t_{EA}$ | Input to Output Enable Delay[7] | | 6 | | 9 | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[7,9] | | 6 | | 9 | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8] | 1 | 4 | 1 | 5 | 1 | 7 | 1 | 10 | 1 | 12 | ns |
| $t_S$ | Input or Feedback Set-Up Time | 3 | | 7 | | 10 | | 12 | | 15 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 7 | | 12 | | 17 | | 22 | | 27 | | ns |
| $t_{WH}$ | Clock Width HIGH[7] | 3 | | 5 | | 8 | | 8 | | 12 | | ns |
| $t_{WL}$ | Clock Width LOW[7] | 3 | | 5 | | 8 | | 8 | | 12 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$)[7,10] | 143 | | 83 | | 58 | | 45.5 | | 37 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[7, 11] | 166.6 | | 100 | | 62.5 | | 62.5 | | 41.6 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[7,12] | 166.6 | | 100 | | 62.5 | | 50 | | 40 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[7, 13] | | 3 | | 3 | | 6 | | 8 | | 10 | ns |
| $t_{PR}$ | Power-Up Reset Time[7] | 1 | | 1 | | 1 | | 1 | | 1 | | µs |

Shaded area contains preliminary information.

**Notes:**

[8] Min. times are tested initially and after any design or process changes that may affect these parameters.

[9] This parameter is measured as the time after $\overline{OE}$ pin or internal disable input disables or enables the output pin. This delay is measured to the point at which a previous HIGH level has fallen to 0.5 volts below $V_{OH}$ min. or a previous LOW level has risen to 0.5 volts above $V_{OL}$ max.

[10] This specification indicates the guaranteed maximum frequency at which a state machine configuration with external feedback can operate.

[11] This specification indicates the guaranteed maximum frequency at which the device can operate in data path mode.

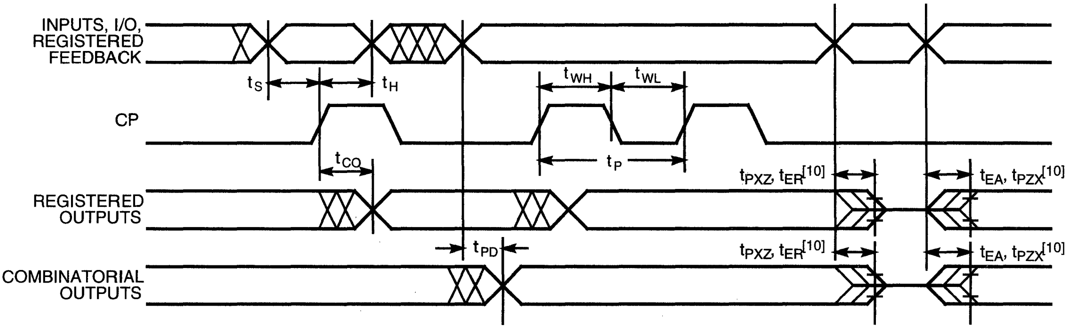
[12] This specification indicates the guaranteed maximum frequency at which a state machine configuration with internal only feedback can operate.

[13] This parameter is calculated from the clock period at $f_{MAX}$ internal ($1/f_{MAX3}$) as measured (see Note 7 above) minus $t_S$.

## Military Switching Characteristics[2]

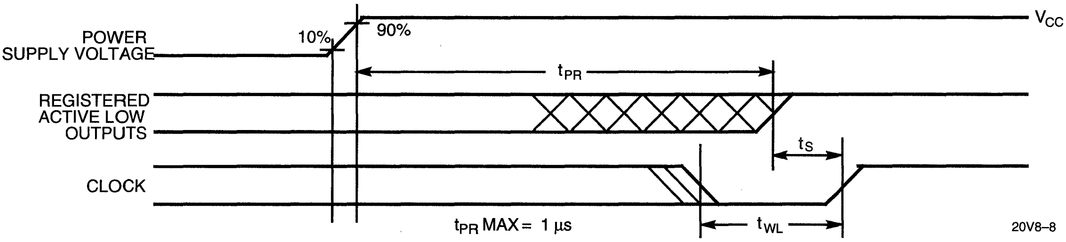
| Parameter | Description | 20V8–10 | | 20V8–15 | | 20V8–25 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8] | 1 | 10 | 1 | 15 | 1 | 25 | ns |
| $t_{PZX}$ | $\overline{OE}$ to Output Enable | | 10 | | 15 | | 20 | ns |
| $t_{PXZ}$ | $\overline{OE}$ to Output Disable | | 10 | | 15 | | 20 | ns |
| $t_{EA}$ | Input to Output Enable Delay[7] | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[7,9] | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8] | 1 | 10 | 1 | 12 | 1 | 20 | ns |
| $t_S$ | Input or Feedback Set-Up Time | 10 | | 12 | | 20 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 20 | | 24 | | 40 | | ns |
| $t_{WH}$ | Clock Width HIGH[7] | 8 | | 10 | | 15 | | ns |
| $t_{WL}$ | Clock Width LOW[7] | 8 | | 10 | | 15 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$)[7,10] | 50 | | 41.7 | | 25 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[7, 11] | 62.5 | | 50 | | 33.3 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[7,12] | 62.5 | | 50 | | 33.3 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[7, 13] | | 6 | | 8 | | 10 | ns |
| $t_{PR}$ | Power-Up Reset Time[7] | 1 | | 1 | | 1 | | µs |

## Switching Waveform



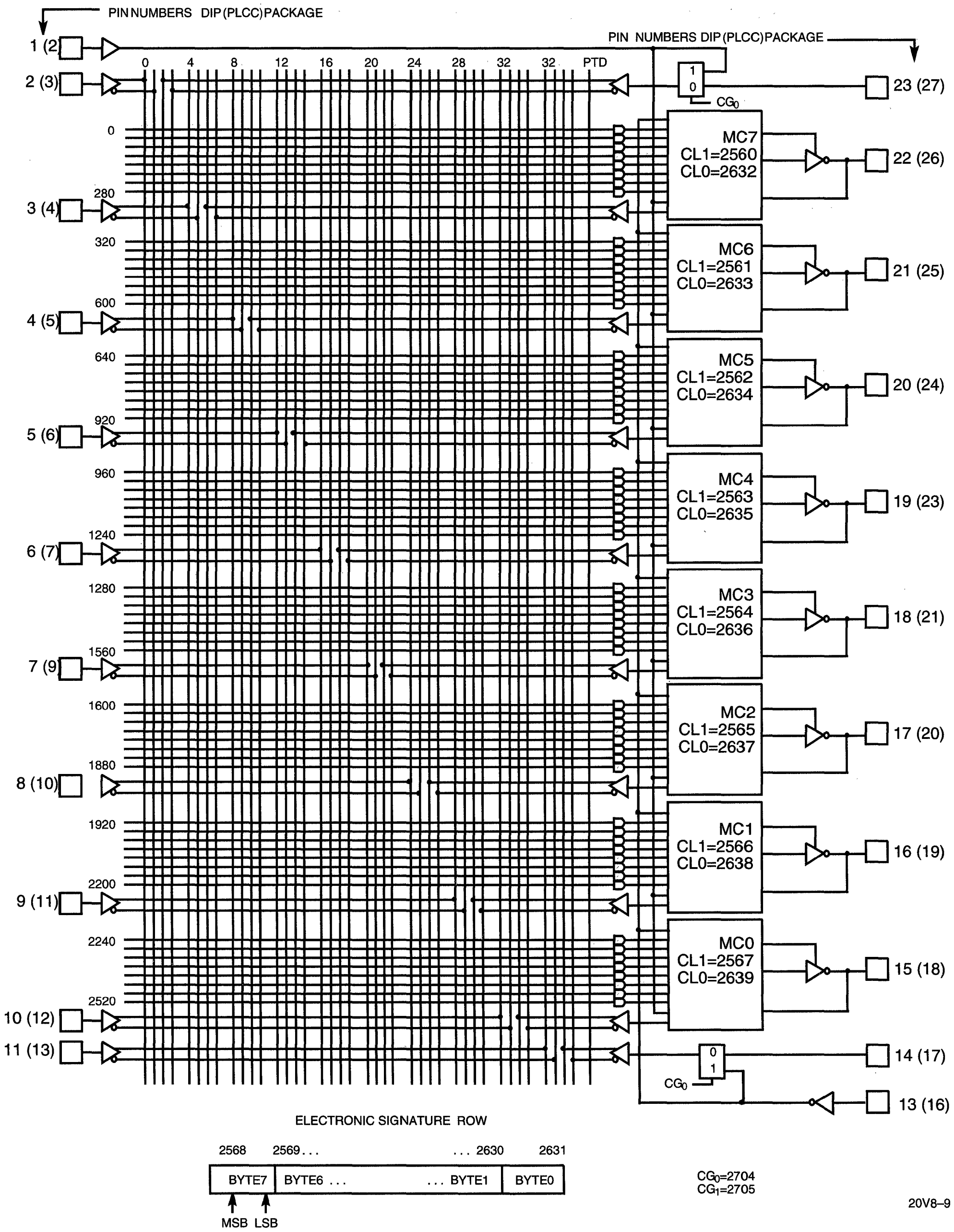
20V8–7

## Power-Up Reset Waveform



20V8–8

## Functional Logic Diagram for PALCE20V8



## Ordering Information for PALCE20V8

| $I_{CC}$ (mA) | $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 115 | 5 | 3 | 4 | PALCE20V8–5JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| 115 | 7.5 | 7 | 5 | PALCE20V8–7JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE20V8–7PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 115 | 10 | 10 | 7 | PALCE20V8–10JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | | | | PALCE20V8–10PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8–10QC | Q13 | 24-Lead Quarter-Size Outline | |
| 130 | 10 | 10 | 10 | PALCE20V8–10JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE20V8–10PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8–10DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE20V8–10LMB | L64 | 28-Pin Square Leadless Chip Carrier | |
| 90 | 15 | 12 | 10 | PALCE20V8–15JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE20V8–15PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8–15QC | Q13 | 24-Lead Quarter-Size Outline | |
| 130 | 15 | 12 | 12 | PALCE20V8–15JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE20V8–15PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8–15QI | Q13 | 24-Lead Quarter-Size Outline | |
| | | | | PALCE20V8–15DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE20V8–15LMB | L64 | 28-Pin Square Leadless Chip Carrier | |
| 90 | 25 | 15 | 12 | PALCE20V8–25JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE20V8–25PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8–25QC | Q13 | 24-Lead Quarter-Size Outline | |
| 130 | 25 | 20 | 20 | PALCE20V8–25JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE20V8–25PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8–25QI | Q13 | 24-Lead Quarter-Size Outline | |
| | | | | PALCE20V8–25DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE20V8–25LMB | L64 | 28-Pin Square Leadless Chip Carrier | |

Shaded area contains preliminary information.

## Ordering Information for PALCE20V8L

| $I_{CC}$ (mA) | $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 55 | 15 | 12 | 10 | PALCE20V8L–15JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE20V8L–15PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8L–15QC | Q13 | 24-Lead Quarter-Size Outline | |
| 65 | 15 | 12 | 12 | PALCE20V8L–15JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE20V8L–15PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8L–15QI | Q13 | 24-Lead Quarter-Size Outline | |
| | | | | PALCE20V8L–15DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE20V8L–15LMB | L64 | 28-Pin Square Leadless Chip Carrier | |
| 55 | 25 | 15 | 12 | PALCE20V8L–25JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE20V8L–25PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8L–25QC | Q13 | 24-Lead Quarter-Size Outline | |
| 65 | 25 | 20 | 20 | PALCE20V8L–25JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE20V8L–25PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | PALCE20V8L–25QI | Q13 | 24-Lead Quarter-Size Outline | |
| | | | | PALCE20V8L–25DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE20V8L–25LMB | L64 | 28-Pin Square Leadless Chip Carrier | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_S$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |

Document #: 38–00367–D

CYPRESS

# PLDC20G10B/PLDC20G10

## CMOS Generic 24-Pin Reprogrammable Logic Device

### Features

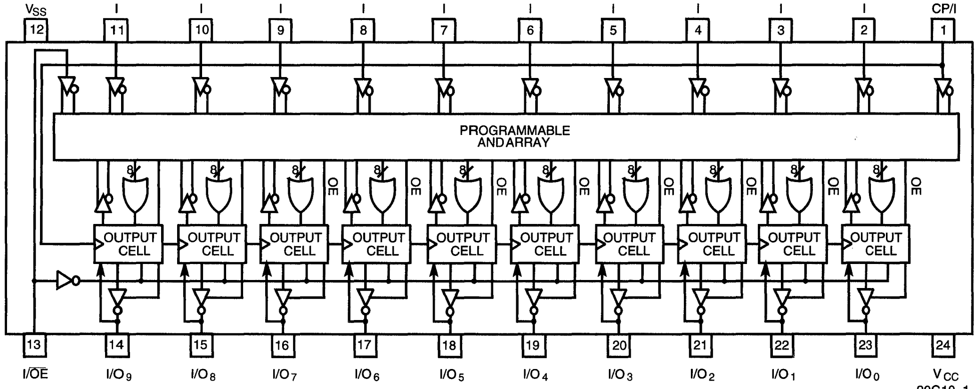
- Fast
  - —Commercial: $t_{PD}$ = 15 ns, $t_{CO}$ = 10 ns, $t_S$ = 12 ns
  - —Military: $t_{PD}$ = 20 ns, $t_{CO}$ = 15 ns, $t_S$ = 15 ns
- Low power
  - —$I_{CC}$ max.: 70 mA, commercial
  - —$I_{CC}$ max.: 100 mA, military
- Commercial and military temperature range
- User-programmable output cells
  - —Selectable for registered or combinatorial operation
  - —Output polarity control
  - —Output enable source selectable from pin 13 or product term
- Generic architecture to replace standard logic functions including: 20L10, 20L8, 20R8, 20R6, 20R4, 12L10, 14L8, 16L6, 18L4, 20L2, and 20V8
- Eight product terms and one OE product term per output
- CMOS EPROM technology for reprogrammability
- Highly reliable
  - —Uses proven EPROM technology
  - —Fully AC and DC tested
  - —Security feature prevents logic pattern duplication
  - —±10% power supply voltage and higher noise immunity

### Functional Description

Cypress PLD devices are high-speed electrically programmable logic devices. These devices utilize the sum-of-products (AND-OR) structure providing users the ability to program custom logic functions for unique requirements.

In an unprogrammed state the AND gates are connected via EPROM cells to both the true and complement of every input. By selectively programming the EPROM cells, AND gates may be connected to either the true or complement or disconnected from both true and complement inputs.

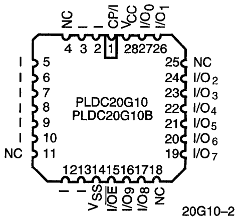
### Logic Block Diagram



20G10–1

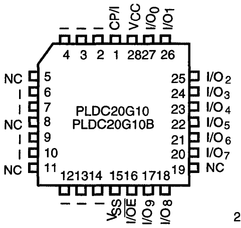
### Pin Configurations

LCC
Top View



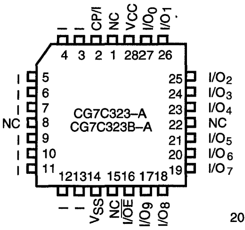
20G10–2

STD PLCC
Top View



20G10–4

JEDEC PLCC [1]
Top View



20G10–3

Note:

[1] The CG7C323 is the PLDC20G10 packaged in the JEDEC-compatible 28-pin PLCC pinout. Pin function and pin order is identical for both PLCC pinouts. The difference is in the location of the "no connect" or NC pins.

## Selection Guide

| Generic Part Number | $I_{CC}$ (mA) |  | $t_{PD}$ (ns) |  | $t_S$ (ns) |  | $t_{CO}$ (ns) |  |
|---|---|---|---|---|---|---|---|---|
|  | Com/Ind | Mil | Com/Ind | Mil | Com/Ind | Mil | Com/Ind | Mil |
| 20G10B–15 | 70 |  | 15 |  | 12 |  | 10 |  |
| 20G10B–20 | 70 | 100 | 20 | 20 | 12 | 15 | 12 | 15 |
| 20G10B–25 |  | 100 |  | 25 |  | 18 |  | 15 |
| 20G10–25 | 55 |  | 25 |  | 15 |  | 15 |  |
| 20G10–30 |  | 80 |  | 30 |  | 20 |  | 20 |
| 20G10–35 | 55 |  | 35 |  | 30 |  | 25 |  |
| 20G10–40 |  | 80 |  | 40 |  | 35 |  | 25 |

## Functional Description (continued)

Cypress PLDC20G10 uses an advanced 0.8-micron CMOS technology and a proven EPROM cell as the programmable element. This technology and the inherent advantage of being able to program and erase each cell enhances the reliability and testability of the circuit. This reduces the burden on the customer to test and to handle rejects.

A preload function allows the registered outputs to be preset to any pattern during testing. Preload is important for testing the functionality of the Cypress PLD device.

## 20G10 Functional Description

The PLDC20G10 is a generic 24-pin device that can be programmed to logic functions that include but are not limited to: 20L10, 20L8, 20R8, 20R6, 20R4, 12L10, 14L8, 16L6, 18L4, 20L2, and 20V8. Thus, the PLDC20G10 provides significant design, inventory and programming flexibility over dedicated 24-pin devices. It is executed in a 24-pin 300-mil molded DIP and a 300-mil windowed cerDIP. It provides up to 22 inputs and 10 outputs. When the windowed cerDIP is exposed to UV light, the 20G10 is erased and then can be reprogrammed.

The programmable output cell provides the capability of defining the architecture of each output individually. Each of the 10 output cells may be configured with registered or combinatorial outputs, active HIGH or active LOW outputs, and product term or Pin 13 generated output enables. Three architecture bits determine the configurations as shown in the Configuration Table and in *Figures 1* through *8*. A total of eight different configurations are possible, with the two most common shown in *Figure 3* and *Figure 5*. The default or unprogrammed state is registered/active/LOW/Pin 11 OE. The entire programmable output cell is shown in the next section.

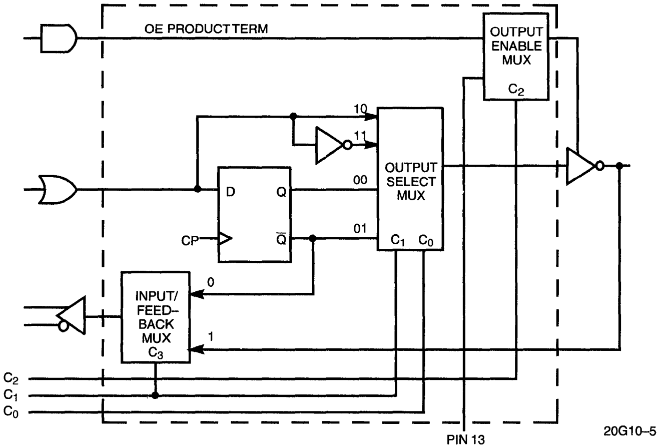
The architecture bit 'C1' controls the registered/combinatorial option. In either combinatorial or registered configuration, the output can serve as an I/O pin, or if the output is disabled, as an input only. Any unused inputs should be tied to ground. In either registered or combinatorial configuration, the output of the register is fed back to the array. This allows the creation of control-state machines by providing the next state. The register is clocked by the signal from Pin 1. The register is initialized on power up to $\overline{Q}$ output LOW and $\overline{Q}$ output HIGH.

In both the combinatorial and registered configurations, the source of the output enable signal can be individually chosen with architecture bit 'C2'. The OE signal may be generated within the array, or from the external $\overline{OE}$ (Pin 13). The Pin 13 allows direct control of the outputs, hence having faster enable/disable times.

Each output cell can be configured for output polarity. The output can be either active HIGH or active LOW. This option is controlled by architecture bit 'C0'.

Along with this increase in functional density, the Cypress PLDC20G10 provides lower-power operation through the use of CMOS technology and increased testability with a register preload feature.

## Programmable Output Cell



20G10–5

## Configuration Table

| Figure | $C_2$ | $C_1$ | $C_0$ | Configuration |
|---|---|---|---|---|
| *1* | 0 | 0 | 0 | Product Term OE/Registered/Active LOW |
| *2* | 0 | 0 | 1 | Product Term OE/Registered/Active HIGH |
| *5* | 0 | 1 | 0 | Product Term OE/Combinatorial/Active LOW |
| *6* | 0 | 1 | 1 | Product Term OE/Combinatorial/Active HIGH |
| *3* | 1 | 0 | 0 | Pin 13 OE/Registered/Active LOW |
| *4* | 1 | 0 | 1 | Pin 13 OE/Registered/Active HIGH |
| *7* | 1 | 1 | 0 | Pin 13 OE/Combinatorial/Active LOW |
| *8* | 1 | 1 | 1 | Pin 13 OE/Combinatorial/Active HIGH |

## Registered Output Configurations

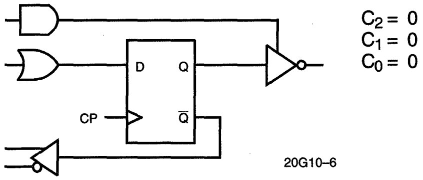


**Figure 1. Product Term OE/Active LOW**

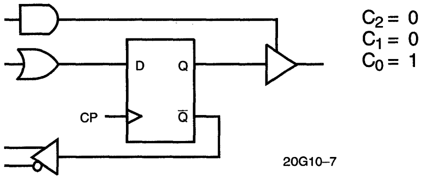


**Figure 2. Product Term OE/Active HIGH**

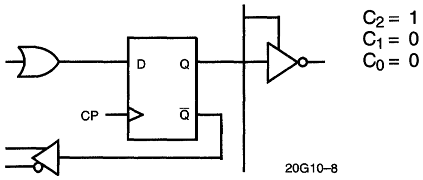


**Figure 3. Pin 13 OE/Active LOW**

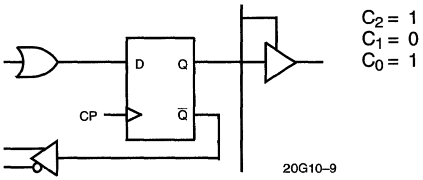


**Figure 4. Pin 13 OE/Active HIGH**

## Combinatorial Output Configurations[2]

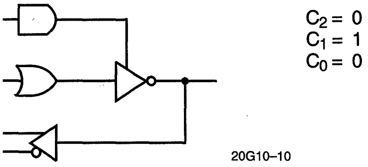


**Figure 5. Product Term OE/Active LOW**

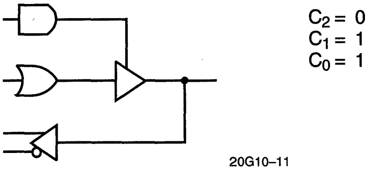


**Figure 6. Product Term OE/Active HIGH**

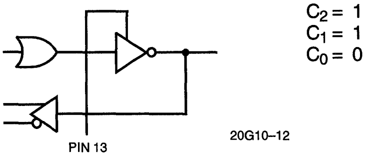


**Figure 7. Pin 13 OE/Active Low**

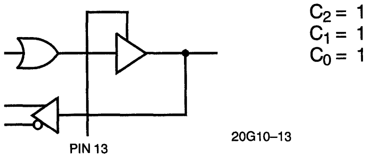


**Figure 8. Pin 13 OE/Active HIGH**

**Note:**

2. Bidirectional I/O configurations are possible only when the combinatorial output option is selected

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | −65°C to +150°C |
| Ambient Temperature with Power Applied | −55°C to +125°C |
| Supply Voltage to Ground Potential | −0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | −0.5V to +7.0V |
| DC Input Voltage | −3.0V to +7.0V |
| Output Current into Outputs (LOW) | 16 mA |
| DC Programming Voltage | |
| PLDC20G10B and CG7C323B–A | 13.0V |
| PLDC20G10 and CG7C323–A | 14.0V |
| Latch-Up Current | >200 mA |
| Static Discharge Voltage (per MIL-STD-883, Method 8015) | >500V |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ±10% |
| Military[3] | −55°C to +125°C | 5V ±10% |
| Industrial | −40°C to +85°C | 5V ±10% |

## Electrical Characteristics Over the Operating Range (Unless Otherwise Noted)[4]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.,<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | Com'l/Ind | 2.4 | | V |
| | | | $I_{OH}$ = −2 mA | Military | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.,<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 24 mA | Com'l/Ind | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | Military | | | |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[5] | | | 2.0 | | V |
| $V_{IL}$ | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[5] | | | | 0.8 | V |
| $I_{IX}$ | Input Leakage Current | $V_{SS} \le V_{IN} \le V_{CC}$ | | | −10 | +10 | μA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[6, 7] | | | | −90 | mA |
| $I_{CC}$ | Power Supply Current | $0 \le V_{IN} \le V_{CC}$<br>$V_{CC}$ = Max.,<br>$I_{OUT}$ = 0 mA<br>Unprogrammed Device | Com'l/Ind–15, –20 | | | 70 | mA |
| | | | Com'l/Ind–25, –35 | | | 55 | mA |
| | | | Military–20, –25 | | | 100 | mA |
| | | | Military–30, –40 | | | 80 | mA |
| $I_{OZ}$ | Output Leakage Current | $V_{CC}$ = Max., $V_{SS} \le V_{OUT} \le V_{CC}$ | | | −100 | 100 | μA |

## Capacitance[7]

| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $T_A$ = 25°C, f = 1 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{IN}$ = 2.0V, $V_{CC}$ = 5.0V | 10 | pF |

**Notes:**

[3] $T_A$ is the "instant on" case temperature.

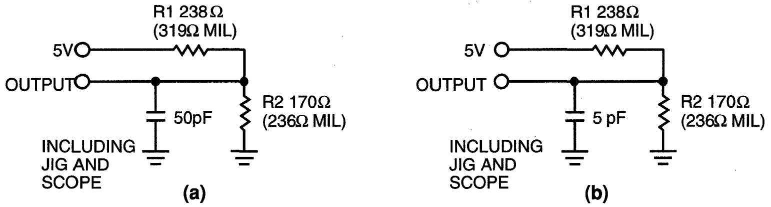
[4] See the last page of this specification for Group A subgroup testing information.

[5] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[6] Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

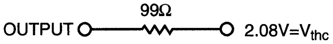
[7] Tested initially and after any design or process changes that may affect these parameters.

## AC Test Loads and Waveforms (Commercial)



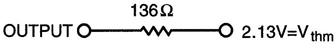
20G10–14

Equivalent to: THÉVENIN EQUIVALENT (Commercial)



20G10–15

Equivalent to: THÉVENIN EQUIVALENT (Military/Industrial)



20G10–16

## Switching Characteristics Over Operating Range[3, 8, 9]

| Parameter | Description | Commercial | | | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | B–15 | | B–20 | | –25 | | –35 | | |
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input or Feedback to Non-Registered Output | | 15 | | 20 | | 25 | | 35 | ns |
| $t_{EA}$ | Input to Output Enable | | 15 | | 20 | | 25 | | 35 | ns |
| $t_{ER}$ | Input to Output Disable | | 15 | | 20 | | 25 | | 35 | ns |
| $t_{PZX}$ | Pin 11 to Output Enable | | 12 | | 15 | | 20 | | 25 | ns |
| $t_{PXZ}$ | Pin 11 to Output Disable | | 12 | | 15 | | 20 | | 25 | ns |
| $t_{CO}$ | Clock to Output | | 10 | | 12 | | 15 | | 25 | ns |
| $t_S$ | Input or Feedback Set-Up Time | 12 | | 12 | | 15 | | 30 | | ns |
| $t_H$ | Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$[10] | Clock Period | 22 | | 24 | | 30 | | 55 | | ns |
| $t_{WH}$ | Clock High Time | 8 | | 10 | | 12 | | 17 | | ns |
| $t_{WL}$ | Clock Low Time | 8 | | 10 | | 12 | | 17 | | ns |
| $f_{MAX}$[11] | Maximum Frequency | 45.4 | | 41.6 | | 33.3 | | 18.1 | | MHz |

**Notes:**

[8] Part (a) of AC Test Loads and Waveforms used for all parameters except $t_{ER}$, $t_{PZX}$, and $t_{PXZ}$. Part (b) of AC Test Loads and Waveforms used for $t_{ER}$, $t_{PZX}$, and $t_{PXZ}$.

[9] The parameters $t_{ER}$ and $t_{PXZ}$ are measured as the delay from the input disable logic threshold transition to $V_{OH} - 0.5V$ for an enabled HIGH output or $V_{OL} + 0.5V$ for an enabled LOW input.

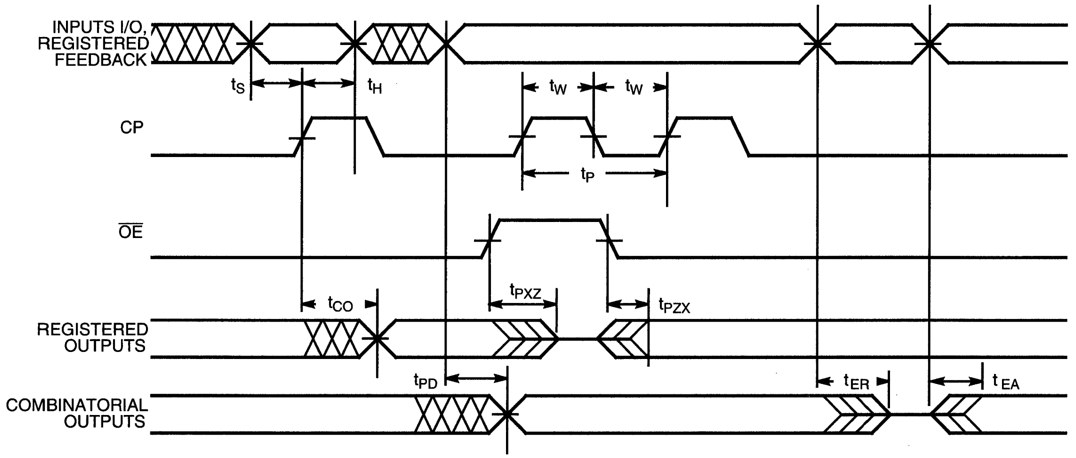
[10] $t_P$, minimum guaranteed clock period is that guaranteed for state machine operation and is calculated from $t_P = t_S + t_{CO}$. The minimum guaranteed period for registered data path operation (no feedback) can be calculated as the greater of $(t_{WH} + t_{WL})$ or $(t_S + t_H)$.

[11] $f_{MAX}$, minimum guaranteed operating frequency, is that guaranteed for state machine operation and is calculated from $f_{MAX} = 1/(t_S + t_{CO})$. The minimum guaranteed $f_{MAX}$ for registered data path operation (no feedback) can be calculated as the lower of $1/(t_{WH} + t_{WL})$ or $1/(t_S + t_H)$.

## Switching Characteristics Over Operating Range[3, 8, 9] (continued)

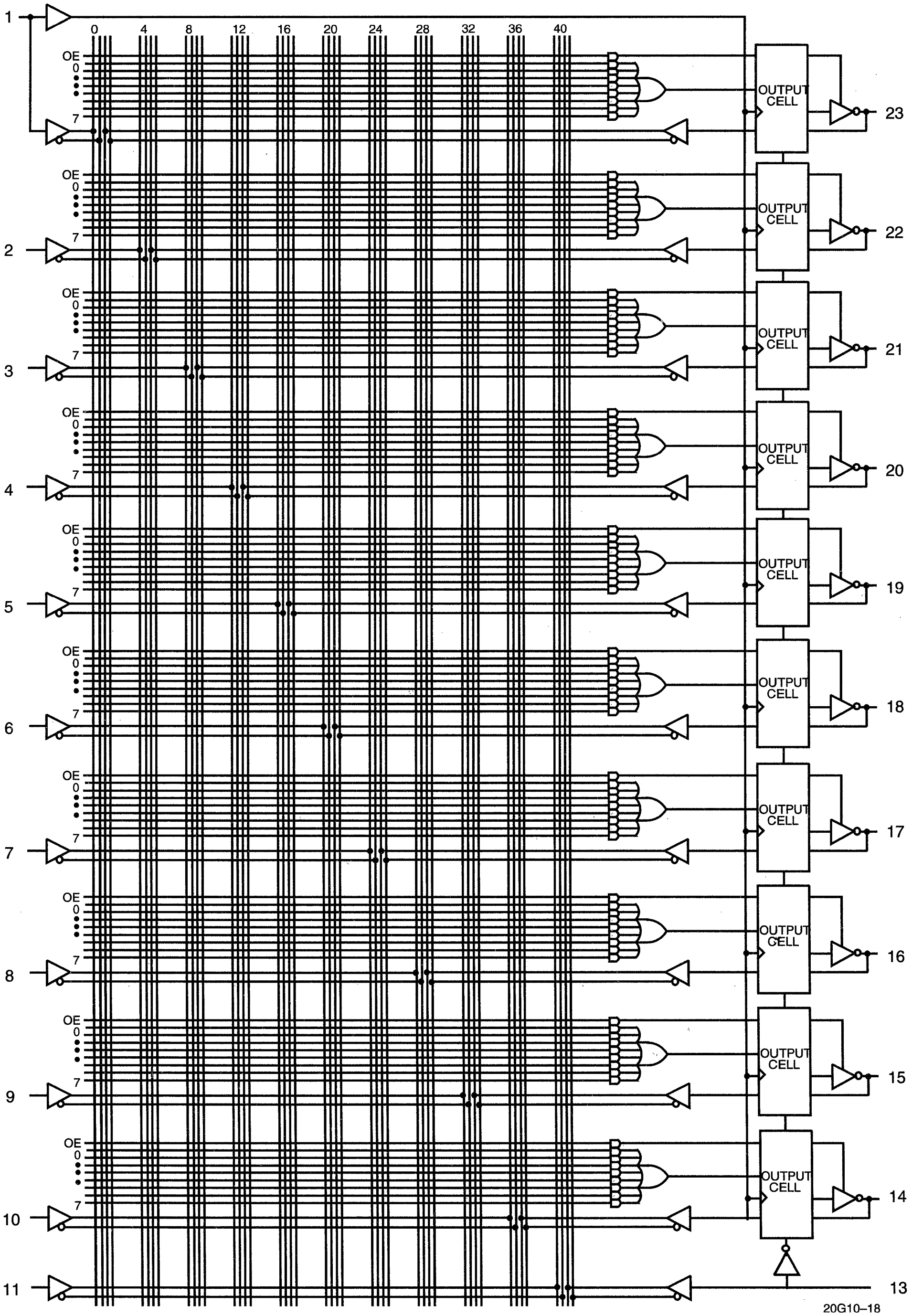
| Parameter | Description | Military/Industrial | | | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | B–20 | | B–25 | | –30 | | –40 | | |
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input or Feedback to Non-Registered Output | | 20 | | 25 | | 30 | | 40 | ns |
| $t_{EA}$ | Input to Output Enable | | 20 | | 25 | | 30 | | 40 | ns |
| $t_{ER}$ | Input to Output Disable | | 20 | | 25 | | 30 | | 40 | ns |
| $t_{PZX}$ | Pin 11 to Output Enable | | 17 | | 20 | | 25 | | 25 | ns |
| $t_{PXZ}$ | Pin 11 to Output Disable | | 17 | | 20 | | 25 | | 25 | ns |
| $t_{CO}$ | Clock to Output | | 15 | | 15 | | 20 | | 25 | ns |
| $t_S$ | Input or Feedback Set-Up Time | 15 | | 18 | | 20 | | 35 | | ns |
| $t_H$ | Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$[10] | Clock Period | 30 | | 33 | | 40 | | 60 | | ns |
| $t_{WH}$ | Clock High Time | 12 | | 14 | | 16 | | 22 | | ns |
| $t_{WL}$ | Clock Low Time | 12 | | 14 | | 16 | | 22 | | ns |
| $f_{MAX}$[11] | Maximum Frequency | 33.3 | | 30.3 | | 25.0 | | 16.6 | | MHz |

## Switching Waveform



20G10–17

## Functional Logic Diagram



20G10–18

## Ordering Information

| $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | $I_{CC}$ (mA) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 15 | 12 | 10 | 70 | PLDC20G10B–15PC | P13 | 24-Lead (300-Mil) Molded DIP | Commercial |
| | | | | PLDC20G10B–15WC | W14 | 24-Lead (300-Mil) Windowed CerDIP | |
| 20 | 15 | 15 | 100 | PLDC20G10B–20DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| 25 | 15 | 15 | 55 | PLDC20G10–25JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PLDC20G10–25PC/PI | P13 | 24-Lead (300-Mil) Molded DIP | Commercial/ Industrial |
| | | | | PLDC20G10–25WC | W14 | 24-Lead (300-Mil) Windowed CerDIP | Commercial |
| 30 | 20 | 20 | 80 | PLDC20G10–30DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PLDC20G10–30LMB | L64 | 28-Square Leadless Chip Carrier | |
| | | | | PLDC20G10–30WMB | W14 | 24-Lead (300-Mil) Windowed CerDIP | |
| 35 | 30 | 25 | 55 | PLDC20G10–35JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PLDC20G10–35PC | P13 | 24-Lead (300-Mil) Molded DIP | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{PZX}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_S$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |

Document #: 38–00019–H

CYPRESS

# PLDC20RA10

# Reprogrammable Asynchronous CMOS Logic Device

## Features

- Advanced-user programmable macrocell
- CMOS EPROM technology for reprogrammability
- Up to 20 input terms
- 10 programmable I/O macrocells
- Output macrocell programmable as combinatorial or asynchronous D-type registered output
- Product-term control of register clock, reset and set and output enable
- Register preload and power-up reset
- Four data product terms per output macrocell
- Fast
  - —Commercial
    $t_{PD}$ = 15 ns
    $t_{CO}$ = 15 ns
    $t_{SU}$ = 7 ns
  - —Military
    $t_{PD}$ = 20 ns
    $t_{CO}$ = 20 ns
    $t_{SU}$ = 10 ns
- Low power
  - —$I_{CC}$ max - 80 mA (Commercial)
  - —$I_{CC}$ max = 85 mA (Military)
- High reliability
  - —Proven EPROM technology
  - —>2001V input protection
  - —100% programming and functional testing
- Windowed DIP, windowed LCC, DIP, LCC, PLCC available

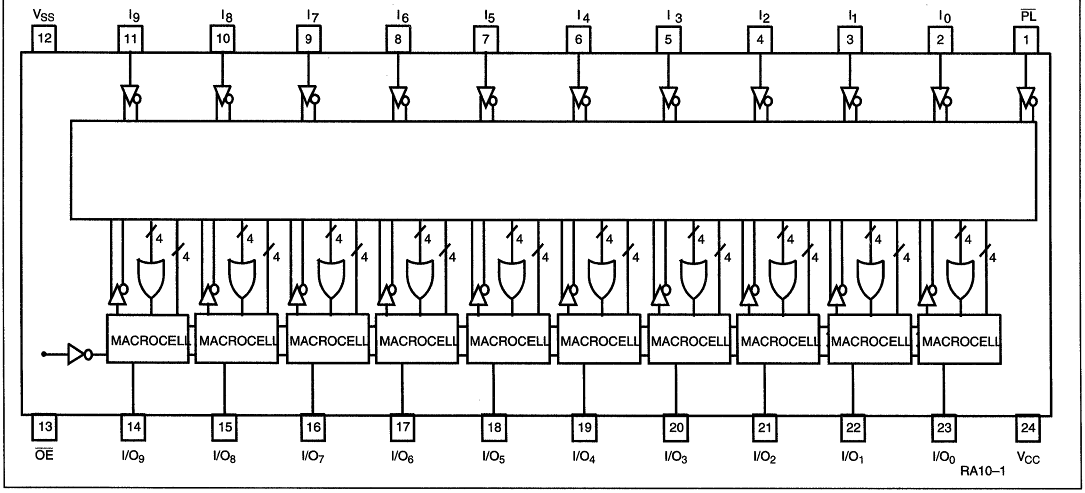
## Functional Description

The Cypress PLDC20RA10 is a high-performance, second-generation programmable logic device employing a flexible macrocell structure that allows any individual output to be configured independently as a combinatorial output or as a fully asynchronous D-type registered output.

The Cypress PLDC20RA10 provides lower-power operation with superior speed performance than functionally equivalent bipolar devices through the use of high-performance 0.8-micron CMOS manufacturing technology.

The PLDC20RA10 is packaged in a 24 pin 300-mil molded DIP, a 300-mil windowed cerDIP, and a 28-lead square leadless chip carrier, providing up to 20 inputs and 10 outputs. When the windowed device is exposed to UV light, the 20RA10 is erased and can then be reprogrammed.

## Logic Block Diagram

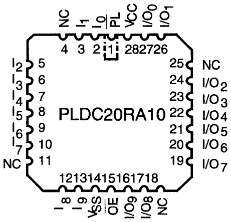


## Selection Guide

| Generic Part Number | $t_{PD}$ ns |  | $t_{SU}$ ns |  | $t_{CO}$ ns |  | $t_{CC}$ ns |  |
|---|---|---|---|---|---|---|---|---|
|  | Com'l | Mil | Com'l | Mil | Com'l | Mil | Com'l | Mil |
| 20RA10-15 | 15 |  | 7 |  | 15 |  | 80 |  |
| 20RA10-20 | 20 | 20 | 10 | 10 | 20 | 20 | 80 | 85 |
| 20RA10-25 |  | 25 |  | 15 |  | 25 |  | 85 |
| 20RA10-35 |  | 35 |  | 20 |  | 35 |  | 85 |

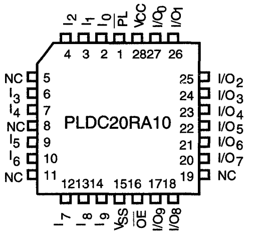
## Pin Configurations

LCC
Top View



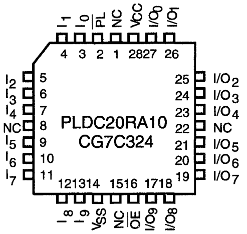
RA10–2

STD PLCC/HLCC
Top View



RA10–3

JEDEC PLCC/HLCC[1]
Top View



RA10–4

## Macrocell Architecture

*Figure 1* illustrates the architecture of the 20RA10 macrocell. The cell dedicates three product terms for fully asynchronous control of the register set, reset, and clock functions, as well as, one term for control of the output enable function.

The output enable product term output is ANDed with the input from pin 13 to allow either product term or hardwired external control of the output or a combination of control from both sources. If product-term-only control is selected, it is automatically chosen for all outputs since, for this case, the external output enable pin must be tied LOW. The active polarity of each output may be programmed independently for each output cell and is subsequently fixed. *Figure 2* illustrates the output enable options available.

When an I/O cell is configured as an output, combinatorial-only capability may be selected by forcing the set and reset product term outputs to be HIGH under all input conditions. This is achieved by programming all input term programming cells for these two product terms. *Figure 3* illustrates the available output configuration options.

An additional four uncommitted product terms are provided in each output macrocell as resources for creation of user-defined logic functions.

## Programmable I/O

Because any of the ten I/O pins may be selected as an input, the device input configuration programmed by the user may vary from a total of nine programmable plus ten dedicated inputs (a total of nineteen inputs) and one output down to a ten-input, ten-output configuration with all ten programmable I/O cells configured as outputs. Each input pin available in a given configuration is available as an input to the four control product terms and four uncommitted product terms of each programmable I/O macrocell that has been configured as an output.

An I/O cell is programmed as an input by tying the output enable pin (pin 13) HIGH or by programming the output enable product term to provide a LOW, thereby disabling the output buffer, for all possible input combinations.

When utilizing the I/O macrocell as an output, the input path functions as a feedback path allowing the output signal to be fed back as an input to the product term array. When the output cell is configured as a registered output, this feedback path may be used to feed back the current output state to the device inputs to provide current state control of the next output state as required for state machine implementation.

## Preload and Power-Up Reset

Functional testability of programmed devices is enhanced by inclusion of register preload capability, which allows the state of each register to be set by loading each register from an external source prior to exercising the device. Testing of complex state machine designs is simplified by the ability to load an arbitrary state without cycling through long test vector sequences to reach the desired state. Recovery from illegal states can be verified by loading illegal states and observing recovery. Preload of a particular register is accomplished by impressing the desired state on the register output pin and lowering the signal level on the preload control pin (pin1) to a logic LOW level. If the specified preload set-up, hold and pulse width minimums have been observed, the desired state is loaded into the register. To insure predictable system initialization, all registers are preset to a logic LOW state upon power-up, thereby setting the active LOW outputs to a logic HIGH.

**Note:**

1. The CG7C324 is the PLDC20RA10 packaged in the JEDEC-compatible 28-pin PLCC pinout. Pin function and pin order is identical for both PLCC pinouts. The principal difference is in the location of the "no connect" (NC) pins

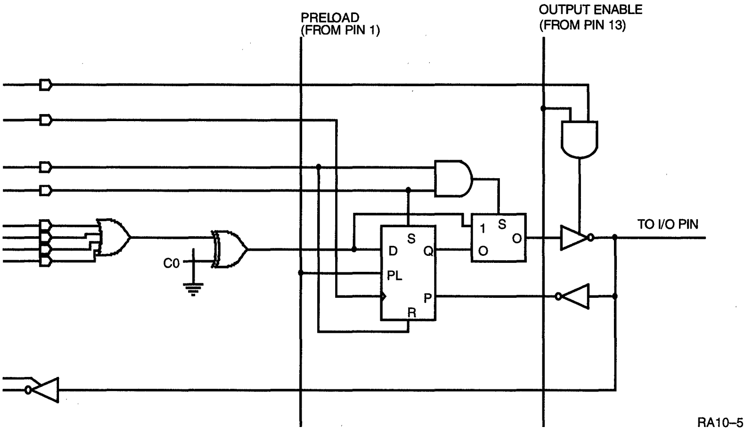
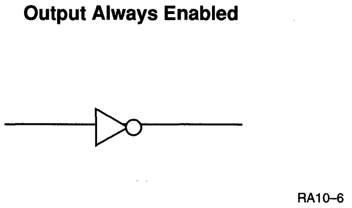
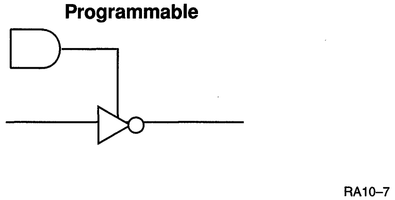
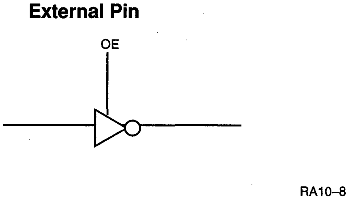


**Figure 1. PLDC20RA10 Macrocell**







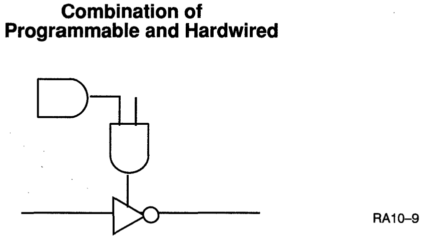
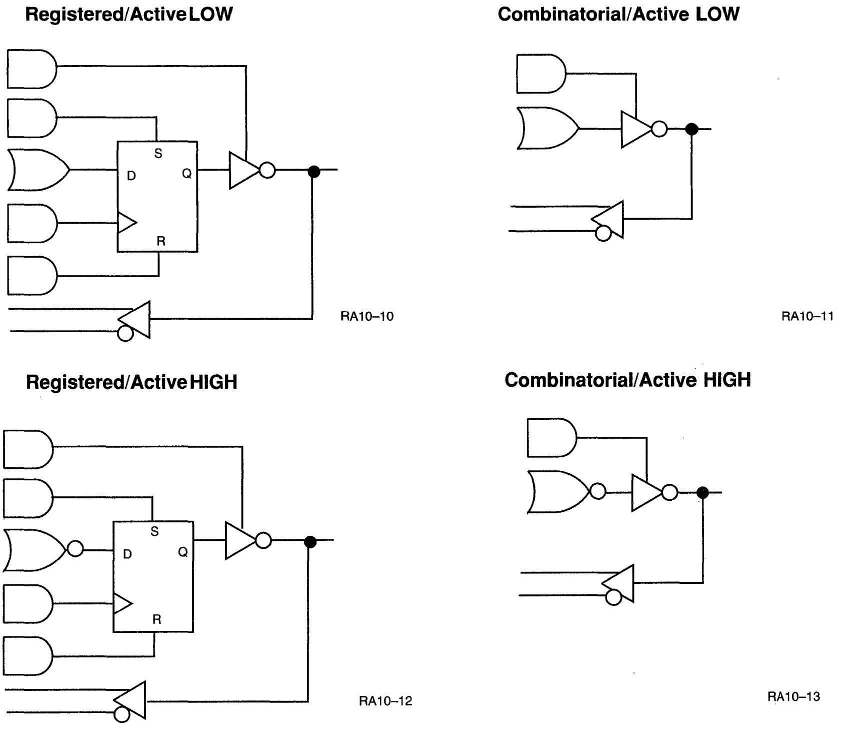


**Figure 2. Four Possible Output Enable Alternatives for the PLDC20RA10**



Registered/Active LOW

Combinatorial/Active LOW

Registered/Active HIGH

Combinatorial/Active HIGH

Figure 3. Four Possible Macrocell Configurations for the PLDC20RA10

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................–65°C to +150°C

Ambient Temperature with
Power Applied ...........................................–55°C to +125°C

Supply Voltage to Ground Potential
(Pin 24 to Pin 12) ..........................................–0.5V to +7.0V

DC Voltage Applied to Outputs
in High Z State ..............................................–0.5V to +7.0V

DC Input Voltage.......................................–3.0 V to + 7.0 V

Output Current into Outputs (LOW) .............................16 mA

Static Discharge Voltage...........................................>2001V
(per MIL-STD-883, Method 3015)

Latch-Up Current ....................................................>200 mA

DC Program Voltage...................................................13.0V

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ± 10% |
| Military[2] | –55°C to +125°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[3]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.,<br>$V_{IN}$ =$V_{IH}$ or $V_{IL}$ | $I_{OH}$ = –3.2 mA | Com'l | 2.4 | | V |
| | | | $I_{OH}$ = –2 mA | Mil | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.,<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 8 mA | | | 0.5 | V |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[4] | | | 2.0 | | V |
| $V_{IL}$ | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[4] | | | | 0.8 | V |
| $I_{IX}$ | Input Leakage Current | $V_{SS} \leq V_{IN} \leq V_{CC}$, $V_{CC}$ = Max | | | –10 | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_{CC}$ = Max., $V_{SS} \leq V_{OUT} \leq V_{CC}$ | | | –40 | +40 | µA |
| $I_{SC}$ | Output Short Circuit Current[5] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[6] | | | –30 | –90 | mA |
| $I_{CC1}$ | Standby Power Supply Current | $V_{CC}$= Max., $V_{IN}$ = GND Outputs Open | | Com'l | | 75 | mA |
| | | | | Mil | | 80 | mA |
| $I_{CC2}$ | Power Supply Current at Frequency[5] | $V_{CC}$ = Max., Outputs Disabled (In High Z State) Device Operating at $f_{MAX}$ | | Com'l | | 80 | mA |
| | | | | Mil | | 85 | mA |

## Capacitance[5]

| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz | 10 | pF |

**Notes:**

[2] $T_A$ is the "instant on" case temperature.

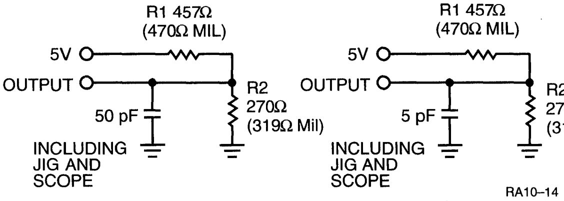
[3] See the last page of this specification for Group A subgroup testing information.

[4] These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.

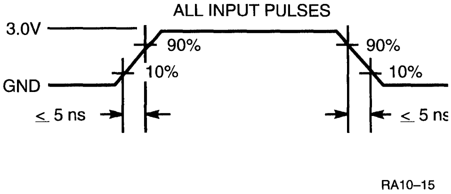
[5] Tested initially and after any design or process changes that may affect these parameters.

[6] Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.

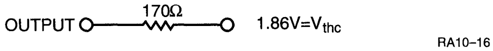
## AC Test Loads and Waveforms (Commercial)



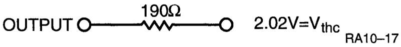
(a) (b)

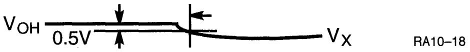
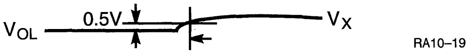
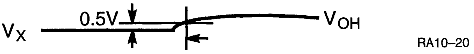
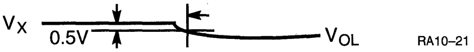
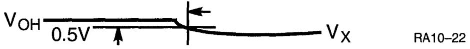
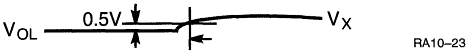
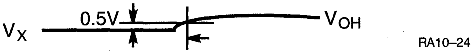
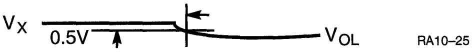


Equivalent to: THÉVENIN EQUIVALENT(Commercial)



Equivalent to: THÉVENIN EQUIVALENT(Military)



| Parameter | $V_{th}$ | Output Waveform Measurement Level |
|---|---|---|
| $t_{PXZ(-)}$ | 1.5V |  |
| $t_{PXZ(+)}$ | 2.6V |  |
| $t_{PZX(+)}$ | $V_{thc}$ |  |
| $t_{PZX(-)}$ | $V_{thc}$ |  |
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | $V_{thc}$ |  |
| $t_{EA(-)}$ | $V_{thc}$ |  |

(c)

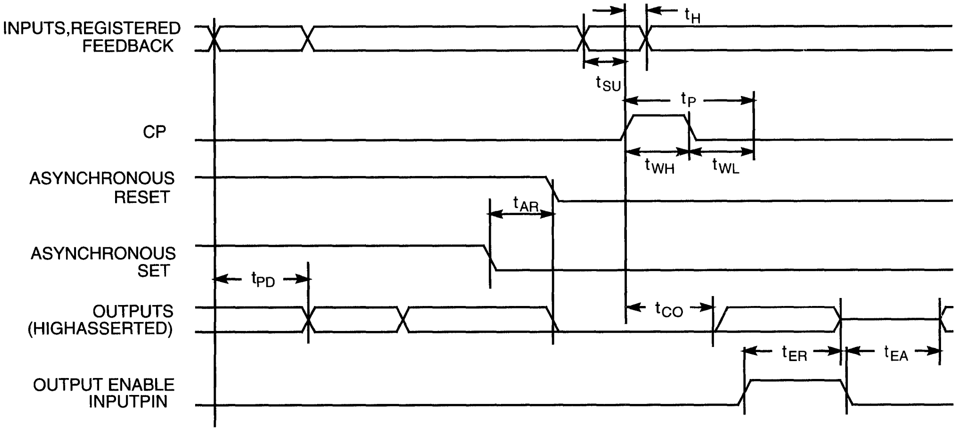
## Switching Characteristics Over the Operating Range[3, 7, 8]

| | | Commercial | | | | Military | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | –15 | | –20 | | –20 | | –25 | | –35 | | |
| Parameter | Description | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| $t_{PD}$ | Input or Feedback to Non-Registered Output | | 15 | | 20 | | 20 | | 25 | | 35 | ns |
| $t_{EA}$ | Input to Output Enable | | 15 | | 20 | | 20 | | 30 | | 35 | ns |
| $t_{ER}$ | Input to Output Disable | | 15 | | 20 | | 20 | | 30 | | 35 | ns |
| $t_{PZX}$ | Pin 13 to Output Enable | | 12 | | 15 | | 15 | | 20 | | 25 | ns |
| $t_{PXZ}$ | Pin 13 to Output Disable | | 12 | | 15 | | 15 | | 20 | | 25 | ns |
| $t_{CO}$ | Clock to Output | | 15 | | 20 | | 20 | | 25 | | 35 | ns |
| $t_{SU}$ | Input or Feedback Set-Up Time | 7 | | 10 | | 10 | | 15 | | 20 | | ns |
| $t_H$ | Hold Time | 3 | | 5 | | 3 | | 5 | | 5 | | ns |
| $t_P$ | Clock Period ($t_{SU}$ + $t_{CO}$) | 22 | | 30 | | 30 | | 40 | | 55 | | ns |
| $t_{WH}$ | Clock Width HIGH[5] | 10 | | 13 | | 12 | | 18 | | 25 | | ns |
| $t_{WL}$ | Clock Width LOW[5] | 10 | | 13 | | 12 | | 18 | | 25 | | ns |
| $f_{MAX}$ | Maximum Frequency ($1/t_P$)[5] | 45.5 | | 33.3 | | 33.3 | | 25.0 | | 18.1 | | MHz |
| $t_S$ | Input of Asynchronous Set to Registered Output | | 15 | | 20 | | 20 | | 25 | | 40 | ns |
| $t_R$ | Input of Asynchronous Reset to Registered Output | | 15 | | 20 | | 20 | | 25 | | 40 | ns |
| $t_{ARW}$ | Asynchronous Reset Width[5] | 15 | | 20 | | 20 | | 25 | | 25 | | ns |
| $t_{ASW}$ | Asynchronous S-Width[5] | 15 | | 20 | | 20 | | 25 | | 25 | | ns |
| $t_{AR}$ | Asynchronous Set/ Reset Recovery Time | 10 | | 12 | | 12 | | 15 | | 20 | | ns |
| $t_{WP}$ | Preload Pulse Width | 15 | | 15 | | 15 | | 15 | | 15 | | ns |
| $t_{SUP}$ | Preload Set-Up Time | 15 | | 15 | | 15 | | 15 | | 15 | | ns |
| $t_{HP}$ | Preload Hold Time | 15 | | 15 | | 15 | | 15 | | 15 | | ns |

**Notes:**

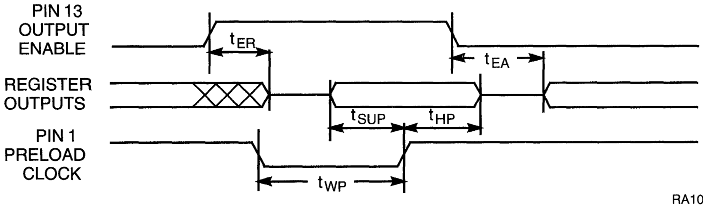
7. Part (a) of AC Test Loads was used for all parameters except $t_{EA}$, $t_{ER}$, $t_{PZX}$ and $t_{PXZ}$, which use part (b).
8. The parameters $t_{ER}$ and $t_{PXZ}$ are measured as the delay from the input disable logic threshold transition to $V_{OH}$ - 0.5 V for an enabled HIGH output or $V_{OL}$ +0.5V for an enabled LOW output. Please see part (c) of AC Test Loads and Waveforms for waveforms and measurement reference levels.

## Switching Waveform



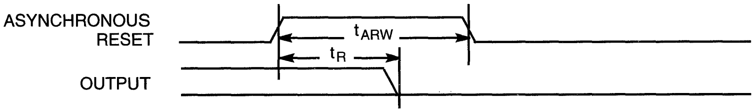
RA10–26

## Preload Switching Waveform



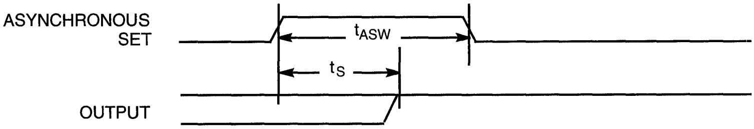
RA10–27

## Asynchronous Reset



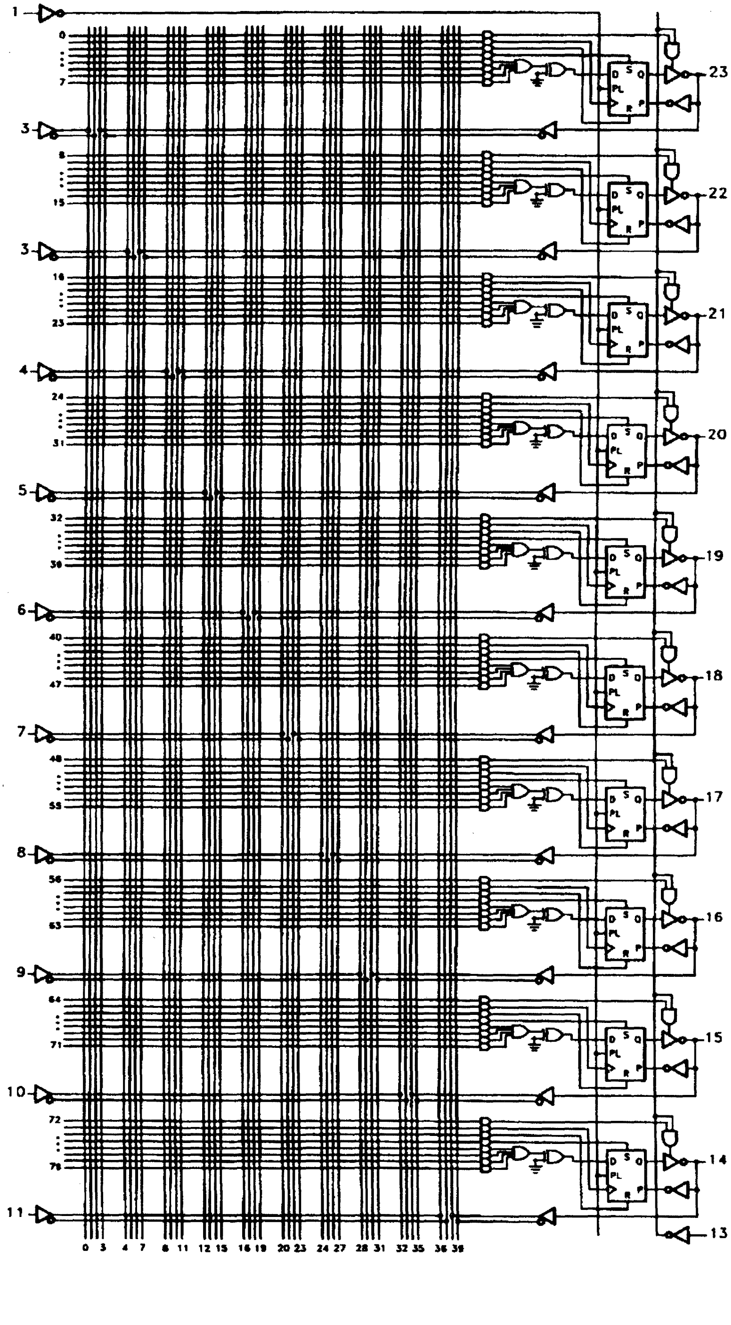
RA10–28

## Asynchronous Set



RA10–29

## Functional Logic Diagram



## Ordering Information

| $I_{CC2}$ | $t_{PD}$ (ns) | $t_{SU}$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 80 | 15 | 7 | 15 | PLDC20RA10-15JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PLDC20RA10-15PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | CG7C324-A15JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | 20 | 10 | 20 | PLDC20RA10-20PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| | | | | CG7C324-A20JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| 85 | 20 | 10 | 20 | PLDC20RA10-20DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PLDC20RA10-20WMB | W14 | 24-Lead (300-Mil) Windowed CerDIP | |
| | 25 | 15 | 25 | PLDC20RA10-25DMB | D14 | 24-Lead (300-Mil) CerDIP | |
| | | | | PLDC20RA10-25WMB | W14 | 24-Lead (300-Mil) Windowed CerDIP | |
| | 35 | 20 | 35 | PLDC20RA10-35DMB | D14 | 24-Lead (300-Mil) CerDIP | |
| | | | | PLDC20RA10-35WMB | W14 | 24-Lead (300-Mil) Windowed CerDIP | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{PZX}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{SU}$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |

Document #: 38-00073-F

CYPRESS

# PALCE22V10

# Flash Erasable, Reprogrammable CMOS PAL® Device

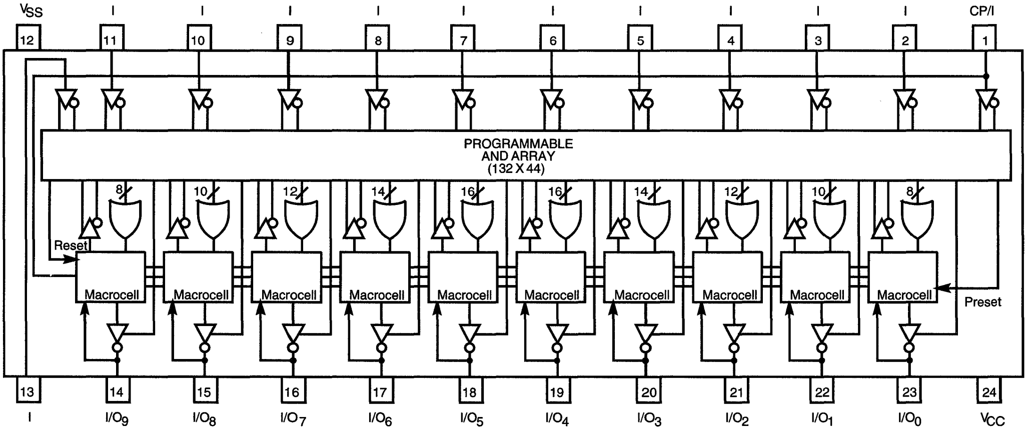
## Features

- Low power
  - —90 mA max. commercial (10 ns)
  - —130 mA max. commercial (5 ns)
- CMOS Flash EPROM technology for electrical erasability and reprogrammability
- Variable product terms
  - —2 x(8 through 16) product terms
- User-programmable macrocell
  - —Output polarity control
  - —Individually selectable for registered or combinatorial operation
- Up to 22 input terms and 10 outputs
- DIP, LCC, and PLCC available
  - —5 ns commercial version
    4 ns $t_{CO}$
    3 ns $t_S$
    5 ns $t_{PD}$
    181-MHz state machine
  - —10 ns military and industrial versions
    7 ns $t_{CO}$
    6 ns $t_S$
    10 ns $t_{PD}$
    110-MHz state machine
  - —15-ns commercial, industrial, and military versions
  - —25-ns commercial, industrial, and military versions
- High reliability
  - —Proven Flash EPROM technology
  - —100% programming and functional testing

## Functional Description

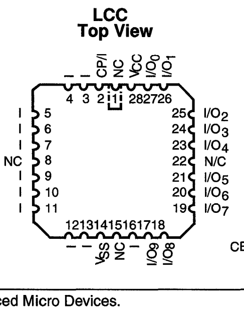
The Cypress PALCE22V10 is a CMOS Flash Erasable second-generation programmable array logic device. It is implemented with the familiar sum-of-products (AND-OR) logic structure and the programmable macrocell.

## Logic Block Diagram (PDIP/CDIP)

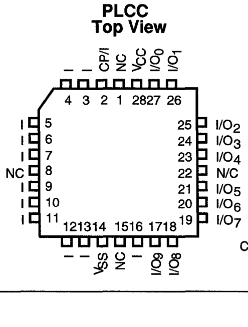


CE22V10–1

## Pin Configuration



CE22V10–2



CE22V10–3

PAL is a registered trademark of Advanced Micro Devices.

## Selection Guide

| | $t_{PD}$ ns | | $t_S$ ns | | $t_{CO}$ ns | | $I_{CC}$ mA | |
|---|---|---|---|---|---|---|---|---|
| Generic Part Number | Com'l | Mil/Ind | Com'l | Mil/Ind | Com'l | Mil/Ind | Com'l | Mil/Ind |
| PALCE22V10-5 | 5 | | 3 | | 4 | | 130 | |
| PALCE22V10-7 | 7.5 | | 5 | | 5 | | 130 | |
| PALCE22V10-10 | 10 | 10 | 6 | 6 | 7 | 7 | 90 | 150 |
| PALCE22V10-15 | 15 | 15 | 10 | 10 | 8 | 8 | 90 | 120 |
| PALCE22V10-25 | 25 | 25 | 15 | 15 | 15 | 15 | 90 | 120 |

## Functional Description (continued)

The PALCE22V10 is executed in a 24-pin 300-mil molded DIP, a 300-mil cerDIP, a 28-lead square ceramic leadless chip carrier, a 28-lead square plastic leaded chip carrier, and provides up to 22 inputs and 10 outputs. The PALCE22V10 can be electrically erased and reprogrammed. The programmable macrocell provides the capability of defining the architecture of each output individually. Each of the 10 potential outputs may be specified as "registered" or "combinatorial." Polarity of each output may also be individually selected, allowing complete flexibility of output configuration. Further configurability is provided through "array" configurable "output enable" for each potential output. This feature allows the 10 outputs to be reconfigured as inputs on an individual basis, or alternately used as a combination I/O controlled by the programmable array.

PALCE22V10 features a variable product term architecture. There are 5 pairs of product term sums beginning at 8 product terms per output and incrementing by 2 to 16 product terms per output. By providing this variable structure, the PALCE 22V10 is optimized to the configurations found in a majority of applications without creating devices that burden the product term structures with unusable product terms and lower performance.

Additional features of the Cypress PALCE22V10 include a synchronous preset and an asynchronous reset product term. These product terms are common to all macrocells, eliminating the need to dedicate standard product terms for initialization functions. The device automatically resets upon power-up.

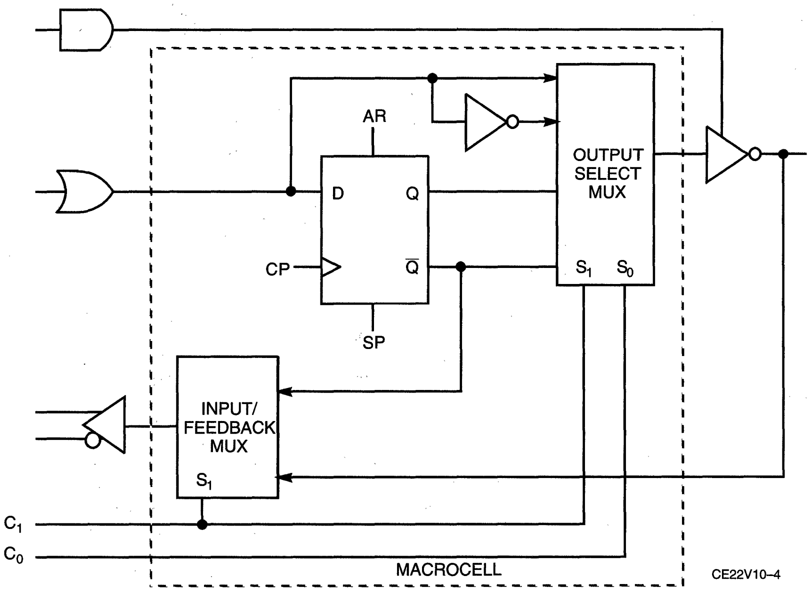
The PALCE22V10, featuring programmable macrocells and variable product terms, provides a device with the flexibility to implement logic functions in the 500- to 800-gate-array complexity. Since each of the 10 output pins may be individually configured as inputs on a temporary or permanent basis, functions requiring up to 21 inputs and only a single output and down to 12 inputs and 10 outputs are possible. The 10 potential outputs are enabled using product terms. Any output pin may be permanently selected as an output or arbitrarily enabled as an output and an input through the selective use of individual product terms associated with each output. Each of these outputs is achieved through an individual programmable macrocell. These macrocells are programmable to provide a combinatorial or registered inverting or non-inverting output. In a registered mode of operation, the output of the register is fed back into the array, providing current status information to the array. This information is available for establishing the next result in applications such as control state machines. In a combinatorial configuration, the combinatorial output or, if the output is disabled, the signal present on the I/O pin is made available to the array. The flexibility provided by both programmable product term control of the outputs and variable product terms allows a significant gain in functional density through the use of programmable logic.

Along with this increase in functional density, the Cypress PALCE22V10 provides lower-power operation through the use of CMOS technology, and increased testability with Flash reprogrammability.

## Configuration Table

| Registered/Combinatorial | | |
|---|---|---|
| $C_1$ | $C_0$ | Configuration |
| 0 | 0 | Registered/Active LOW |
| 0 | 1 | Registered/Active HIGH |
| 1 | 0 | Combinatorial/Active LOW |
| 1 | 1 | Combinatorial/Active HIGH |

## Macrocell



## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | −65°C to +150°C |
| Ambient Temperature with Power Applied | −55°C to +125°C |
| Supply Voltage to Ground Potential (Pin 24 to Pin 12) | −0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | −0.5V to +7.0V |
| DC Input Voltage | −0.5V to +7.0V |
| Output Current into Outputs (LOW) | 16 mA |
| DC Programming Voltage | 12.5V |
| Latch-Up Current | >200 mA |
| Static Discharge Voltage (per MIL-STD-883, Method 3015) | >2001V |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ±5% |
| Industrial | −40°C to +85°C | 5V ±10% |
| Military[1] | −55°C to +125°C | 5V ±10% |

**Note:**

[1] $T_A$ is the “instant on” case temperature.

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | Com'l | 2.4 | | V |
| | | | $I_{OH}$ = −2 mA | Mil/Ind | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 16 mA | Com'l | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | Mil/Ind | | | |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[3] | | | 2.0 | | V |
| $V_{IL}$[4] | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[3] | | | −0.5 | 0.8 | V |
| $I_{IX}$ | Input Leakage Current | $V_{SS} \leq V_{IN} \leq V_{CC}$, $V_{CC}$ = Max. | | | −10 | 10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_{CC}$ = Max., $V_{SS} \leq V_{OUT} \leq V_{CC}$ | | | −40 | 40 | µA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[5,6] | | | −30 | −130 | mA |
| $I_{CC1}$ | Standby Power Supply Current | $V_{CC}$ = Max., $V_{IN}$ = GND, Outputs Open in Unprogrammed Device | 10, 15, 25 ns | Com'l | | 90 | mA |
| | | | 5, 7.5 ns | | | 130 | mA |
| | | | 15, 25 ns | Mil/Ind | | 120 | mA |
| | | | 10 ns | | | 120 | mA |
| $I_{CC2}$[6] | Operating Power Supply Current | $V_{CC}$ = Max., $V_{IL}$ = 0V, $V_{IH}$ = 3V, Output Open, Device Programmed as a 10-Bit Counter, f = 25 MHz | 10, 15, 25 ns | Com'l | | 110 | mA |
| | | | 5, 7.5 ns | Com'l | | 140 | mA |
| | | | 15, 25 ns | Mil/Ind | | 130 | mA |
| | | | 10 ns | Mil/Ind | | 130 | mA |

## Capacitance[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0V @ f = 1 MHz | | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V @ f = 1 MHz | | 10 | pF |

## Endurance Characteristics[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | | Cycles |

**Notes:**

[2] See the last page of this specification for Group A subgroup testing information.

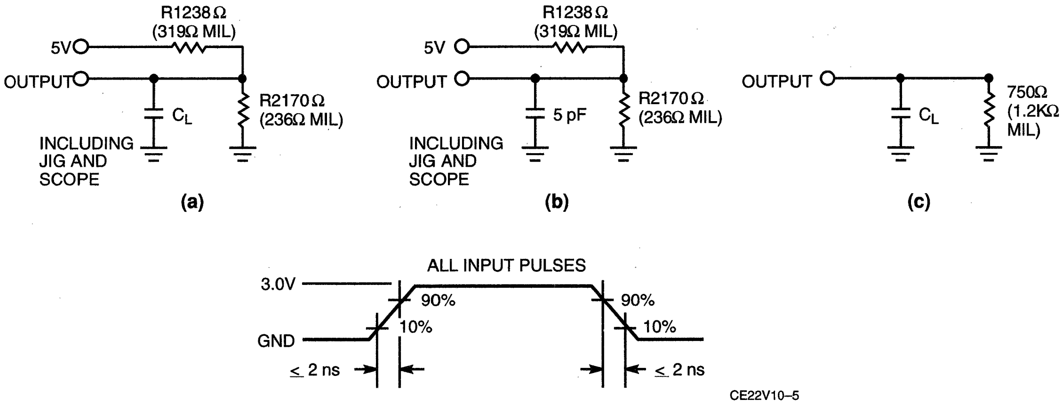
[3] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[4] $V_{IL}$ (Min.) is equal to -3.0V for pulse durations less than 20 ns.

[5] Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[6] Tested initially and after any design or process changes that may affect these parameters.

## AC Test Loads and Waveforms



(a)

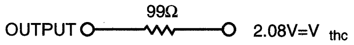
(b)

(c)

CE22V10–5

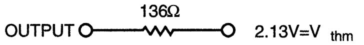
(d)

Equivalent to: THÉVENIN EQUIVALENT(Commercial)



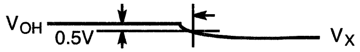
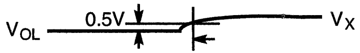
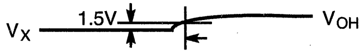
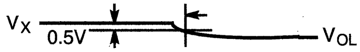
CE22V10–6

Equivalent to: THÉVENIN EQUIVALENT(Military)



CE22V10–7

| Load Speed | $C_L$ | Package |
|---|---|---|
| 5, 7.5, 10, 15, 25 ns | 50 pF | PDIP, CDIP, PLCC, LCC |

| Parameter | $V_X$ | Output Waveform Measurement Level |
|---|---|---|
| $t_{ER}$ (–) | 1.5V | <br> |
| $t_{ER}$ (+) | 2.6V | <br> |
| $t_{EA}$ (+) | 0V | <br> |
| $t_{EA}$ (–) | $V_{thc}$ | <br> |

(e) Test Waveforms

## Commercial Switching Characteristics PALCE22V10[2,7]

| Parameter | Description | 22V10-5 | | 22V10-7 | | 22V10-10 | | 22V10-15 | | 22V10-25 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8] | 3 | 5 | 3 | 7.5 | 3 | 10 | 3 | 15 | 3 | 25 | ns |
| $t_{EA}$ | Input to Output Enable Delay[9] | | 6 | | 8 | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[10] | | 6 | | 8 | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8] | 2 | 4 | 2 | 5 | 2 | 7 | 2 | 8 | 2 | 15 | ns |
| $t_{S1}$ | Input or Feedback Set-Up Time | 3 | | 5 | | 6 | | 10 | | 15 | | ns |
| $t_{S2}$ | Synchronous Preset Set-Up Time | 4 | | 6 | | 7 | | 10 | | 15 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 7 | | 10 | | 12 | | 20 | | 30 | | ns |
| $t_{WH}$ | Clock Width HIGH[6] | 2.5 | | 3 | | 3 | | 6 | | 13 | | ns |
| $t_{WL}$ | Clock Width LOW[6] | 2.5 | | 3 | | 3 | | 6 | | 13 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$)[11] | 143 | | 100 | | 76.9 | | 55.5 | | 33.3 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[6, 12] | 200 | | 166 | | 142 | | 83.3 | | 35.7 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[6,13] | 181 | | 133 | | 111 | | 68.9 | | 38.5 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[6,14] | | 2.5 | | 2.5 | | 3 | | 4.5 | | 13 | ns |
| $t_{AW}$ | Asynchronous Reset Width | 8 | | 8 | | 10 | | 15 | | 25 | | ns |
| $t_{AR}$ | Asynchronous Reset Recovery Time | 4 | | 5 | | 6 | | 10 | | 25 | | ns |
| $t_{AP}$ | Asynchronous Reset to Registered Output Delay | | 7.5 | | 12 | | 13 | | 20 | | 25 | ns |
| $t_{SPR}$ | Synchronous Preset Recovery Time | 4 | | 6 | | 8 | | 10 | | 15 | | ns |
| $t_{PR}$ | Power-Up Reset Time[6,15] | 1 | | 1 | | 1 | | 1 | | 1 | | µs |

**Notes:**

[7] Part (a) of AC Test Loads and Waveforms is used for all parameters except $t_{ER}$ and $t_{EA(+)}$. Part (b) of AC Test Loads and Waveforms is used for $t_{ER}$. Part (c) of AC Test Loads and Waveforms is used for $t_{EA(+)}$.

[8] Min. times are tested initially and after any design or process changes that may affect these parameters.

[9] The test load of part (a) of AC Test Loads and Waveforms is used for measuring $t_{EA(-)}$. The test load of part (c) of AC Test Loads and Waveforms is used for measuring $t_{EA(+)}$ only. Please see part (e) of AC Test Loads and Waveforms for enable and disable test waveforms and measurement reference levels.

[10] This parameter is measured as the time after output disable input that the previous output data state remains stable on the output. This delay is measured to the point at which a previous HIGH level has fallen to 0.5 volts below $V_{OH}$ min. or a previous LOW level has risen to 0.5 volts above $V_{OL}$ max. Please see part (e) of AC Test Loads and Waveforms for enable and disable test waveforms and measurement reference levels.

[11] This specification indicates the guaranteed maximum frequency at which a state machine configuration with external feedback can operate.

[12] This specification indicates the guaranteed maximum frequency at which the device can operate in data path mode.

[13] This specification indicates the guaranteed maximum frequency at which a state machine configuration with internal only feedback can operate.

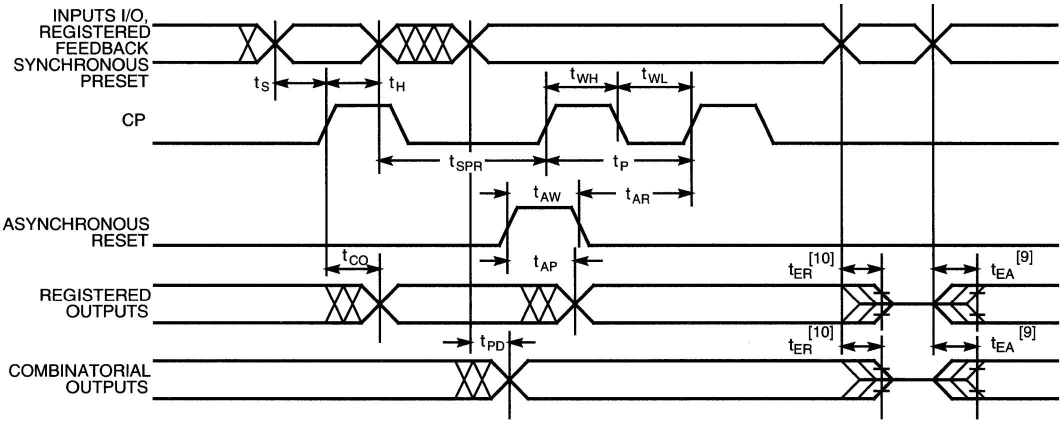
[14] This parameter is calculated from the clock period at $f_{MAX}$ internal ($1/f_{MAX3}$) as measured (see Note above) minus $t_S$.

[15] The registers in the PALCE22V10 have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to a logic LOW state. The output state will depend on the polarity of the output buffer. This feature is useful in establishing state machine initialization. To insure proper operation, the rise in $V_{CC}$ must be monotonic and the timing constraints depicted in Power-Up Reset Waveform must be satisfied

## Military and Industrial Switching Characteristics PALCE22V10[2,7]

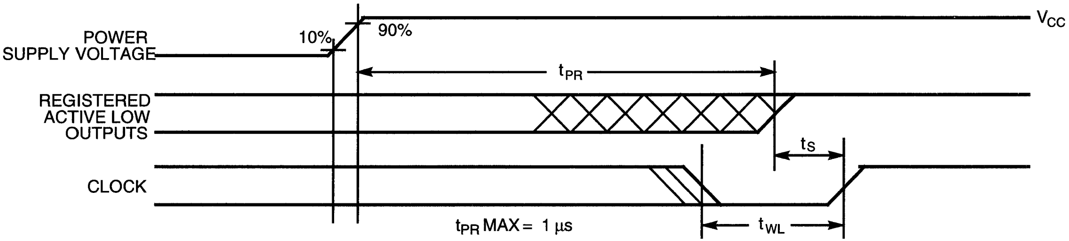
| Parameter | Description | 22V10-10 | | 22V10-15 | | 22V10-25 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8] | 3 | 10 | 3 | 15 | 3 | 25 | ns |
| $t_{EA}$ | Input to Output Enable Delay[9] | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[10] | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8] | 2 | 7 | 2 | 8 | 2 | 15 | ns |
| $t_{S1}$ | Input or Feedback Set-Up Time | 6 | | 10 | | 18 | | ns |
| $t_{S2}$ | Synchronous Preset Set-Up Time | 7 | | 10 | | 18 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 12 | | 20 | | 33 | | ns |
| $t_{WH}$ | Clock Width HIGH[6] | 3 | | 6 | | 14 | | ns |
| $t_{WL}$ | Clock Width LOW[6] | 3 | | 6 | | 14 | | ns |
| $f_{MAX1}$ | External Maximum Frequency (1/($t_{CO} + t_S$))11] | 76.9 | | 50.0 | | 30.3 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency (1/($t_{WH} + t_{WL}$))[6,12] | 142 | | 83.3 | | 35.7 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency (1/($t_{CF} + t_S$))[6,13] | 111 | | 68.9 | | 32.2 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[6,14] | | 3 | | 4.5 | | 13 | ns |
| $t_{AW}$ | Asynchronous Reset Width | 10 | | 15 | | 25 | | ns |
| $t_{AR}$ | Asynchronous Reset Recovery Time | 6 | | 12 | | 25 | | ns |
| $t_{AP}$ | Asynchronous Reset to Registered Output Delay | | 12 | | 20 | | 25 | ns |
| $t_{SPR}$ | Synchronous Preset Recovery Time | 8 | | 20 | | 25 | | ns |
| $t_{PR}$ | Power-Up Reset Time[6,15] | 1 | | 1 | | 1 | | µs |

## Switching Waveforms



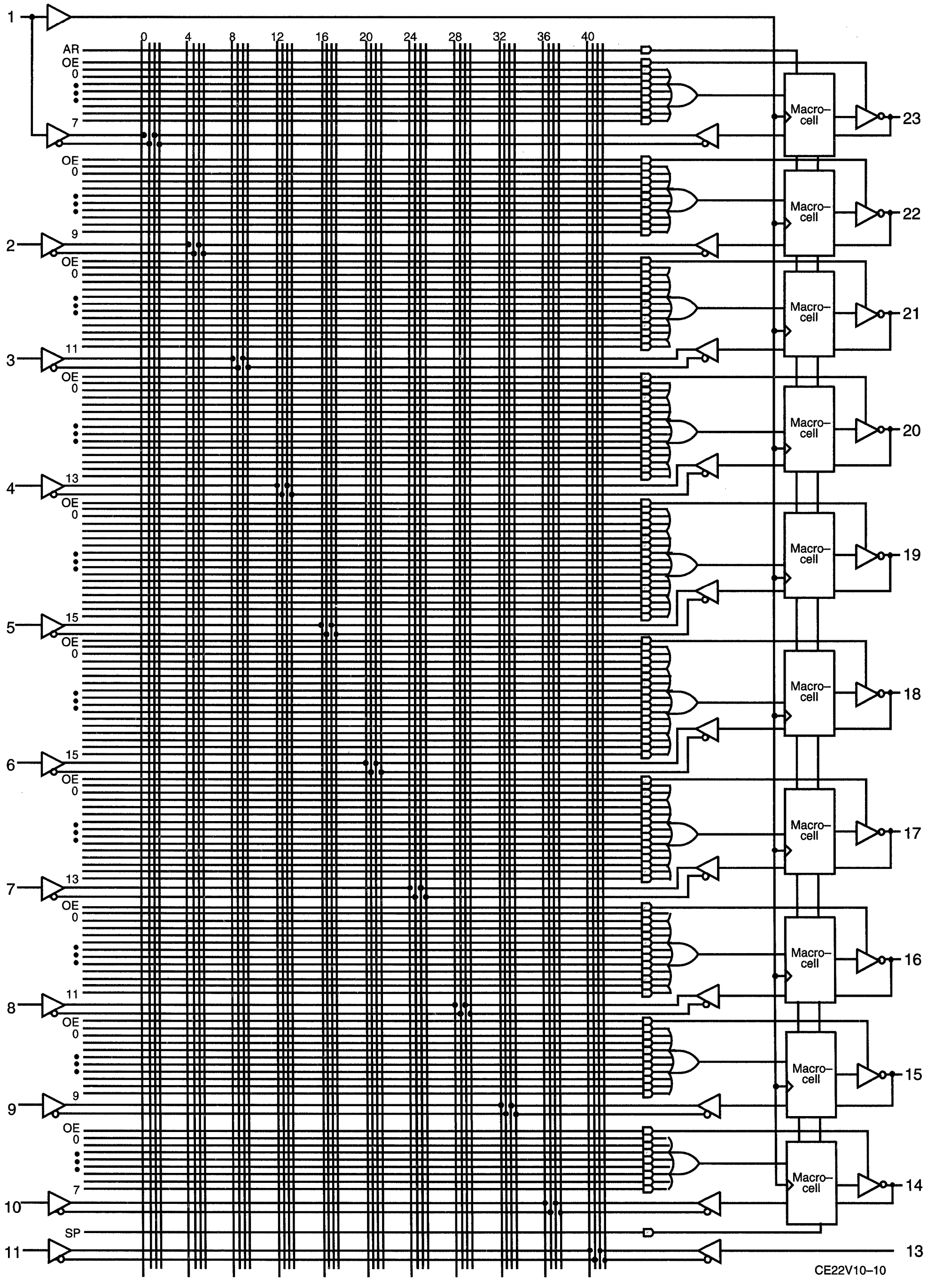
CE22V10–8

## Power-Up Reset Waveform[15]



CE22V10–9

## Functional Logic Diagram for PALCE22V10



CE22V10–10

## Ordering Information

| $I_{CC}$ (mA) | $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 130 | 5 | 3 | 4 | PALCE22V10-5AC | P13 | 24-Lead (300 MIL) Molded DIP | Commercial |
| 130 | 5 | 3 | 4 | PALCE22V10-5JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| 130 | 7.5 | 5 | 5 | PALCE22V10-7JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE22V10-7PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 90 | 10 | 6 | 7 | PALCE22V10-10JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE22V10-10PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 150 | 10 | 6 | 7 | PALCE22V10-10JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE22V10-10PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| 150 | 10 | 6 | 7 | PALCE22V10-10DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE22V10-10KMB | K73 | 24-Lead Rectangular Cerpack | |
| | | | | PALCE22V10-10LMB | L64 | 28-Square Leadless Chip Carrier | |
| 90 | 15 | 7.5 | 10 | PALCE22V10-15JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE22V10-15PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 15 | 7.5 | 10 | PALCE22V10-15JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE22V10-15PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 15 | 7.5 | 10 | PALCE22V10-15DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE22V10-15KMB | K73 | 24-Lead Rectangular Cerpack | |
| | | | | PALCE22V10-15LMB | L64 | 28-Square Leadless Chip Carrier | |
| 90 | 25 | 15 | 15 | PALCE22V10-25JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALCE22V10-25PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 25 | 15 | 15 | PALCE22V10-25JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALCE22V10-25PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 25 | 15 | 15 | PALCE22V10-25DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALCE22V10-25KMB | K73 | 24-Lead Rectangular Cerpack | |
| | | | | PALCE22V10-25LMB | L64 | 28-Square Leadless Chip Carrier | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_S$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |

Document #: 38-00447-B

**This is an abbreviated datasheet. Contact a Cypress representative for complete specifications. For new designs, please refer to the PALCE22V10.**

CYPRESS

# PALC22V10

# Reprogrammable CMOS PAL® Device

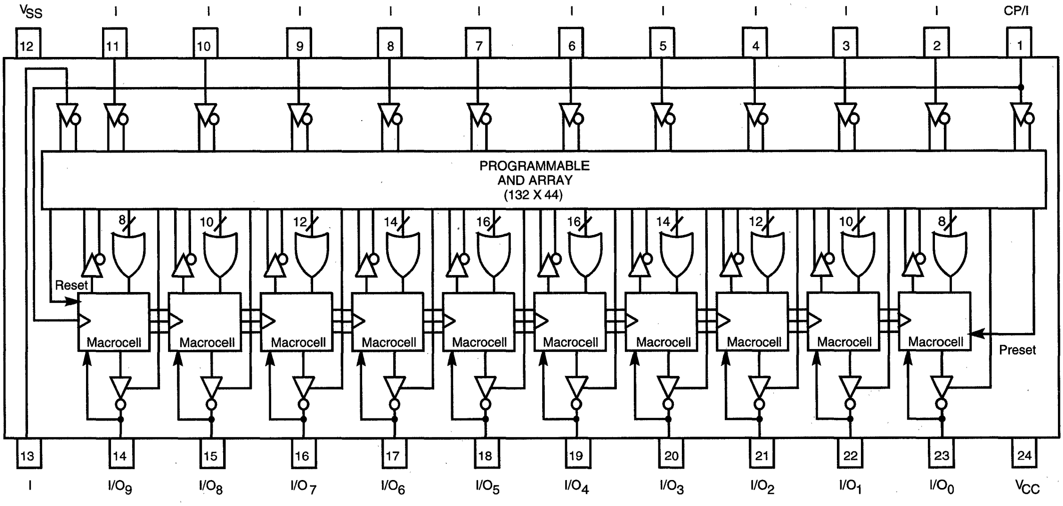
## Features

- **Advanced second-generation PAL architecture**
- **Low power**
  - **—55 mA max. "L"**
  - **—90 mA max. standard**
  - **—120 mA max. military**
- **CMOS EPROM technology for reprogrammability**
- **Variable product terms**
  - **—2 x (8 through 16) product terms**
- **User-programmable macrocell**
  - **—Output polarity control**
  - **—Individually selectable for registered or combinatorial operation**
- **20, 25, 35 ns commercial and industrial**
- **25, 30, 40 ns military**
- **Up to 22 input terms and 10 outputs**
- **High reliability**
  - **—Proven EPROM technology**
  - **—100% programming and functional testing**
- **Windowed DIP, windowed LCC, DIP, LCC, and PLCC available**

## Functional Description

The Cypress PALC22V10 is a CMOS second-generation programmable logic array device. It is implemented with the familiar sum-of-products (AND-OR) logic structure and a new concept, the "programmable macrocell."

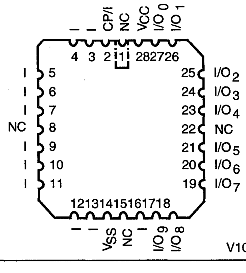
## Logic Block Diagram (PDIP/CDIP)



V10–1

## Pin Configuration

**LCC/PLCC**
**Top View**



V10–2

PAL is a registered trademark of Advanced Micro Devices.

## Functional Description (continued)

The PALC22V10 is available in 24-pin 300-mil molded DIPs, 300-mil windowed cerDIPs, 28-lead square ceramic leadless chip carriers, 28-lead square plastic leaded chip carriers, and provides up to 22 inputs and 10 outputs. When the windowed cerDIP is exposed to UV light, the 22V10 is erased and can then be reprogrammed. The programmable macrocell provides the capability of defining the architecture of each output individually. Each of the 10 potential outputs may be specified as registered or combinatorial. Polarity of each output may also be individually selected, allowing complete flexibility of output configuration. Further configurability is provided through array-configurable output enable for each potential output. This feature allows the 10 outputs to be reconfigured as inputs on an individual basis, or alternately used as a combination I/O controlled by the programmable array.

Document #: 38-00020-H

CYPRESS

# PALC22V10B

# Reprogrammable CMOS PAL® Device

## Features

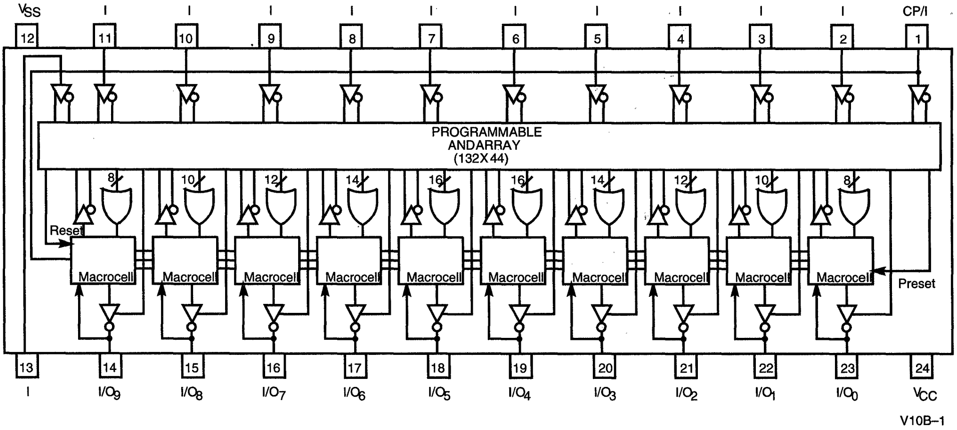
- **Advanced second generation PAL architecture**
- **Low power**
  - **—90 mA max. standard**
  - **—100 mA max. military**
- **CMOS EPROM technology for reprogrammability**
- **Variable product terms**
  - **—2 x (8 through 16) product terms**
- **User-programmable macrocell**
  - **—Output polarity control**
  - **—Individually selectable for registered or combinatorial operation**
  - **—15 ns commercial and industrial**
    **10 ns $t_{CO}$**
    **10 ns $t_S$**
    **15 ns $t_{PD}$**
    **50 MHz**
  - **—15 ns and "20 ns" military**
    **10/15 ns $t_{CO}$**
    **10/17 ns $t_S$**
    **15/20 ns $t_{PD}$**
    **50/31 MHz**
- **Up to 22 input terms and 10 outputs**
- **Enhanced test features**
  - **—Phantom array**
  - **—Top test**
  - **—Bottom test**
  - **—Preload**
- **High reliability**
  - **—Proven EPROM technology**
  - **—100% programming and functional testing**
- **Windowed DIP, windowed LCC, DIP, LCC, PLCC available**

## Functional Description

The Cypress PALC22V10B is a CMOS second-generation programmable logic array device. It is implemented with the familiar sum-of-products (AND-OR) logic structure and a new concept, the "Programmable Macrocell."

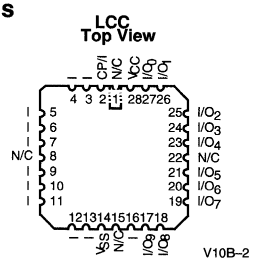
The PALC22V10B is executed in a 24-pin 300-mil molded DIP, a 300-mil windowed cerDIP, a 28-lead square ceramic leadless chip carrier, a 28-lead square plastic leaded chip carrier, and provides up to 22 inputs and 10 outputs. When the windowed cerDIP is exposed to UV light, the 22V10B is erased and can then be reprogrammed. The programmable macrocell provides the capability of defining the architecture of each output individually. Each of the 10 potential outputs may be specified as "registered" or "combinatorial." Polarity of each output may also be individually

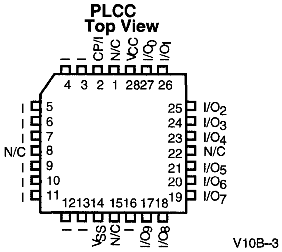
## Logic Block Diagram (PDIP/CDIP)



PAL is a registered trademark of Advanced Micro Devices.

## Pin Configurations





Document #: 38-00195-C

CYPRESS

# PALC22V10D

# Flash Erasable, Reprogrammable CMOS PAL® Device

## Features

- **Advanced second-generation PAL architecture**
- **Low power**
  - **—90 mA max. commercial (10 ns)**
  - **—130 mA max. commercial (7.5 ns)**
- **CMOS Flash EPROM technology for electrical erasability and reprogrammability**
- **Variable product terms**
  - **—2 x(8 through 16) product terms**
- **User-programmable macrocell**
  - **—Output polarity control**
  - **—Individually selectable for registered or combinatorial operation**
- **Up to 22 input terms and 10 outputs**
- **DIP, LCC, and PLCC available**
  - **—7.5 ns commercial version**
    **5 ns $t_{CO}$**
    **5 ns $t_S$**
    **7.5 ns $t_{PD}$**
    **133-MHz state machine**
  - **—10 ns military and industrial versions**
    **6 ns $t_{CO}$**
    **6 ns $t_S$**
    **10 ns $t_{PD}$**
    **110-MHz state machine**
  - **—15-ns commercial and military versions**
  - **—25-ns commercial and military versions**
- **High reliability**
  - **—Proven Flash EPROM technology**
- **100% programming and functional testing**

## Functional Description

The Cypress PALC22V10D is a CMOS Flash Erasable second-generation programmable array logic device. It is implemented with the familiar sum-of-products (AND-OR) logic structure and the programmable macrocell.

The PALC22V10D is executed in a 24-pin 300-mil molded DIP, a 300-mil cerDIP, a 28-lead square ceramic leadless chip carrier, a 28-lead square plastic leaded chip carrier, and provides up to 22 inputs and 10 outputs. The 22V10D can be electrically erased and reprogrammed. The programmable macrocell provides the capability of defining the architecture of each output individually. Each of the 10 potential outputs may be specified as "registered" or "combinatorial." Polarity of each output may also be individually selected, allowing complete flexibility of output configuration. Further configurability is provided through "array" configurable "output enable" for each potential output. This feature allows the 10 outputs to be reconfigured as inputs on an individual basis, or alternately used as a combination I/O controlled by the programmable array.

PALC22V10D features a variable product term architecture. There are 5 pairs of product term sums beginning at 8 product terms per output and incrementing by 2 to 16 product terms per output. By providing this variable structure, the PAL C 22V10D is optimized to the configurations found in a majority of applications without creating devices that burden the product term structures with unusable product terms and lower performance.

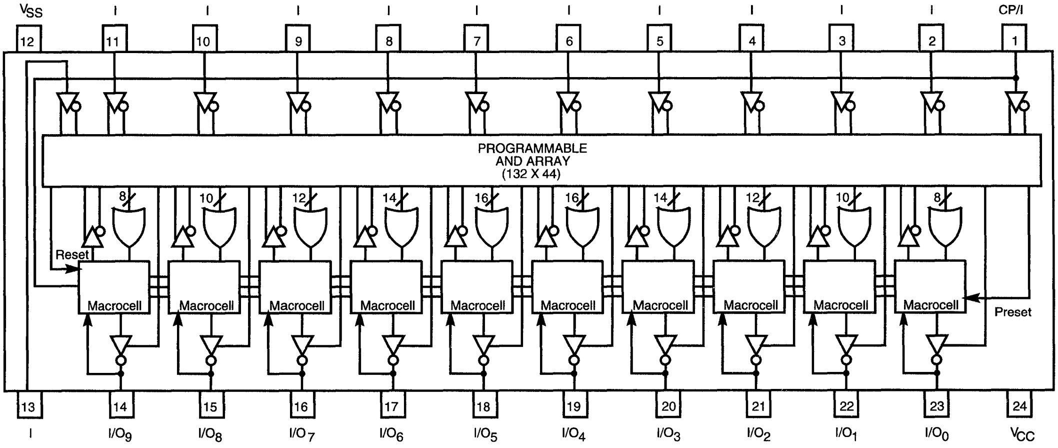
Additional features of the Cypress PALC22V10D include a synchronous preset and an asynchronous reset product term. These product terms are common to all macrocells, eliminating the need to dedicate standard product terms for initialization functions. The device automatically resets upon power-up.

The PALC22V10D, featuring programmable macrocells and variable product terms, provides a device with the flexibility to implement logic functions in the 500- to 800-gate-array complexity. Since each of the 10 output pins may be individually configured as inputs on a temporary or permanent basis, functions requiring up to 21 inputs and only a single output and down to 12 inputs and 10 outputs are possible. The 10 potential outputs are enabled using product terms. Any output pin may be permanently selected as an output or arbitrarily enabled as an output and an input through the selective use of individual product terms associated with each output. Each of these outputs is achieved through an individual programmable macrocell. These macrocells are programmable to provide a combinatorial or registered inverting or non-inverting output. In a registered mode of operation, the output of the register is fed back into the array, providing current status information to the array. This information is available for establishing the next result in applications such as control state machines. In a combinatorial configuration, the combinatorial output or, if the output is disabled, the signal present on the I/O pin is made available to the array. The flexibility provided by both programmable product term control of the outputs and variable product terms allows a significant gain in functional density through the use of programmable logic.

Along with this increase in functional density, the Cypress PALC22V10D provides lower-power operation through the use of CMOS technology, and increased testability with Flash reprogrammability.

PAL is a registered trademark of Advanced Micro Devices

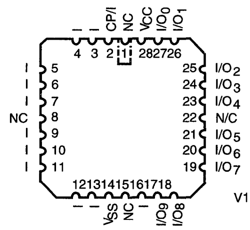
## Logic Block Diagram (PDIP/CDIP)



V10D–1

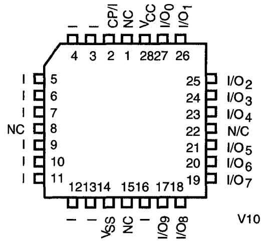
## Pin Configuration

LCC
Top View



V10D–2

PLCC
Top View



V10D–3

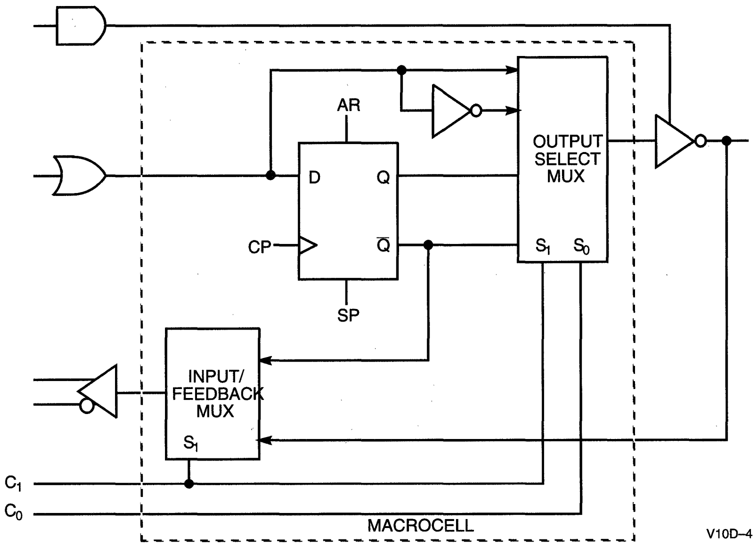
## Configuration Table

| Registered/Combinatorial | | |
|---|---|---|
| $C_1$ | $C_0$ | Configuration |
| 0 | 0 | Registered/Active LOW |
| 0 | 1 | Registered/Active HIGH |

## Configuration Table

| Registered/Combinatorial | | |
|---|---|---|
| $C_1$ | $C_0$ | Configuration |
| 1 | 0 | Combinatorial/Active LOW |
| 1 | 1 | Combinatorial/Active HIGH |

## Macrocell



## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ....................................–65°C to +150°C

Ambient Temperature with
Power Applied.................................................–55°C to +125°C

Supply Voltage to Ground Potential
(Pin 24 to Pin 12)...............................................–0.5V to +7.0V

DC Voltage Applied to Outputs
in High Z State ...................................................–0.5V to +7.0V

DC Input Voltage............................................. -0.5V to +7.0V

Output Current into Outputs (LOW) .............................16 mA

DC Programming Voltage ............................................12.5V

Latch-Up Current.....................................................>200 mA

Static Discharge Voltage
(per MIL-STD-883, Method 3015) .............................>2001V

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ±5% |
| Military[1] | –55°C to +125°C | 5V ±10% |
| Industrial | –40°C to +85°C | 5V ±10% |

**Note:**

[1] $T_A$ is the "instant on" case temperature.

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.,<br>$V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OH}$ = –3.2 mA | Com'l | 2.4 | | V |
| | | | $I_{OH}$ = –2 mA | Mil/Ind | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.,<br>$V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 16 mA | Com'l | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | Mil/Ind | | | |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[3] | | | 2.0 | | V |
| $V_{IL}$[4] | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[3] | | | –0.5 | 0.8 | V |
| $I_{IX}$ | Input Leakage Current | $V_{SS} \le V_{IN} \le V_{CC}$, $V_{CC}$ = Max. | | | –10 | 10 | μA |
| $I_{OZ}$ | Output Leakage Current | $V_{CC}$ = Max., $V_{SS} \le V_{OUT} \le V_{CC}$ | | | –40 | 40 | μA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[5,6] | | | –30 | -90 | mA |
| $I_{CC1}$ | Standby Power Supply Current | $V_{CC}$ = Max.,<br>$V_{IN}$ = GND,<br>Outputs Open in Unprogrammed Device | 10, 15, 25 ns | Com'l | | 90 | mA |
| | | | 7.5 ns | Com'l | | 130 | mA |
| | | | 15, 25 ns | Mil/Ind | | 120 | mA |
| | | | 10 ns | Mil/Ind | | 120 | mA |
| $I_{CC2}$[6] | Operating Power Supply Current | $V_{CC}$ = Max.,<br>$V_{IL}$ = 0V, $V_{IH}$ = 3V,<br>Output Open, Device Programmed as a 10-Bit Counter,<br>f = 25 MHz | 10, 15, 25 ns | Com'l | | 110 | mA |
| | | | 7.5 ns | Com'l | | 140 | mA |
| | | | 15, 25 ns | Mil/Ind | | 130 | mA |
| | | | 10 ns | Mil/Ind | | 130 | mA |

## Capacitance[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0V @ f = 1 MHz | | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V @ f = 1 MHz | | 10 | pF |

## Endurance Characteristics[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | | Cycles |

**Notes:**

[2] See the last page of this specification for Group A subgroup testing information.

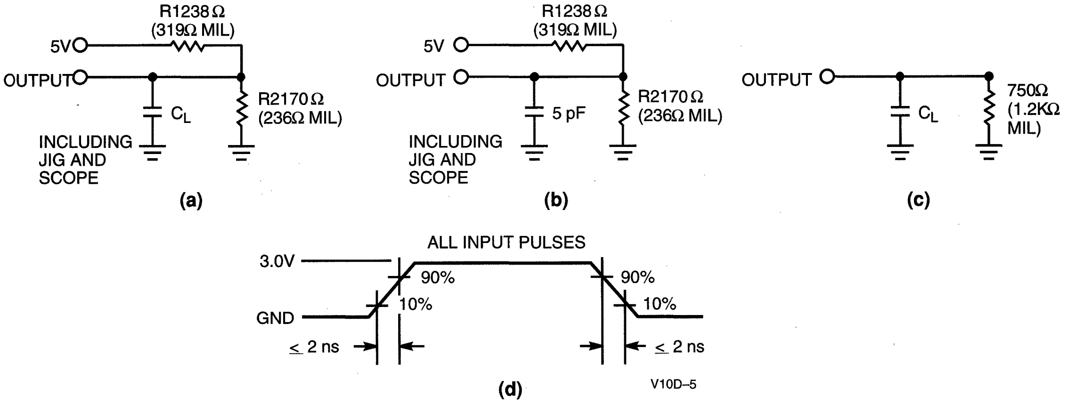
[3] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[4] $V_{IL}$ (Min.) is equal to -3.0V for pulse durations less than 20 ns.

[5] Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

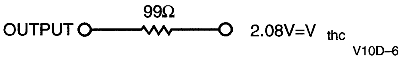
[6] Tested initially and after any design or process changes that may affect these parameters.

## AC Test Loads and Waveforms

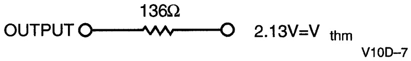


(a) (b) (c) (d)

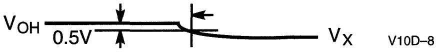
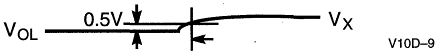
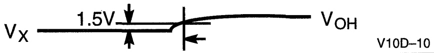
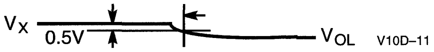
Equivalent to: THÉVENIN EQUIVALENT(Commercial)



Equivalent to: THÉVENIN EQUIVALENT(Military)



| Load Speed | $C_L$ | Package |
|---|---|---|
| 7.5, 10, 15, 25 ns | 50 pF | PDIP, CDIP, PLCC, LCC |

| Parameter | $V_X$ | Output Waveform Measurement Level |
|---|---|---|
| $t_{ER}$ (–) | 1.5V |  |
| $t_{ER}$ (+) | 2.6V |  |
| $t_{EA}$ (+) | 0V |  |
| $t_{EA}$ (–) | $V_{thc}$ |  |

**(e) Test Waveforms**

## Commercial Switching Characteristics PALC22V10D[2, 7]

| Parameter | Description | 22V10D-7 | | 22V10D-10 | | 22V10D-15 | | 22V10D-25 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8, 9] | 3 | 7.5 | 3 | 10 | 3 | 15 | 3 | 25 | ns |
| $t_{EA}$ | Input to Output Enable Delay[10] | | 8 | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[11] | | 8 | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8, 9] | 2 | 5 | 2 | 7 | 2 | 8 | 2 | 15 | ns |
| $t_{S1}$ | Input or Feedback Set-Up Time | 5 | | 6 | | 10 | | 15 | | ns |
| $t_{S2}$ | Synchronous Preset Set-Up Time | 6 | | 7 | | 10 | | 15 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 10 | | 12 | | 20 | | 30 | | ns |
| $t_{WH}$ | Clock Width HIGH[6] | 3 | | 3 | | 6 | | 13 | | ns |
| $t_{WL}$ | Clock Width LOW[6] | 3 | | 3 | | 6 | | 13 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$)[12] | 100 | | 76.9 | | 55.5 | | 33.3 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[6, 13] | 166 | | 142 | | 83.3 | | 35.7 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[6,14] | 133 | | 111 | | 68.9 | | 38.5 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[6, 15] | | 2.5 | | 3 | | 4.5 | | 13 | ns |
| $t_{AW}$ | Asynchronous Reset Width | 8 | | 10 | | 15 | | 25 | | ns |
| $t_{AR}$ | Asynchronous Reset Recovery Time | 5 | | 6 | | 10 | | 25 | | ns |
| $t_{AP}$ | Asynchronous Reset to Registered Output Delay | | 12 | | 13 | | 20 | | 25 | ns |
| $t_{SPR}$ | Synchronous Preset Recovery Time | 6 | | 8 | | 10 | | 15 | | ns |
| $t_{PR}$ | Power-Up Reset Time[6,16] | 1 | | 1 | | 1 | | 1 | | µs |

**Notes:**

[7] Part (a) of AC Test Loads and Waveforms is used for all parameters except $t_{ER}$ and $t_{EA(+)}$. Part (b) of AC Test Loads and Waveforms is used for $t_{ER}$. Part (c) of AC Test Loads and Waveforms is used for $t_{EA(+)}$.

[8] Min. times are tested initially and after any design or process changes that may affect these parameters.

[9] This specification is guaranteed for all device outputs changing state in a given access cycle.

[10] The test load of part (a) of AC Test Loads and Waveforms is used for measuring $t_{EA(-)}$. The test load of part (c) of AC Test Loads and Waveforms is used for measuring $t_{EA(+)}$ only. Please see part (e) of AC Test Loads and Waveforms for enable and disable test waveforms and measurement reference levels.

[11] This parameter is measured as the time after output disable input that the previous output data state remains stable on the output. This delay is measured to the point at which a previous HIGH level has fallen to 0.5 volts below $V_{OH}$ min. or a previous LOW level has risen to 0.5 volts above $V_{OL}$ max. Please see part (e) of AC Test Loads and Waveforms for enable and disable test waveforms and measurement reference levels.

[12] This specification indicates the guaranteed maximum frequency at which a state machine configuration with external feedback can operate.

[13] This specification indicates the guaranteed maximum frequency at which the device can operate in data path mode.

[14] This specification indicates the guaranteed maximum frequency at which a state machine configuration with internal only feedback can operate.

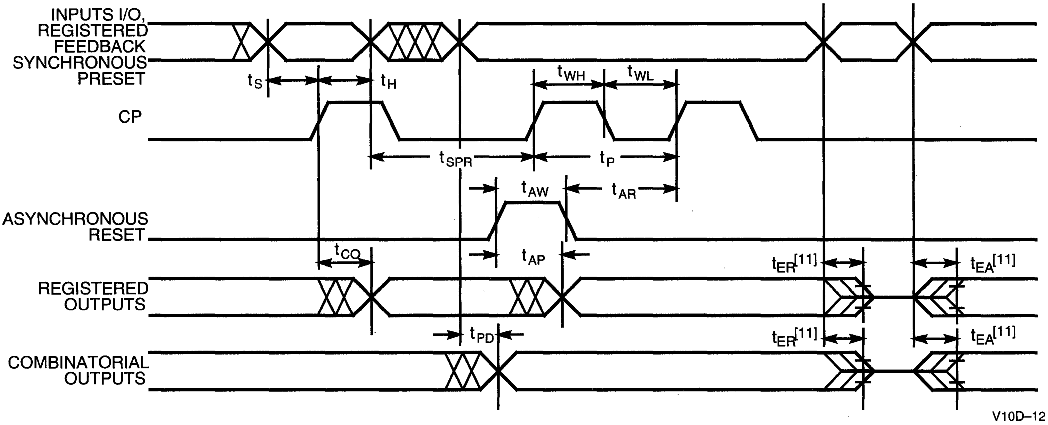
[15] This parameter is calculated from the clock period at $f_{MAX}$ internal ($1/f_{MAX3}$) as measured (see Note  above) minus $t_S$.

[16] The registers in the PALC22V10D have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to a logic LOW state. The output state will depend on the polarity of the output buffer. This feature is useful in establishing state machine initialization. To insure proper operation, the rise in $V_{CC}$ must be monotonic and the timing constraints depicted in Power-Up Reset Waveform must be satisfied.

## Military and Industrial Switching Characteristics PALC22V10D[2, 7]

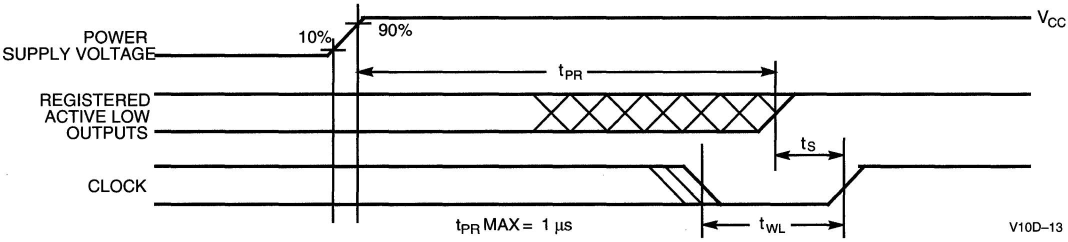
| Parameter | Description | 22V10D-10 | | 22V10D-15 | | 22V10D-25 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[8, 9] | 3 | 10 | 3 | 15 | 3 | 25 | ns |
| $t_{EA}$ | Input to Output Enable Delay[10] | | 10 | | 15 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[11] | | 10 | | 15 | | 25 | ns |
| $t_{CO}$ | Clock to Output Delay[8, 9] | 2 | 7 | 2 | 8 | 2 | 15 | ns |
| $t_{S1}$ | Input or Feedback Set-Up Time | 6 | | 10 | | 18 | | ns |
| $t_{S2}$ | Synchronous Preset Set-Up Time | 7 | | 10 | | 18 | | ns |
| $t_H$ | Input Hold Time | 0 | | 0 | | 0 | | ns |
| $t_P$ | External Clock Period ($t_{CO} + t_S$) | 12 | | 20 | | 33 | | ns |
| $t_{WH}$ | Clock Width HIGH[6] | 3 | | 6 | | 14 | | ns |
| $t_{WL}$ | Clock Width LOW[6] | 3 | | 6 | | 14 | | ns |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO} + t_S)$)[12] | 76.9 | | 50.0 | | 30.3 | | MHz |
| $f_{MAX2}$ | Data Path Maximum Frequency ($1/(t_{WH} + t_{WL})$)[6, 13] | 142 | | 83.3 | | 35.7 | | MHz |
| $f_{MAX3}$ | Internal Feedback Maximum Frequency ($1/(t_{CF} + t_S)$)[6,14] | 111 | | 68.9 | | 32.2 | | MHz |
| $t_{CF}$ | Register Clock to Feedback Input[6,15] | | 3 | | 4.5 | | 13 | ns |
| $t_{AW}$ | Asynchronous Reset Width | 10 | | 15 | | 25 | | ns |
| $t_{AR}$ | Asynchronous Reset Recovery Time | 6 | | 12 | | 25 | | ns |
| $t_{AP}$ | Asynchronous Reset to Registered Output Delay | | 12 | | 20 | | 25 | ns |
| $t_{SPR}$ | Synchronous Preset Recovery Time | 8 | | 20 | | 25 | | ns |
| $t_{PR}$ | Power-Up Reset Time[6, 16] | 1 | | 1 | | 1 | | μs |

## Switching Waveform



V10D–12

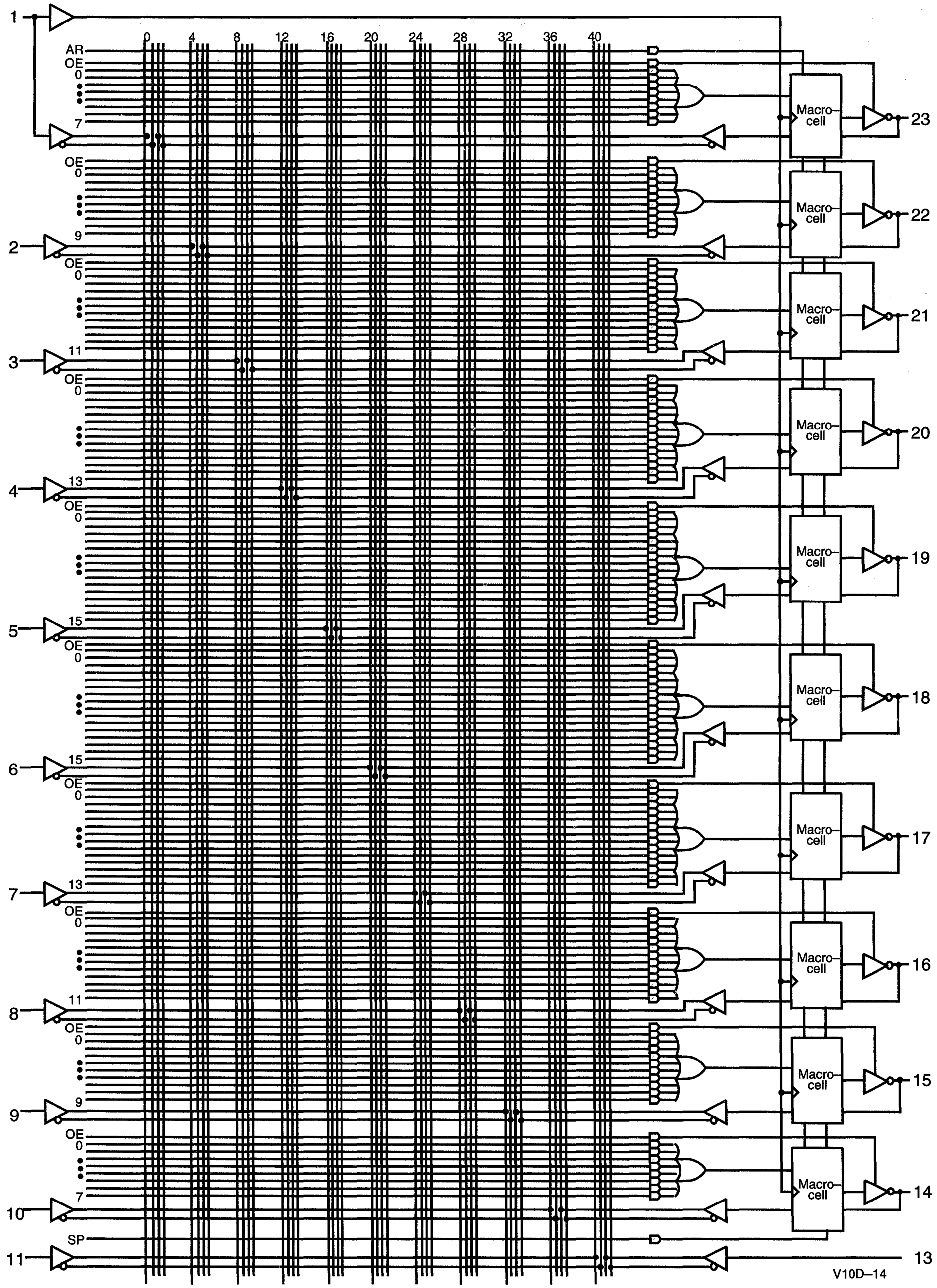
## Power-Up Reset Waveform[16]



$t_{PR}$ MAX = 1 µs

V10D–13

## Functional Logic Diagram for PALC22V10D



## Ordering Information

| $I_{CC}$ (mA) | $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 130 | 7.5 | 5 | 5 | PALC22V10D-7JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALC22V10D-7PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 90 | 10 | 6 | 7 | PALC22V10D-10JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALC22V10D-10PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 150 | 10 | 6 | 7 | PALC22V10D-10JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALC22V10D-10PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| 150 | 10 | 6 | 7 | PALC22V10D-10DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALC22V10D-10KMB | K73 | 24-Lead Rectangular Cerpack | |
| | | | | PALC22V10D-10LMB | L64 | 28-Square Leadless Chip Carrier | |
| 90 | 15 | 7.5 | 10 | PALC22V10D-15JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALC22V10D-15PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 15 | 7.5 | 10 | PALC22V10D-15JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALC22V10D-15PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 15 | 7.5 | 10 | PALC22V10D-15DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALC22V10D-15KMB | K73 | 24-Lead Rectangular Cerpack | |
| | | | | PALC22V10D-15LMB | L64 | 28-Square Leadless Chip Carrier | |
| 90 | 25 | 15 | 15 | PALC22V10D-25JC | J64 | 28-Lead Plastic Leaded Chip Carrier | Commercial |
| | | | | PALC22V10D-25PC | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 25 | 15 | 15 | PALC22V10D-25JI | J64 | 28-Lead Plastic Leaded Chip Carrier | Industrial |
| | | | | PALC22V10D-25PI | P13 | 24-Lead (300-Mil) Molded DIP | |
| 120 | 25 | 15 | 15 | PALC22V10D-25DMB | D14 | 24-Lead (300-Mil) CerDIP | Military |
| | | | | PALC22V10D-25KMB | K73 | 24-Lead Rectangular Cerpack | |
| | | | | PALC22V10D-25LMB | L64 | 28-Square Leadless Chip Carrier | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_S$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |

Document #: 38-00185-H

CYPRESS

# CY7C330

# CMOS Programmable Synchronous State Machine

## Features

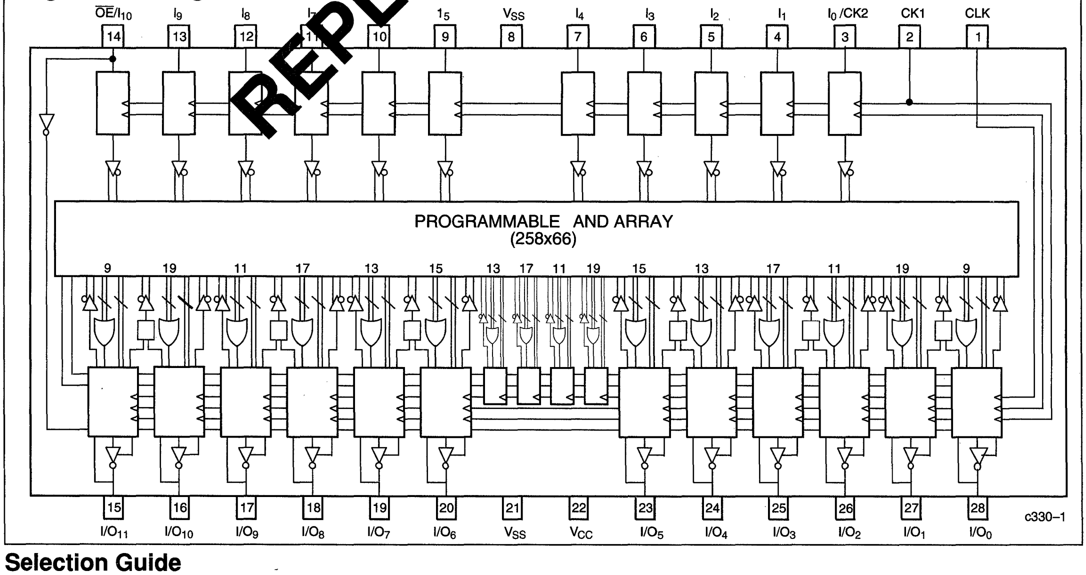
- **Twelve I/O macrocells each having:**
  - **—registered, three-state I/O pins**
  - **—input register clock select multiplexer**
  - **—feed back multiplexer**
  - **—output enable (OE) multiplexer**
- **All twelve macrocell state registers can be hidden**
- **User-configurable state registers—JK, RS, T, or D**
- **One input multiplexer per pair of I/O macrocells allows I/O pin associated with a hidden macrocell state register to be saved for use as an input**
- **Four dedicated hidden registers**
- **Eleven dedicated, registered inputs**
- **Three separate clocks—two inputs, one output**
- **Common (pin 14-controlled) or product term-controlled output enable for each I/O pin**
- **256 product terms—32 per pair of macrocells, variable distribution**
- **Global, synchronous, product term-controlled, state register set and reset—inputs to product term are clocked by input clock**
- **66-MHz operation**
  - **—3-ns input set-up and 12-ns clock to output**
  - **—15-ns input register clock to state register clock**
- **Low power**
  - **—130 mA $I_{CC}$**
- **28-pin, 300-mil DIP, LCC**
- **Erasable and reprogrammable**

## Functional Description

The CY7C330 is a high-performance, erasable, programmable, logic device (EPLD) whose architecture has been optimized to enable the user to easily and efficiently construct very high-performance synchronous state machines.

The unique architecture of the CY7C330, consisting of the user-configurable output macrocell, bidirectional I/O capability, input registers, and three separate clocks, enables the user to design high-performance state machines that can communicate either with each other or with microprocessors over bidirectional parallel buses of user-definable widths.

## Logic Block Diagram



## Selection Guide

| | | 7C330-66 | 7C330-50 | 7C330-40 | 7C330-33 | 7C330-28 |
|---|---|---|---|---|---|---|
| Maximum Operating Frequency, $f_{MAX}$ (MHz) | Commercial | 66.6 | 50.0 | | 33.3 | |
| | Military | | 50.0 | 40.0 | | 28.5 |
| Power Supply Current $I_{CC1}$ (mA) | Commercial | 140 | 130 | | 130 | |
| | Military | | 160 | 150 | | 150 |

CYPRESS

# CY7C331

# Asynchronous Registered EPLD

## Features

- **Twelve I/O macrocells each having:**
  - **—One state flip-flop with an XOR sum-of-products input**
  - **—One feedback flip-flop with input coming from the I/O pin**
  - **—Independent (product term) set, reset, and clock inputs on all registers**
  - **—Asynchronous bypass capability on all registers under product term control (r = s = 1)**
  - **—Global or local output enable on three-state I/O**
  - **—Feedback from either register to the array**
- **192 product terms with variable distribution to macrocells**
- **13 inputs, 12 feedback I/O pins, plus 6 shared I/O macrocell feedbacks for a total of 31 true and complementary inputs**
- **High speed: 20 ns maximum $t_{PD}$**
- **Security bit**
- **Space-saving 28-pin slim-line DIP package; also available in 28-pin PLCC**
- **Low power**
  - **—90 mA typical $I_{CC}$ quiescent**
  - **—180 mA $I_{CC}$ maximum**
  - **—UV-erasable and reprogrammable**
  - **—Programming and operation 100% testable**

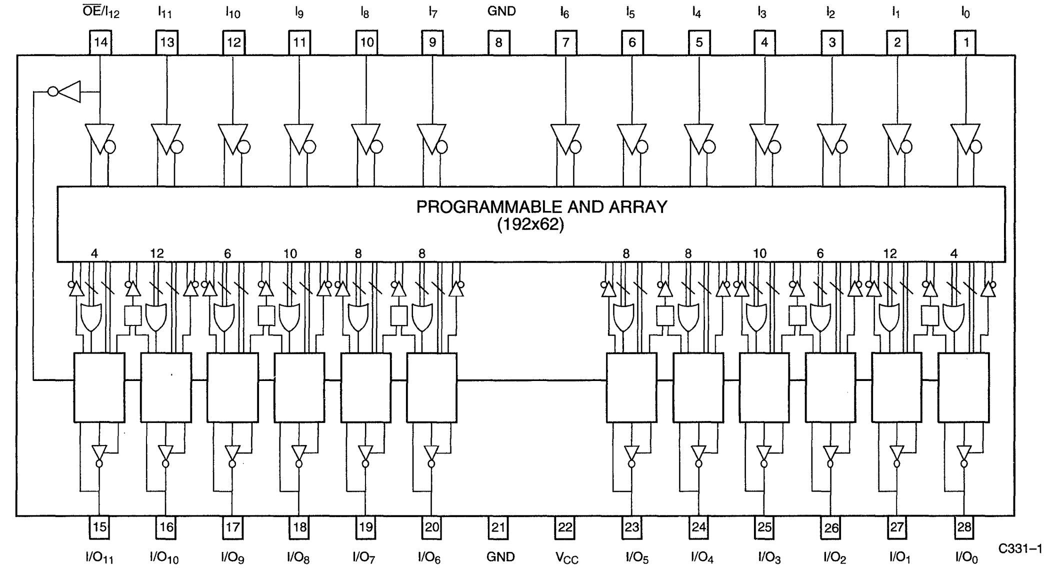
## Functional Description

The CY7C331 is the most versatile PLD available for asynchronous designs. Central resources include twelve full D-type flip-flops with separate set, reset, and clock capability. For increased utility, XOR gates are provided at the D-inputs and the product term allocation per flip-flop is variably distributed.

## I/O Resources

Pins 1 through 7 and 9 through 14 serve as array inputs; pin 14 may also be used as a global output enable for the I/O macrocell three-state outputs. Pins 15 through 20 and 23 through 28 are connected to I/O macrocells and may be managed as inputs or outputs depending on the configuration and the macrocell OE terms.

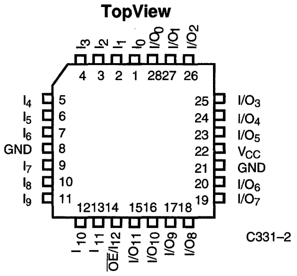
## Logic Block Diagram



## Selection Guide

| Generic Part Number | $I_{CC1}$ (mA) | | $t_{PD}$ (ns) | | $t_S$ (ns) | | $t_{CO}$ (ns) | |
|---|---|---|---|---|---|---|---|---|
| | Com'l | Mil | Com'l | Mil | Com'l | Mil | Com'l | Mil |
| CY7C331–20 | 130 | | 20 | | 12 | | 20 | |
| CY7C331–25 | 120 | 160 | 25 | 25 | 12 | 15 | 25 | 25 |
| CY7C331–30 | | 150 | | 30 | | 15 | | 30 |
| CY7C331–40 | | 150 | | 40 | | 20 | | 40 |

## Pin Configuration



## I/O Resources (continued)

It should be noted that there are two ground connections (pins 8 and 21) which, together with $V_{CC}$ (pin 22) are located centrally on the package. The reason for this placement and dual-ground structure is to minimize the ground-loop noise when the outputs are driving simultaneously into a heavy capacitive load.

The CY7C331 has twelve I/O macrocells (see *Figure 1*). Each macrocell has two D-type flip-flops. One is fed from the array, and one from the I/O pin. For each flip-flop there are three dedicated product terms driving the R, S, and clock inputs, respectively. Each macrocell has one input to the array and for each pair of macrocells there is one shared input to the array. The macrocell input to the array may be configured to come from the 'Q' output of either flip-flop.

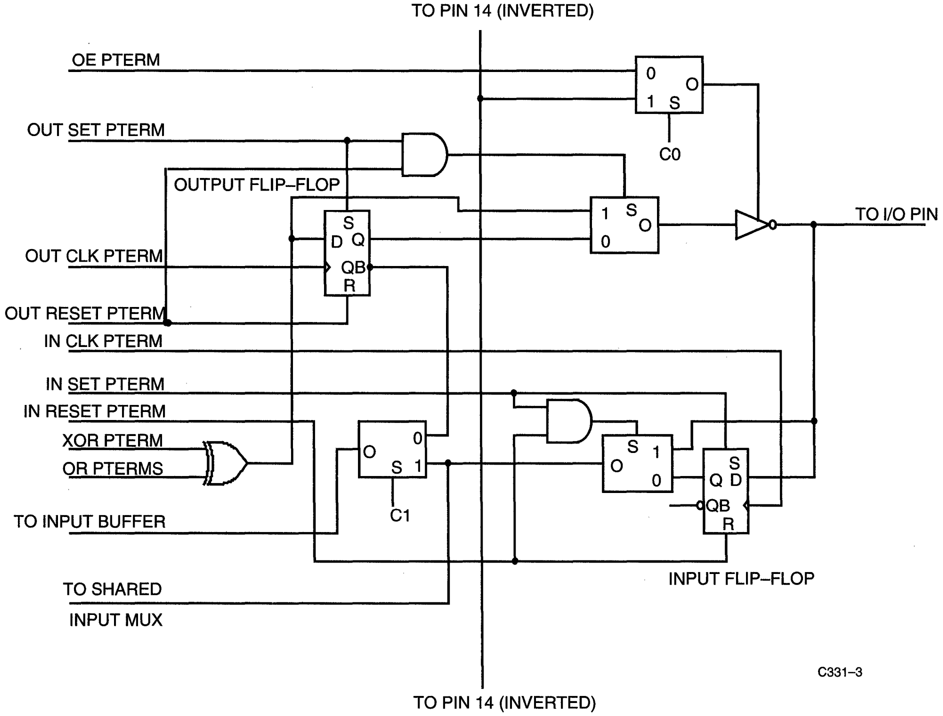


**Figure 1. I/O Macrocell**

## I/O Resources (continued)

The D-type flip-flop that is fed from the array (i.e., the state flip-flop) has a logical XOR function on its input that combines a single product term with a sum(OR) of a number of product terms. The single product term is used to set the polarity of the output or to implement toggling (by including the current output in the product term).

The R and S inputs to the flip-flops override the current setting of the 'Q' output. The S input sets 'Q' true and the R input resets 'Q' (sets it false). If both R and S are asserted (true) at once, then the output will follow the input ('Q' = 'D') (see *Table 1*).

**Table 1. RS Truth Table**

| R | S | Q |
|---|---|---|
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 1 | D |

## Shared Input Multiplexer

The input associated with each pair of macrocells may be configured by the shared input multiplexer to come from either macrocell; the 'Q' output of the flip-flop coming from the I/O pin is used as the input signal source (see *Figure 2*).

## Product Term Distribution

The product terms are distributed to the macrocells such that 32 product terms are distributed between two adjacent macrocells.

The pairing of macrocells is the same as it is for the shared inputs. Eight of the product terms are used in each macrocell for set, reset, clock, output enable, and the upper part of the XOR gate. This leaves 16 product terms per pair of macrocells to be divided between the sum-of-products inputs to the two state registers. The following table shows the I/O pin pairing for shared inputs, and the product term (PT) allocation to macrocells associated with the I/O pins (see *Table 2*).

**Table 2. Product Term Distribution**

| Macrocell | Pin Number | Product Terms |
|---|---|---|
| 0 | 28 | 4 |
| 1 | 27 | 12 |
| 2 | 26 | 6 |
| 3 | 25 | 10 |
| 4 | 24 | 8 |
| 5 | 23 | 8 |
| 6 | 20 | 8 |
| 7 | 19 | 8 |
| 8 | 18 | 10 |
| 9 | 17 | 6 |
| 10 | 16 | 12 |
| 11 | 15 | 4 |

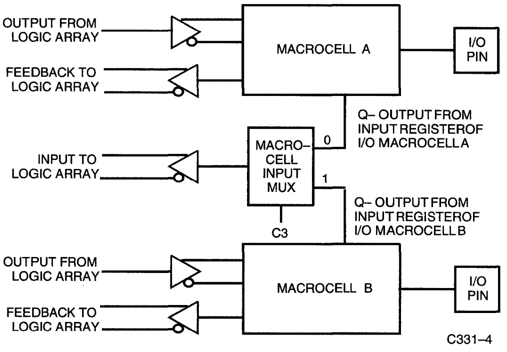


**Figure 2. Shared Input Multiplexer**

The CY7C331 is configured by three arrays of configuration bits (C0, C1, C2). For each macrocell, there is one C0 bit and one C1 bit. For each pair of macrocells there is one C2 bit.

There are twelve C0 bits, one for each macrocell. If C0 is programmed for a macrocell, then the three-state enable (OE) will be controlled by pin 14 (the global OE). If C0 is not programmed, then the OE product term for that macrocell will be used.

There are twelve C1 bits, one for each macrocell. The C1 bit selects inputs for the product term (PT) array from either the state register (if the bit is unprogrammed) or the input register (if the bit is programmed).

There are six C2 bits, providing one C2 bit for each pair of macrocells. The C2 bit controls the shared input multiplexer; if the C2 bit is not programmed, then the input to the product term array comes from the upper macrocell (A). If the C2 bit is programmed, then the input comes from the lower macrocell (B).

The timing diagrams for the CY7C331 cover state register, input register, and various combinational delays. Since internal clocks are the outputs of product terms, all timing is from the transition of the inputs causing the clock transition.

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................. –65°C to +150°C

Ambient Temperature with
Power Applied ............................................. –55°C to +125°C

Supply Voltage to Ground Potential
(Pin 28 to Pin 8 or 21) ................................... –0.5V to +7.0V

DC Input Voltage ........................................... –3.0V to +7.0V

Output Current into Outputs (LOW) ............................. 12 mA

Static Discharge Voltage ........................................... >1500V
(per MIL-STD-883, Method 3015)

Latch-Up Current ..................................................... >200 mA

DC Programming Voltage ........................................... 13.0 V

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 10% |
| Military[1] | –55°C to +125°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$<br>$I_{OH}$ = –3.2 mA (Com'l), $I_{OH}$ = –2 mA (Mil) | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$<br>$I_{OL}$ = 12 mA (Com'l), $I_{OL}$ = 8 mA (Mil) | | | 0.5 | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed HIGH Input, all Inputs[3] | | 2.2 | | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed LOW Input, all Inputs[3] | | | 0.8 | V |
| $I_{IX}$ | Input Leakage Current | $V_{SS}$ < $V_{IN}$ < $V_{CC}$, $V_{CC}$ = Max. | | –10 | +10 | μA |
| $I_{OZ}$ | Output Leakage Current | $V_{SS}$ < $V_{OUT}$ < $V_{CC}$, $V_{CC}$ = Max. | | –40 | +40 | μA |
| $I_{SC}$ | Output Short Circuit Current[4] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[5] | | –30 | –90 | mA |
| $I_{CC1}$ | Standby Power Supply Current | $V_{CC}$ = Max., $V_{IN}$ = GND, Outputs Open | Com'l –20 | | 130 | mA |
| | | | Com'l –25 | | 120 | |
| | | | Mil –25 | | 160 | mA |
| | | | Mil –30, –40 | | 150 | |
| $I_{CC2}$ | Power Supply Current at Frequency[4, 6] | $V_{CC}$ = Max., Outputs Disabled (in High Z State) Device Operating at $f_{MAX}$ External ($f_{MAX1}$) | Com'l | | 180 | mA |
| | | | Mil | | 200 | |

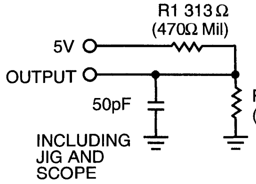
## Capacitance[4]

| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0V at f = 1 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V at f = 1 MHz | 10 | pF |

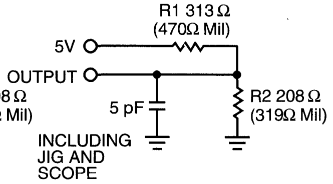
**Notes:**

1. $T_A$ is the "instant on" case temperature.
2. See the last page of this specification for Group A subgroup testing information.
3. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
4. Tested initially and after any design or process changes that may affect these parameters.
5. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
6. Because these input signals are controlled by product terms, active input polarity may be of either polarity. Internal active input polarity has been shown for clarity.

## AC Test Loads and Waveforms

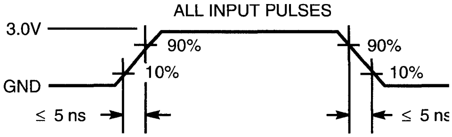


(a)



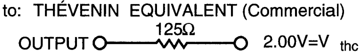
(b)

C331-5



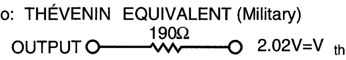
C331-6

Equivalent to: THÉVENIN EQUIVALENT (Commercial)

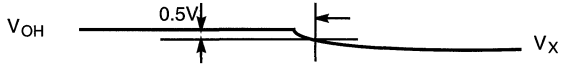
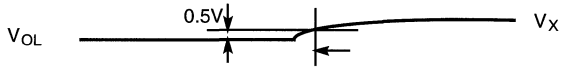
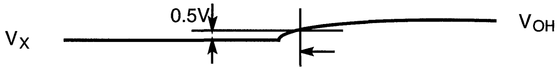
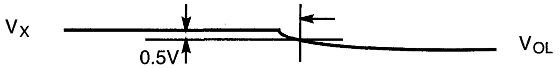
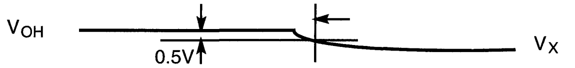
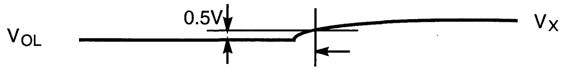
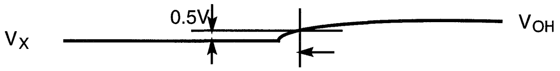
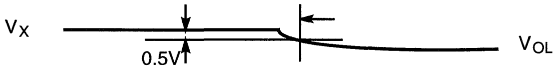


C331-7

Equivalent to: THÉVENIN EQUIVALENT (Military)



C331-8

| Parameter | $V_X$ | Output Waveform—Measurement Level |
|---|---|---|
| $t_{PXZ(-)}$ | 1.5V |  C331-9 |
| $t_{PXZ(+)}$ | 2.6V |  C331-10 |
| $t_{PZX(+)}$ | $V_{thc}$ |  C331-11 |
| $t_{PZX(-)}$ | $V_{thc}$ |  C331-12 |
| $t_{ER(-)}$ | 1.5V |  C331-13 |
| $t_{ER(+)}$ | 2.6V |  C331-14 |
| $t_{EA(+)}$ | $V_{thc}$ |  C331-15 |
| $t_{EA(-)}$ | $V_{thc}$ |  C331-16 |

(c) Test Waveforms and Measurement Levels

## Switching Characteristics Over the Operating Range[2]

| Parameter | Description | Commercial | | | | Unit |
|---|---|---|---|---|---|---|
| | | –20 | | –25 | | |
| | | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input to Output Propagation Delay[7] | | 20 | | 25 | ns |
| $t_{ICO}$ | Input Register Clock to Output Delay[8] | | 35 | | 40 | ns |
| $t_{IOH}$ | Output Data Stable Time from Input Clock[8] | 5 | | 5 | | ns |
| $t_{IS}$ | Input or Feedback Set-Up Time to Input Register Clock[8] | 2 | | 2 | | ns |
| $t_{IH}$ | Input Register Hold Time from Input Clock[8] | 11 | | 13 | | ns |
| $t_{IAR}$ | Input to Input Register Asynchronous Reset Delay[8] | | 35 | | 40 | ns |

## Switching Characteristics Over the Operating Range[2] (continued)

| | | Commercial | | | | |
|---|---|---|---|---|---|---|
| | | –20 | | –25 | | |
| Parameter | Description | Min. | Max. | Min. | Max. | Unit |
| $t_{IRW}$ | Input Register Reset Width[4, 8] | 35 | | 40 | | ns |
| $t_{IRR}$ | Input Register Reset Recovery Time[4, 8] | 35 | | 40 | | ns |
| $t_{IAS}$ | Input to Input Register Asynchronous Set Delay[8] | | 35 | | 40 | ns |
| $t_{ISW}$ | Input Register Set Width[4, 8] | 35 | | 40 | | ns |
| $t_{ISR}$ | Input Register Set Recovery Time[4, 8] | 35 | | 40 | | ns |
| $t_{WH}$ | Input and Output Clock Width HIGH[8, 9, 10] | 12 | | 15 | | ns |
| $t_{WL}$ | Input and Output Clock Width LOW[8, 9, 10] | 12 | | 15 | | ns |
| $f_{MAX1}$ | Maximum Frequency with Feedback in Input Registered Mode ($1/(t_{ICO} + t_{IS})$)[11] | 27.0 | | 23.8 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Input Registered Mode (Lowest of $1/t_{ICO}$, $1/(t_{WH} + t_{WL})$, or $1/(t_{IS} + t_{IH})$)[8] | 28.5 | | 25.0 | | MHz |
| $t_{IOH}$–$t_{IH}$33X | Output Data Stable from Input Clock Minus Input Register Input Hold Time for 7C335[12, 13] | 0 | | 0 | | ns |
| $t_{CO}$ | Output Register Clock to Output Delay[9] | | 20 | | 25 | ns |
| $t_{OH}$ | Output Data Stable Time from Output Clock[9] | 3 | | 3 | | ns |
| $t_S$ | Output Register Input Set-Up Time to Output Clock[9] | 12 | | 12 | | ns |
| $t_H$ | Output Register Input Hold Time from Output Clock[9] | 8 | | 8 | | ns |
| $t_{OAR}$ | Input to Output Register Asynchronous Reset Delay[9] | | 20 | | 25 | ns |
| $t_{ORW}$ | Output Register Reset Width[9] | 20 | | 25 | | ns |
| $t_{ORR}$ | Output Register Reset Recovery Time[9] | 20 | | 25 | | ns |
| $t_{OAS}$ | Input to Output Register Asynchronous Set Delay[9] | | 20 | | 25 | ns |
| $t_{OSW}$ | Output Register Set Width[9] | 20 | | 25 | | ns |
| $t_{OSR}$ | Output Register Set Recovery Time[9] | 20 | | 25 | | ns |
| $t_{EA}$ | Input to Output Enable Delay[14, 15] | | 25 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable Delay[14, 15] | | 25 | | 25 | ns |
| $t_{PZX}$ | Pin 14 to Output Enable Delay[14, 15] | | 20 | | 20 | ns |
| $t_{PXZ}$ | Pin 14 to Output Disable Delay[14, 15] | | 20 | | 20 | ns |
| $f_{MAX3}$ | Maximum Frequency with Feedback in Output Registered Mode ($1/(t_{CO} + t_S)$)[16, 17] | 31.2 | | 27.0 | | MHz |
| $f_{MAX4}$ | Maximum Frequency Data Path in Output Registered Mode (Lowest of $1/t_{CO}$, $1/(t_{WH} + t_{WL})$, or $1/(t_S + t_H)$)[9] | 41.6 | | 33.3 | | MHz |
| $t_{OH}$–$t_{IH}$33X | Output Data Stable from Output Clock Minus Input Register Input Hold Time for 7C335[13, 18] | 0 | | 0 | | ns |
| $f_{MAX5}$ | Maximum Frequency Pipelined Mode[10, 17] | 35.0 | | 30.0 | | MHz |

**Notes:**

[7] Refer to *Figure 3*, configuration 1.

[8] Refer to *Figure 3*, configuration 2.

[9] Refer to *Figure 3*, configuration 3.

[10] Refer to *Figure 3*, configuration 6.

[11] Refer to *Figure 3*, configuration 7.

[12] Refer to *Figure 3*, configuration 9.

[13] This specification is intended to guarantee interface compatibility of the other members of the CY7C330 family with the CY7C331. This specification is met for the devices noted operating at the same ambient temperature and at the same power supply voltage. These parameters are tested periodically by sampling of production product.

[14] Part (a) of AC Test Loads and Waveforms used for all parameters except $t_{PZXI}$, $t_{PXZI}$, $t_{PZX}$, and $t_{PXZ}$, which use part (b). Part (c) shows the test waveforms and measurement levels.

[15] Refer to *Figure 3*, configuration 4.

[16] Refer to *Figure 3*, configuration 8.

[17] This specification is intended to guarantee that a state machine configuration created with internal or external feedback can be operated with output register and input register clocks controlled by the same source. These parameters are tested by periodic sampling of production product.

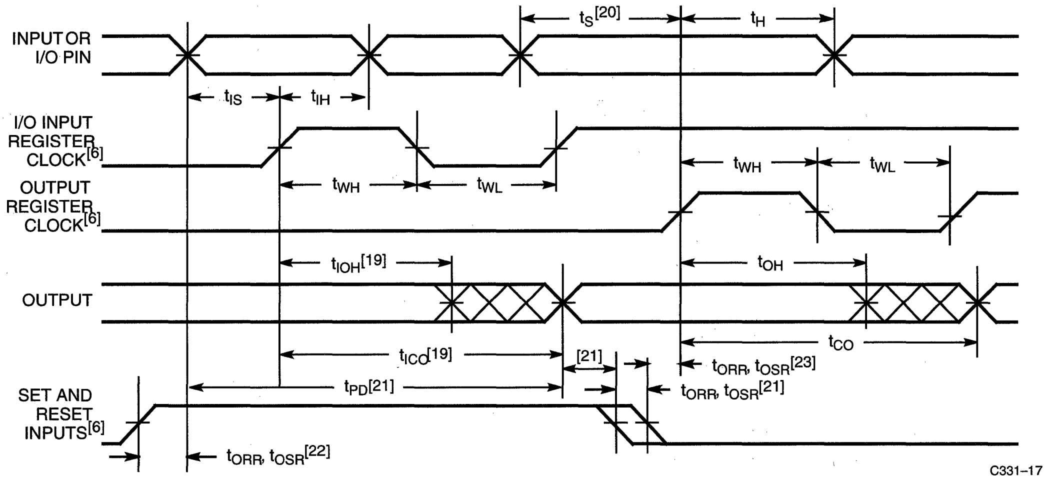
## Switching Characteristics Over the Operating Range[2] (continued)

| | | Military | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | –25 | | –30 | | –40 | | |
| Parameter | Description | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| $t_{PD}$ | Input to Output Propagation Delay[7] | | 25 | | 30 | | 40 | ns |
| $t_{ICO}$ | Input Register Clock to Output Delay[4, 8] | | 45 | | 50 | | 65 | ns |
| $t_{IOH}$ | Output Data Stable Time from Input Clock[4, 8] | 5 | | 5 | | 5 | | ns |
| $t_{IS}$ | Input or Feedback Set-Up Time to Input Register Clock[8] | 5 | | 5 | | 5 | | ns |
| $t_{IH}$ | Input Register Hold Time from Input Clock[4, 8] | 13 | | 15 | | 20 | | ns |
| $t_{IAR}$ | Input to Input Register Asynchronous Reset Delay[4, 8] | | 45 | | 50 | | 65 | ns |
| $t_{IRW}$ | Input Register Reset Width[8] | 45 | | 50 | | 65 | | ns |
| $t_{IRR}$ | Input Register Reset Recovery Time[8] | 45 | | 50 | | 65 | | ns |
| $t_{IAS}$ | Input to Input Register Asynchronous Set Delay[8] | | 45 | | 50 | | 65 | ns |
| $t_{ISW}$ | Input Register Set Width[8] | 45 | | 50 | | 65 | | ns |
| $t_{ISR}$ | Input Register Set Recovery Time[8] | 45 | | 50 | | 65 | | ns |
| $t_{WH}$ | Input and Output Clock Width High[8, 9, 10] | 15 | | 20 | | 25 | | ns |
| $t_{WL}$ | Input and Output Clock Width Low[8, 9, 10] | 15 | | 20 | | 25 | | ns |
| $f_{MAX1}$ | Maximum frequency with Feedback in Input Registered Mode ($1/(t_{ICO} + t_{IS})$)[11] | 20.0 | | 18.1 | | 14.2 | | MHz |
| $f_{MAX2}$ | Maximum frequency Data Path in Input Registered Mode (Lowest of $1/t_{ICO}$, $1/(t_{WH} + t_{WL})$, or $1/(t_{IS} + t_{IH})$[8] | 22.2 | | 20.0 | | 15.3 | | MHz |
| $t_{IOH}$–$t_{IH}$33X | Output Data Stable from Input Clock Minus Input Register Input Hold Time for 7C335[12, 13] | 0 | | 0 | | 0 | | ns |
| $t_{CO}$ | Output Register Clock to Output Delay[9] | | 25 | | 30 | | 40 | ns |
| $t_{OH}$ | Output Data Stable Time from Output Clock[9] | 3 | | 3 | | 3 | | ns |
| $t_S$ | Output Register Input Set-Up Time to Output Clock[9] | 15 | | 15 | | 20 | | ns |
| $t_H$ | Output Register Input Hold Time from Output Clock[9] | 10 | | 10 | | 12 | | ns |
| $t_{OAR}$ | Input to Output Register Asynchronous Reset Delay[9] | | 25 | | 30 | | 40 | ns |
| $t_{ORW}$ | Output Register Reset Width[9] | 25 | | 30 | | 40 | | ns |
| $t_{ORR}$ | Output Register Reset Recovery Time[9] | 25 | | 30 | | 40 | | ns |
| $t_{OAS}$ | Input to Output Register Asynchronous Set Delay[9] | | 25 | | 30 | | 40 | ns |
| $t_{OSW}$ | Output Register Set Width[9] | 25 | | 30 | | 40 | | ns |
| $t_{OSR}$ | Output Register Set Recovery Time[9] | 25 | | 30 | | 40 | | ns |
| $t_{EA}$ | Input to Output Enable Delay[14, 15] | | 25 | | 30 | | 40 | ns |
| $t_{ER}$ | Input to Output Disable Delay[14, 15] | | 25 | | 30 | | 40 | ns |
| $t_{PZX}$ | Pin 14 to Output Enable Delay[14, 15] | | 20 | | 25 | | 35 | ns |
| $t_{PXZ}$ | Pin 14 to Output Disable Delay[14, 15] | | 20 | | 25 | | 35 | ns |
| $f_{MAX3}$ | Maximum Frequency with Feedback in Output Registered Mode )$1/(t_{CO} + t_S)$[16, 17] | 25.0 | | 22.2 | | 16.6 | | MHz |
| $f_{MAX4}$ | Maximum Frequency Data Path in Output Registered Mode (Lowest of $1/t_{CO}$, $1/(t_{WH} + t_{WL})$, or $1/(t_S + t_H)$[9] | 33.3 | | 25.0 | | 20.0 | | MHz |
| $t_{OH}$–$t_{IH}$33X | Output Data Stable from Output Clock Minus Input Register Input Hold Time for 7C335[13, 18] | 0 | | 0 | | 0 | | ns |
| $f_{MAX5}$ | Maximum Frequency Pipelined Mode[10, 17] | 28.0 | | 23.5 | | 18.5 | | MHz |

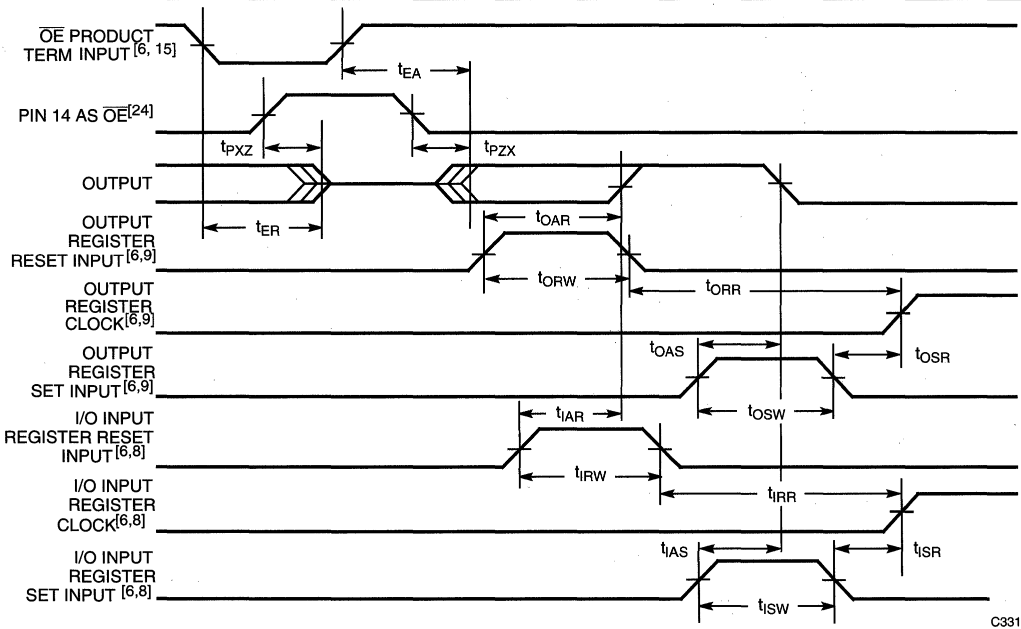
**Note:**

[18] Refer to *Figure 3*, configuration 10.

## Switching Waveforms



C331-17


OE PRODUCT TERM INPUT[6, 15]
PIN 14 AS OE[24]
OUTPUT
OUTPUT REGISTER RESET INPUT[6,9]
OUTPUT REGISTER CLOCK[6,9]
OUTPUT REGISTER SET INPUT[6,9]
I/O INPUT REGISTER RESET INPUT[6,8]
I/O INPUT REGISTER CLOCK[6,8]
I/O INPUT REGISTER SET INPUT[6,8]
tEA
tPXZ
tPZX
tER
tOAR
tORW
tORR
tOAS
tOSR
tOSW
tIAR
tIRW
tIRR
tIAS
tISR
tISW

C331-18

**Notes:**

19. Output register is set in Transparent mode. Output register set and reset inputs are in a HIGH state.
20. Dedicated input or input register set in Transparent mode. Input register set and reset inputs are in a HIGH state.
21. Combinatorial Mode. Reset and set inputs of the input and output registers should remain in a HIGH state at least until the output responds at $t_{PD}$. When returning set and reset inputs to a LOW state, one of these signals should go LOW a minimum of $t_{OSR}$ (set input) or $t_{ORR}$ (reset input) prior to the other. This guarantees predictable register states upon exit from Combinatorial mode.
22. When entering the Combinatorial mode, input and output register set and reset inputs must be stable in a HIGH state a minimum of $t_{ISR}$ or $t_{IRR}$ and $t_{OSR}$ or $t_{ORR}$ respectively prior to application of logic input signals.
23. When returning to the input and/or output Registered mode, register set and reset inputs must be stable in a LOW state a minimum of $t_{ISR}$ or $t_{IRR}$ and $t_{OSR}$ or $t_{ORR}$ respectively prior to the application of the register clock input.
24. Refer to *Figure 3*, configuration 5.

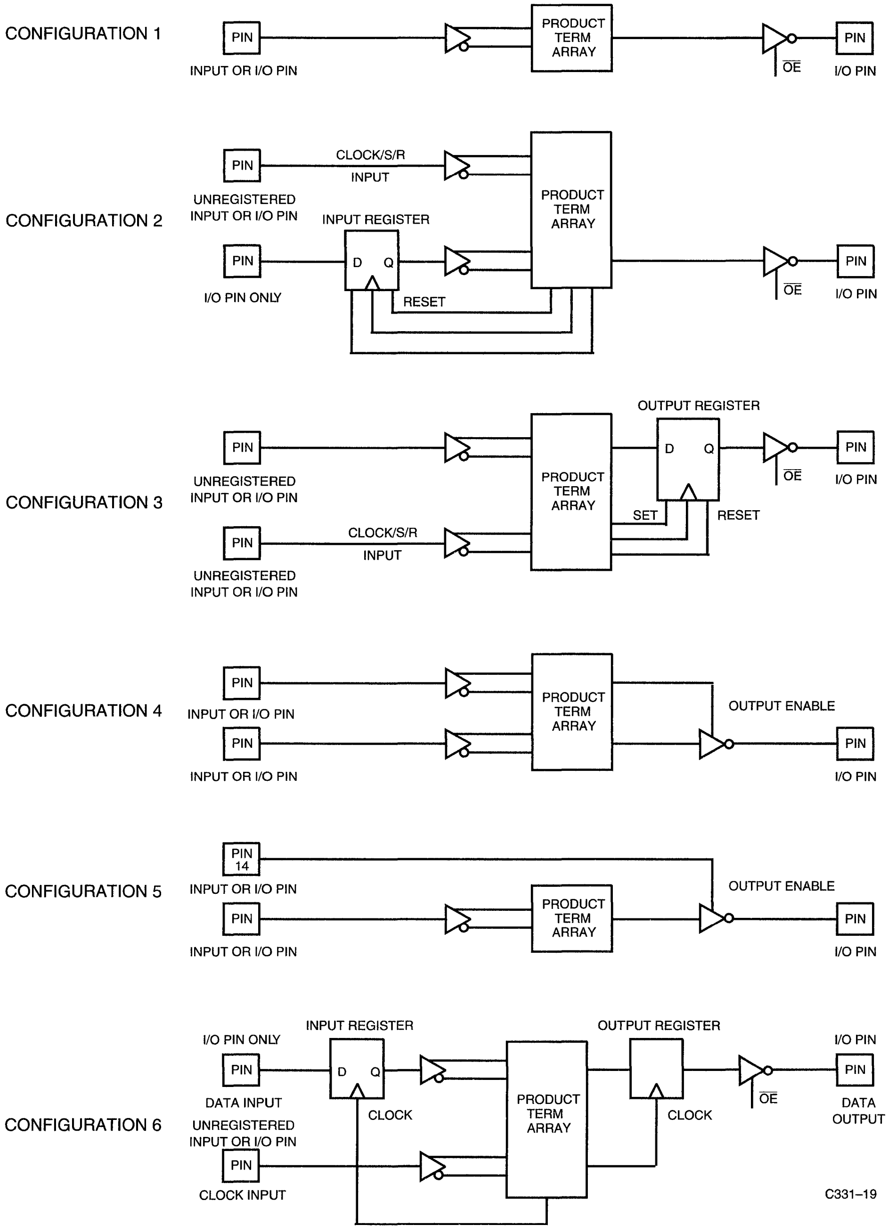


**Figure 3. Timing Configurations**

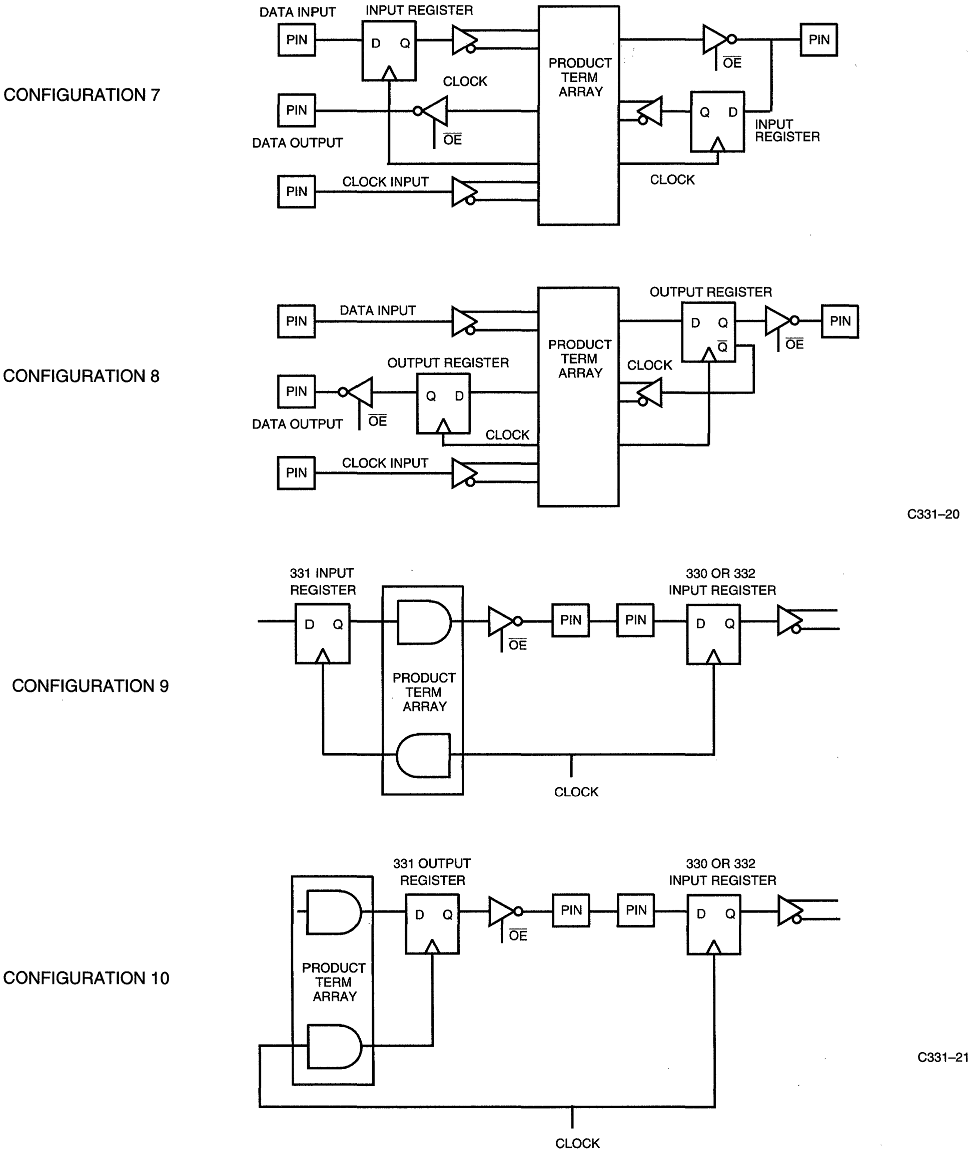
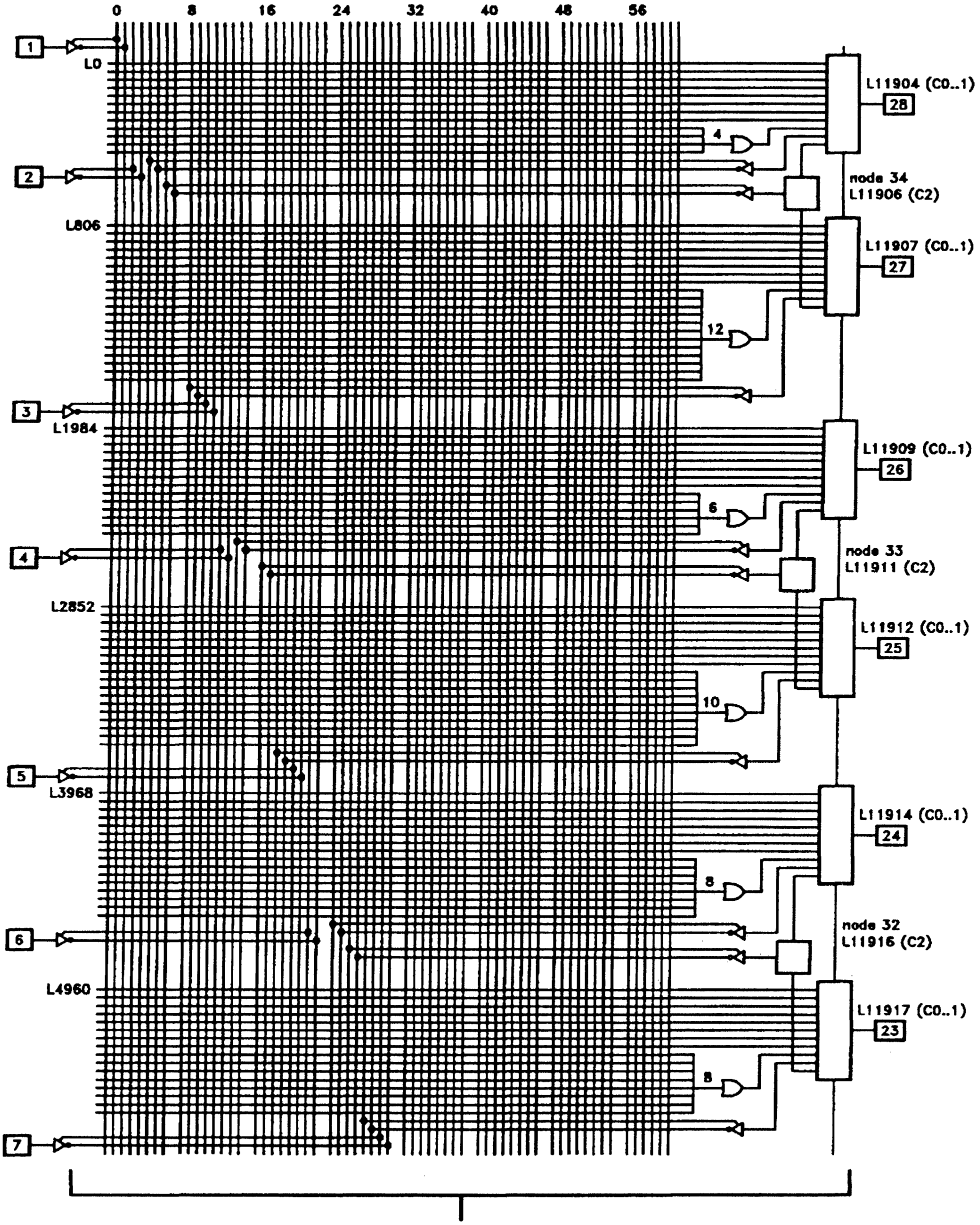


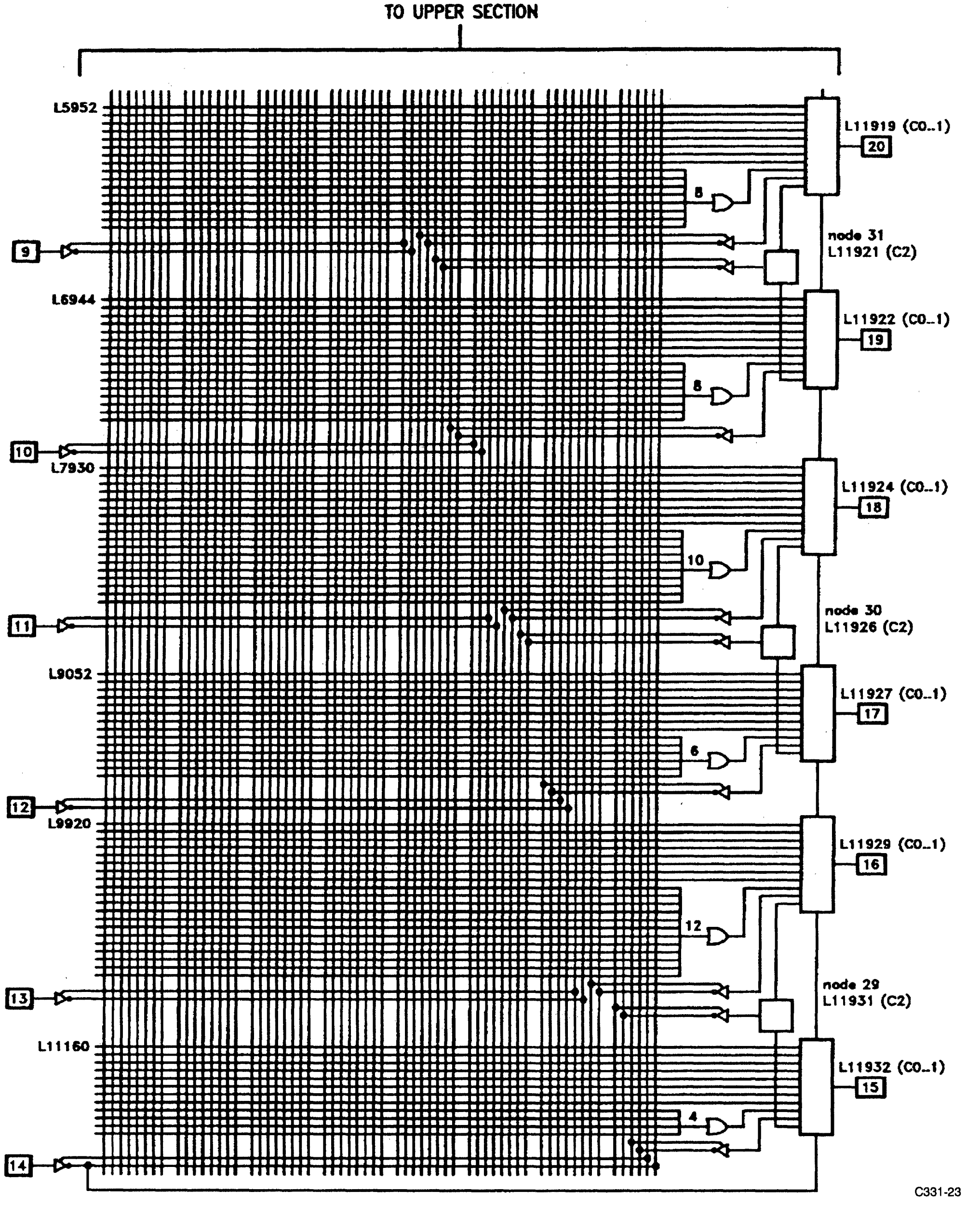
**Figure 3. Timing Configurations** (continued)

## CY7C331 Logic Diagram (Upper Half)



C331-22

## CY7C331 Logic Diagram (Lower Half)



C331-23

## Ordering Information

| $I_{CC1}$ (mA) | $t_{PD}$ (ns) | $t_S$ (ns) | $t_{CO}$ (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|---|---|
| 130 | 20 | 12 | 20 | CY7C331–20HC | H64 | 28-Pin Windowed Leaded Chip Carrier | Commercial |
| | | | | CY7C331–20JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | | | | CY7C331–20PC | P21 | 28-Lead (300-Mil) Molded DIP | |
| | | | | CY7C331–20WC | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| 160 | 25 | 15 | 25 | CY7C331–25DMB | D22 | 28-Lead (300-Mil) CerDIP | Military |
| | | | | CY7C331–25HMB | H64 | 28-Pin Windowed Leaded Chip Carrier | |
| | | | | CY7C331–25LMB | L64 | 28-Square Leadless Chip Carrier | |
| | | | | CY7C331–25QMB | Q64 | 28-Pin Windowed Leadless Chip Carrier | |
| | | | | CY7C331–25TMB | T74 | 28-Lead Windowed Cerpack | |
| | | | | CY7C331–25WMB | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| 120 | 25 | 12 | 25 | CY7C331–25HC | H64 | 28-Pin Windowed Leaded Chip Carrier | Commercial |
| | | | | CY7C331–25JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | | | | CY7C331–25PC | P21 | 28-Lead (300-Mil) Molded DIP | |
| | | | | CY7C331–25WC | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| 150 | 30 | 15 | 30 | CY7C331–30DMB | D22 | 28-Lead (300-Mil) CerDIP | Military |
| | | | | CY7C331–30HMB | H64 | 28-Pin Windowed Leaded Chip Carrier | |
| | | | | CY7C331–30LMB | L64 | 28-Square Leadless Chip Carrier | |
| | | | | CY7C331–30QMB | Q64 | 28-Pin Windowed Leadless Chip Carrier | |
| | | | | CY7C331–30TMB | T74 | 28-Lead Windowed Cerpack | |
| | | | | CY7C331–30WMB | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| 150 | 40 | 20 | 40 | CY7C331–40DMB | D22 | 28-Lead (300-Mil) CerDIP | Military |
| | | | | CY7C331–40HMB | H64 | 28-Pin Windowed Leaded Chip Carrier | |
| | | | | CY7C331–40LMB | L64 | 28-Square Leadless Chip Carrier | |
| | | | | CY7C331–40QMB | Q64 | 28-Pin Windowed Leadless Chip Carrier | |
| | | | | CY7C331–40TMB | T74 | 28-Lead Windowed Cerpack | |
| | | | | CY7C331–40WMB | W22 | 28-Lead (300-Mil) Windowed CerDIP | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{IS}$ | 9, 10, 11 |
| $t_{IH}$ | 9, 10, 11 |
| $t_{WH}$ | 9, 10, 11 |
| $t_{WL}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{PD}$ | 9, 10, 11 |
| $t_{IAR}$ | 9, 10, 11 |
| $t_{IAS}$ | 9, 10, 11 |
| $t_{PXZ}$ | 9, 10, 11 |
| $t_{PZX}$ | 9, 10, 11 |
| $t_{ER}$ | 9, 10, 11 |
| $t_{EA}$ | 9, 10, 11 |
| $t_{S}$ | 9, 10, 11 |
| $t_{H}$ | 9, 10, 11 |

Document #: 38–00066–D

CYPRESS

# CY7C335

# Universal Synchronous EPLD

## Features

- 100-MHz output registered operation
- Twelve I/O macrocells, each having:
  - —Registered, three-state I/O pins
  - —Input and output register clock select multiplexer
  - —Feed back multiplexer
  - —Output enable (OE) multiplexer
- Bypass on input and output registers
- All twelve macrocell state registers can be hidden
- User configurable I/O macrocells to implement JK or RS flip-flops and T or D registers
- Input multiplexer per pair of I/O macrocells allows I/O pin associated with a hidden macrocell state register to be saved for use as an input
- Four dedicated hidden registers
- Twelve dedicated registered inputs with individually programmable bypass option
- Three separate clocks—two input clocks, two output clocks
- Common (pin 14-controlled) or product term-controlled output enable for each I/O pin
- 256 product terms—32 per pair of macrocells, variable distribution
- Global, synchronous, product term-controlled, state register set and reset—inputs to product term are clocked by input clock
  - —2-ns input set-up and 9-ns output register clock to output
  - —10-ns input register clock to state register clock
- 28-pin, 300-mil DIP, LCC, PLCC
- Erasable and reprogrammable
- Programmable security bit

## Functional Description

The CY7C335 is a high-performance, erasable, programmable logic device (EPLD) whose architecture has been optimized to enable the user to easily and efficiently construct very high performance state machines.

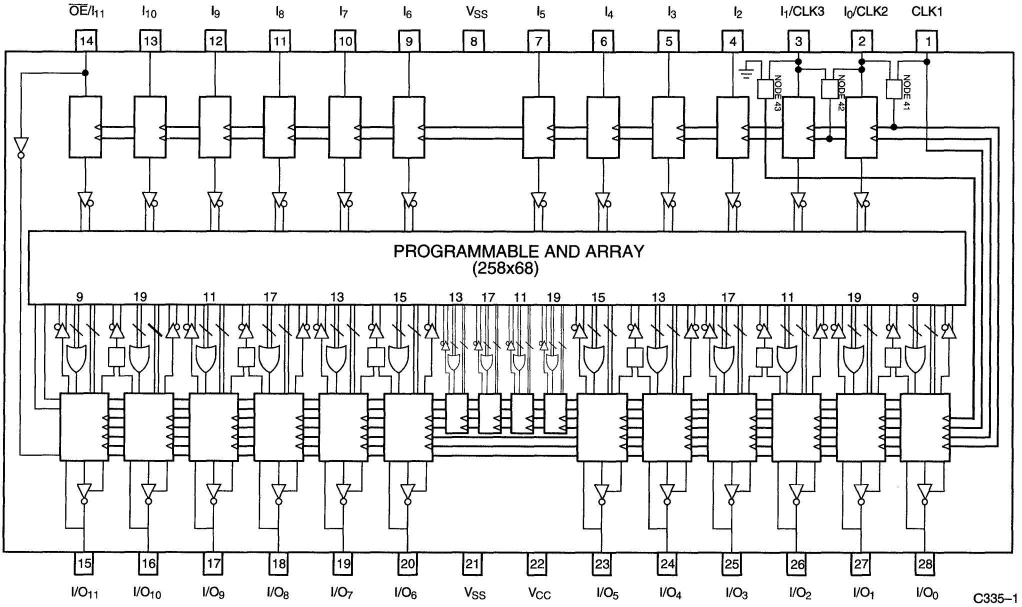
The architecture of the CY7C335, consisting of the user-configurable output macrocell, bidirectional I/O capability, input registers, and three separate clocks, enables the user to design high-performance state machines that can communicate either with each other or with microprocessors over bidirectional parallel buses of user-definable widths.

The four clocks permit independent, synchronous state machines to be synchronized to each other.

The user-configurable macrocells enable the designer to designate JK-, RS-, T-, or D-type devices so that the number of product terms required to implement the logic is minimized.

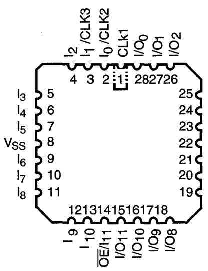
The CY7C335 is available in a wide variety of packages including 28-pin, 300-mil plastic and ceramic DIPs, PLCCs, and LCCs.

## Logic Block Diagram



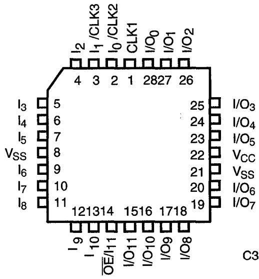
## Pin Configurations

**LCC**
**Top View**



C335–2

**PLCC**
**Top View**



C335–3

## Selection Guide

| | | CY7C335–100 | CY7C335–83 | CY7C335–66 | CY7C335–50 |
|---|---|---|---|---|---|
| Maximum Operating Frequency (MHz) | Commercial | 100 | 83.3 | 66.6 | 50 |
| | Military | | 83.3 | 66.6 | 50 |
| $I_{CC1}$ (mA) | Commercial | 140 | 140 | 140 | 140 |
| | Military | | 160 | 160 | 160 |

## Architecture Configuration Bits

The architecture configuration bits are used to program the multiplexers. The function of the architecture bits is outlined in *Table 1.*

**Table 1. Architecture Configuration Bits**

| Architecture Configuration Bit | | Number of Bits | Value | Function |
|---|---|---|---|---|
| C0 | Output Enable Select MUX | 12 Bits, 1 Per I/O Macrocell | 0—Virgin State | Output Enable Controlled by Product Term |
| | | | 1—Programmed | Output Enable Controlled by Pin 14 |
| C1 | State Register Feed Back MUX | 12 Bits, 1 Per I/O Macrocell | 0—Virgin State | State Register Output is Fed Back to Input Array |
| | | | 1—Programmed | I/O Macrocell is Configured as an Input and Output of Input Path is Fed to Array |
| C2 | I/O Macrocell Input Register Clock Select MUX | 12 Bits, 1 Per I/O Macrocell | 0—Virgin State | ICLK1 Controls the Input Register I/O Macrocell Input Register Clock Input |
| | | | 1—Programmed | ICLK2 Controls the Input Register I/O Macrocell Input Register Clock Input |
| C3 | Input Register Bypass MUX—I/O Macrocell | 12 Bits, 1 Per I/O Macrocell | 0—Virgin State | Selects Input to Feedback MUX from Input Register |
| | | | 1—Programmed | Selects Input to Feedback MUX from I/O pin |
| C4 | Output Register Bypass MUX | 12 Bits, 1 Per I/O Macrocell | 0—Virgin State | Selects Output from the State Register |
| | | | 1—Programmed | Selects Output from the Array, Bypassing the State Register |
| C5 | State Clock MUX | 16 Bits, 1 Per I/O Macrocell and 1 Per Hidden Macrocell | 0—Virgin State | State Clock 1 Controls the State Register |
| | | | 1—Programmed | State Clock 2 Controls the State Register |

**Table 1. Architecture Configuration Bits** (continued)

| Architecture Configuration Bit | | Number of Bits | Value | Function |
|---|---|---|---|---|
| C6 | Dedicated Input Register Clock Select MUX | 12 Bits, 1 Per Dedicated Input Cell | 0—Virgin State | ICLK1 Controls the Input Register I/O Macrocell Dedicated Input Register Clock Input |
| | | | 1—Programmed | ICLK2 Controls the Input Register I/O Macrocell Dedicated Input Register Clock Input |
| C7 | Input Register Bypass MUX—Input Cell | 12 Bits, 1 Per Dedicated Input Cell | 0—Virgin State | Selects Input to Array from Input Register |
| | | | 1—Programmed | Selects Input to Array from Input Pin |
| C8 | ICLK2 Select MUX | 1 Bit | 0—Virgin State | Input Clock 2 Controlled by Pin 2 |
| | | | 1—Programmed | Input Clock 2 Controlled by Pin 3 |
| C9 | ICLK1 Select MUX | 1 Bit | 0—Virgin State | Input Clock 1 Controlled by Pin 2 |
| | | | 1—Programmed | Input Clock 1 Controlled by Pin 1 |
| C10 | SCLK2 Select MUX | 1 Bit | 0—Virgin State | State Clock 2 Grounded |
| | | | 1—Programmed | State Clock 2 Controlled by Pin 3 |
| CX (11–16) | I/O Macrocell Pair Input Select MUX | 6 Bits, 1 Per I/O Macrocell Pair | 0—Virgin State | Selects Data from I/O Macrocell Input Path of Macrocell A of Macrocell Pair |
| | | | 1—Programmed | Selects Data from I/O Macrocell Input Path of Macrocell B of Macrocell Pair |

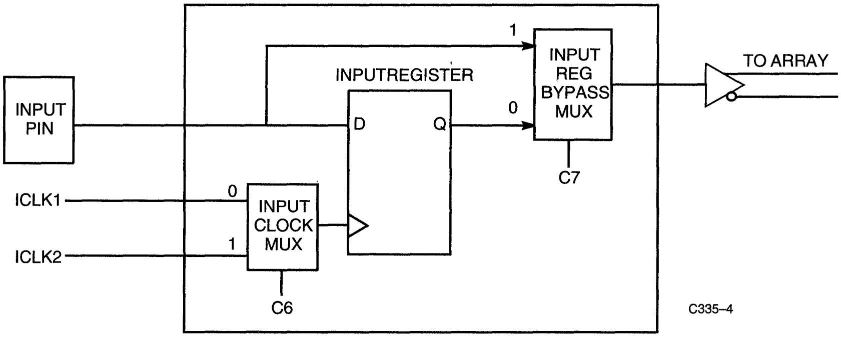


**Figure 1. CY7C335 Input Macrocell**

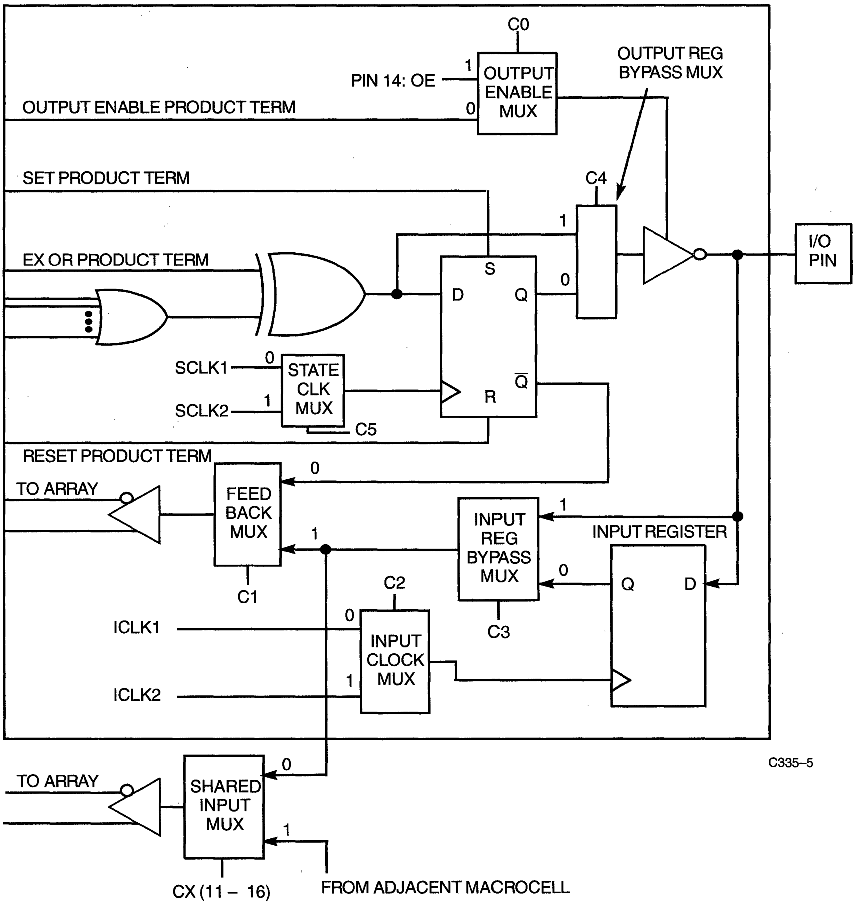
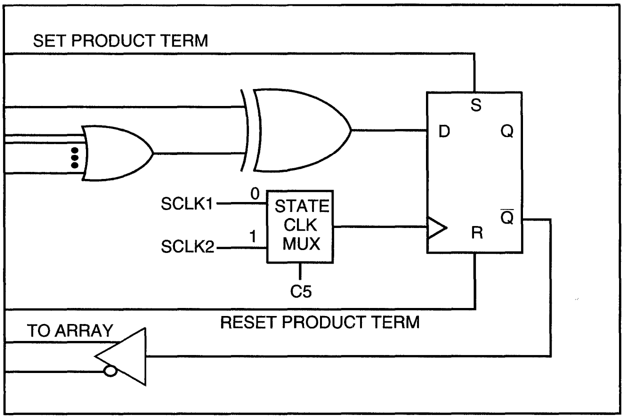
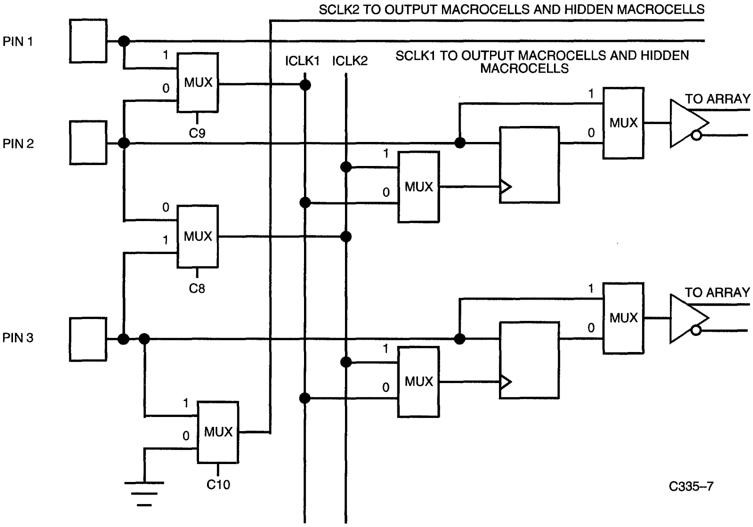


**Figure 2. CY7C335 Input/Output Macrocell**



C335–6

**Figure 3. CY7C335 Hidden Macrocell**


SCLK2 TO OUTPUT MACROCELLS AND HIDDEN MACROCELLS
PIN 1
1
0
MUX
C9
ICLK1
ICLK2
SCLK1 TO OUTPUT MACROCELLS AND HIDDEN MACROCELLS
1
MUX
0
TO ARRAY
PIN 2
1
0
MUX
0
1
MUX
C8
1
MUX
0
TO ARRAY
PIN 3
1
0
MUX
1
0
MUX
C10

C335–7

**Figure 4. CY7C335 Input Clocking Scheme**

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................ −65°C to +150°C

Ambient Temperature with
Power Applied ............................................ −55°C to +125°C

Supply Voltage to Ground Potential
(Pin 22 to Pins 8 and 21) .............................. −0.5V to +7.0V

DC Voltage Applied to Outputs
in High Z State ............................................ −0.5V to +7.0V

DC Input Voltage .......................................... −3.0V to +7.0V

Output Current into Outputs (Low) .............................. 12 mA

Static Discharge Voltage ........................................... >2001V
(per MIL-STD-883, Method 3015)

Latch-Up Current ..................................................... >200 mA

DC Programming Voltage ............................................ 13.0V

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +75°C | 5V ± 10% |
| Industrial | −40°C to +85°C | 5V ± 10% |
| Military[1] | −55°C to +125°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.,<br>$V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | Com'l | 2.4 | | V |
| | | | | Mil/Ind | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.,<br>$V_{IN} = V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 12 mA | Com'l | | 0.5 | V |
| | | | | Mil/Ind | | | |
| $V_{IH}$ | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs[3] | | | 2.2 | | V |
| $V_{IL}$ | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs[3] | | | | 0.8 | V |
| $I_{IX}$ | Input Leakage Current | $V_{SS} \leq V_{IN} \leq V_{CC}$, $V_{CC}$ = Max. | | | −10 | 10 | μA |
| $I_{OZ}$ | Output Leakage Current | $V_{CC}$ = Max., $V_{SS} \leq V_{OUT} \leq V_{CC}$ | | | −40 | 40 | μA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[4, 5] | | | −30 | −90 | mA |
| $I_{CC1}$ | Standby Power Supply Current | $V_{CC}$ = Max., $V_{IN}$ = GND<br>Outputs Open | | Com'l | | 140 | mA |
| | | | | Mil/Ind | | 160 | mA |
| $I_{CC2}$ | Power Supply Current at Frequency[5] | $V_{CC}$ = Max.,<br>Outputs Disabled (in High Z State),<br>Device Operating at $f_{MAX}$ External ($f_{MAX5}$) | | Com'l | | 180 | mA |
| | | | | Mil/Ind | | 200 | mA |

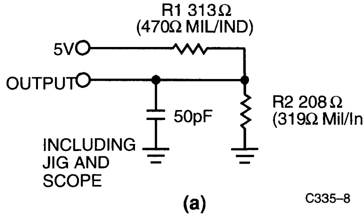
## Capacitance[5]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0V @ f = 1 MHz | | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V @ f = 1 MHz | | 10 | pF |

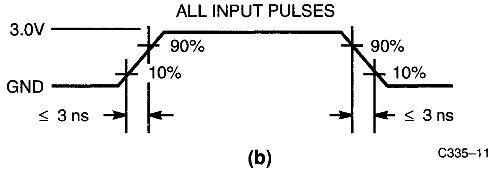
**Notes:**

1. $t_A$ is the "instant on" case temperature.
2. See the last page of this specification for Group A subgroup testing information.
3. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
4. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by ground degradation.
5. Tested initially and after any design or process changes that may affect these parameters.

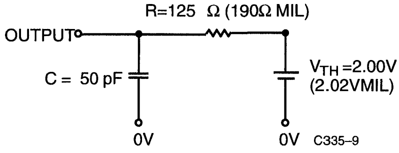
## AC Test Loads and Waveforms (Commercial)



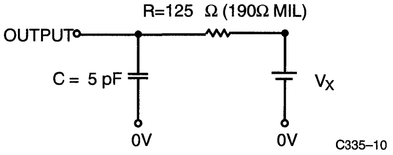
(a)



(b)



(c) Thévenin Equivalent (Load 1)



(d) Three-state Delay Load (Load2)

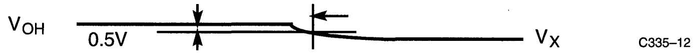
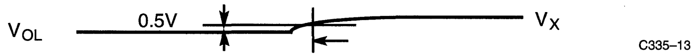
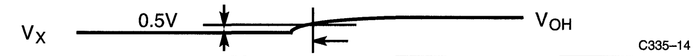
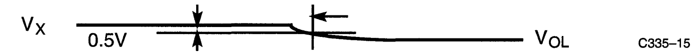
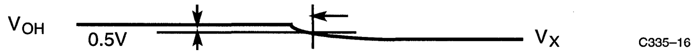
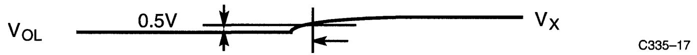
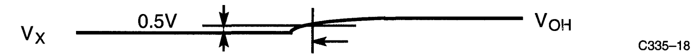
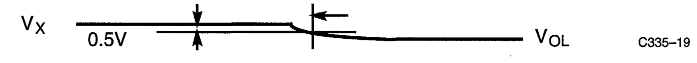
| Parameter | $V_X$ | Output Waveform Measurement Level |
|---|---|---|
| $t_{PXZ}$ (–) | 1.5V |  |
| $t_{PXZ}$ (+) | 2.6V |  |
| $t_{PZX}$ (+) | $V_{th}$ |  |
| $t_{PZX}$ (–) | $V_{th}$ |  |
| $t_{CER}$ (–) | 1.5V |  |
| $t_{CER}$ (+) | 2.6V |  |
| $t_{CEA}$ (+) | $V_{th}$ |  |
| $t_{CEA}$ (–) | $V_{th}$ |  |

**Figure 5. Test Waveforms**

## Commercial AC Characteristics

| | | 7C335–100 | | 7C335–83 | | 7C335–66 | | 7C335–50 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | Description | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| **Combinatorial Mode Parameters** | | | | | | | | | | |
| $t_{PD}$ | Input to Output Propagation Delay | | 15 | | 15 | | 20 | | 25 | ns |
| $t_{EA}$ | Input to Output Enable | | 15 | | 15 | | 20 | | 25 | ns |
| $t_{ER}$ | Input to Output Disable | | 15 | | 15 | | 20 | | 25 | ns |
| **Input Registered Mode Parameters** | | | | | | | | | | |
| $t_{WH}$ | Input and Output Clock Width HIGH[5] | 4 | | 5 | | 6 | | 8 | | ns |
| $t_{WL}$ | Input and Output Clock Width LOW[5] | 4 | | 5 | | 6 | | 8 | | ns |
| $t_{IS}$ | Input or Feedback Set-Up Time to Input Clock | 2 | | 2 | | 2 | | 3 | | ns |
| $t_{IH}$ | Input Register Hold Time from Input Clock | 2 | | 2 | | 2 | | 3 | | ns |
| $t_{ICO}$ | Input Register Clock to Output Delay | | 18 | | 18 | | 20 | | 25 | ns |
| $t_{IOH}$ | Output Data Stable Time from Input Clock | 3 | | 3 | | 3 | | 3 | | ns |
| $t_{IOH} - t_{IH}$ 33x | Output Data Stable from Input Clock Minus Input Register Hold Time for 7C335[6] | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{PZX}$ | Pin 14 Enable to Output Enabled | | 12 | | 12 | | 15 | | 20 | ns |
| $t_{PXZ}$ | Pin 14 Disable to Output Disabled | | 12 | | 12 | | 15 | | 20 | ns |
| $f_{MAX1}$ | Maximum Frequency of (2) CY7C335s in Input Registered Mode (Lowest of $1/(t_{ICO}+t_{IS})$ & $1/(t_{WL}+t_{WH})$)[5] | 50 | | 50 | | 45.4 | | 35.7 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Input Registered Mode (Lowest of ($1/(t_{ICO})$, $1/(t_{WH}+t_{WL})$, $1/(t_{IS}+t_{IH})$)[5] | 55.5 | | 55.5 | | 50 | | 40 | | MHz |
| $t_{ICEA}$ | Input Clock to Output Enabled | | 17 | | 17 | | 20 | | 25 | ns |
| $t_{ICER}$ | Input Clock to Output Disabled | | 15 | | 15 | | 20 | | 25 | ns |
| **Output Registered Mode Parameters** | | | | | | | | | | |
| $t_{CEA}$ | Output Clock to Output Enabled[5] | | 17 | | 17 | | 20 | | 25 | ns |
| $t_{CER}$ | Output Clock to Output Disabled[5] | | 15 | | 15 | | 20 | | 25 | ns |
| $t_S$ | Output Register Input Set-Up Time from Output Clock | 8 | | 9 | | 12 | | 15 | | ns |
| $t_H$ | Output Register Input Hold Time from Output Clock | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{CO}$ | Output Register Clock to Output Delay | | 9 | | 10 | | 12 | | 15 | ns |
| $t_{CO2}$ | Input Output Register Clock or Latch Enable to Combinatorial Output Delay (Through Logic Array)[5] | | 17 | | 18 | | 23 | | 30 | ns |
| $t_{OH}$ | Output Data Stable Time from Output Clock | 2 | | 2 | | 2 | | 2 | | ns |
| $t_{OH2}$ | Output Data Stable Time From Output Clock (Through Memory Array)[5] | 3 | | 3 | | 3 | | 3 | | ns |
| $t_{OH2}–t_{IH}$ | Output Data Clock Stable Time From Output Clock Minus Input Register Hold Time[5] | 0 | | 0 | | 0 | | 0 | | ns |
| $f_{MAX3}$ | Maximum Frequency with Internal Feedback in Output Registered Mode[5] | 100 | | 83.3 | | 66.6 | | 50 | | MHz |
| $f_{MAX4}$ | Maximum Frequency of (2) CY7C335s in Output Registered Mode (Lowest of $1/(t_{CO} + t_S)$ & $1/(t_{WL} + t_{WH})$)[5] | 58.8 | | 50 | | 41.6 | | 33.3 | | MHz |
| $f_{MAX5}$ | Maximum Frequency Data Path in Output Registered Mode (Lowest of $1/(t_{CO})$, $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$)[5] | 111 | | 100 | | 83.3 | | 62.5 | | MHz |
| $t_{OH} - t_{IH}$ 33x | Output Data Stable from Output Clock Minus Input Register Hold Time for 7C335[6] | 0 | | 0 | | 0 | | 0 | | ns |

## Commercial AC Characteristics (continued)

| Parameter | Description | 7C335–100 | | 7C335–83 | | 7C335–66 | | 7C335–50 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Pipelined Mode Parameters** | | | | | | | | | | |
| $t_{COS}$ | Input Clock to Output Clock | 10 | | 12 | | 15 | | 20 | | ns |
| $f_{MAX6}$ | Maximum Frequency Pipelined Mode (Lowest of $1/(t_{COS})$, $1/(t_{CO})$, $1/(t_{WL}+t_{WH})$), $1/(t_{IS}+t_{IH})$[5] | 100 | | 83.3 | | 66.6 | | 50 | | MHz |
| $f_{MAX7}$ | Maximum Frequency of (2) CY7C335s in Pipelined Mode (Lowest of $1/(t_{CO}+t_{IS})$ or $1/t_{COS}$) | 90.9 | | 83.3 | | 66.6 | | 50 | | MHz |
| **Power-Up Reset Parameters** | | | | | | | | | | |
| $t_{POR}$ | Power-Up Reset Time[5, 7] | | 1 | | 1 | | 1 | | 1 | µs |

## Military/Industrial AC Characteristics

| Parameter | Description | 7C335–83 | | 7C335–66 | | Unit |
|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | |
| $t_{PD}$ | Input to Output Propagation Delay | | 20 | | 20 | ns |
| $t_{EA}$ | Input to Output Enable | | 20 | | 20 | ns |
| $t_{ER}$ | Input to Output Disable | | 20 | | 20 | ns |
| **Input Registered Mode Parameters** | | | | | | |
| $t_{WH}$ | Input and Output Clock Width HIGH[5] | 5 | | 6 | | ns |
| $t_{WL}$ | Input and Output Clock Width LOW[5] | 5 | | 6 | | ns |
| $t_{IS}$ | Input or Feedback Set-Up Time to Input Clock | 3 | | 3 | | ns |
| $t_{IH}$ | Input Register Hold Time from Input Clock | 3 | | 3 | | ns |
| $t_{ICO}$ | Input Register Clock to Output Delay | | 23 | | 23 | ns |
| $t_{IOH}$ | Output Data Stable Time from Input Clock | 3 | | 3 | | ns |
| $t_{IOH}-t_{IH}$ 33x | Output Data Stable from Input Clock Minus Input Register Hold Time for 7C335[6] | 0 | | 0 | | ns |
| $t_{PZX}$ | Pin 14 Enable to Output Enabled | | 15 | | 15 | ns |
| $t_{PXZ}$ | Pin 14 Disable to Output Disabled | | 15 | | 15 | ns |
| $f_{MAX1}$ | Maximum Frequency of (2) CY7C335s in Input Registered Mode (Lowest of $1/(t_{ICO}+t_{IS})$ & $1/(t_{WL}+t_{WH})$)[5] | 38.4 | | 38.4 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Input Registered Mode (Lowest of ($1/(t_{ICO})$, $1/(t_{WH}+t_{WL})$, $1/(t_{IS}+t_{IH})$)[5] | 43.4 | | 43.4 | | MHz |
| $t_{ICEA}$ | Input Clock to Output Enabled | | 20 | | 20 | ns |
| $t_{ICER}$ | Input Clock to Output Disabled | | 20 | | 20 | ns |
| **Output Registered Mode Parameters** | | | | | | |
| $t_{CEA}$ | Output Clock to Output Enabled [5] | | 20 | | 20 | ns |
| $t_{CER}$ | Output Clock to Output Disabled [5] | | 20 | | 20 | ns |
| $t_S$ | Output Register Input Set-Up Time to Output Clock | 10 | | 12 | | ns |
| $t_H$ | Output Register Input Hold Time from Output Clock | 0 | | 0 | | ns |
| $t_{CO}$ | Output Register Clock to Output Delay | | 11 | | 12 | ns |
| $t_{CO2}$ | Output Register Clock or Latch Enable to Combinatorial Output Delay (Through Logic Array)[5] | | 22 | | 23 | ns |

**Notes:**

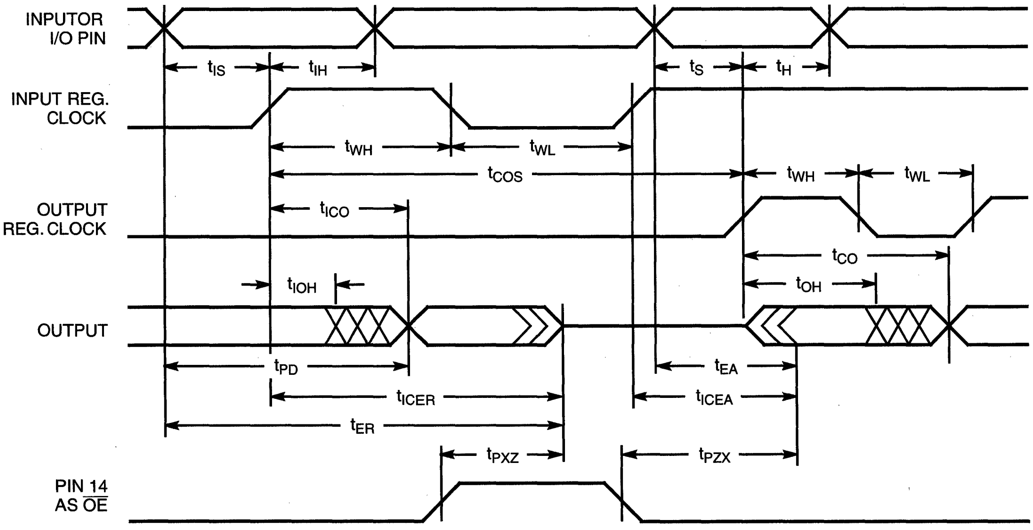
[6] This specification is intended to guarantee interface compatibility of the other members of the CY7C330 family with the CY7C335. This specification is met for the devices operating at the same ambient temperature and at the same power supply voltage.

[7] This part has been designed with the capability to reset during system power-up. Following power-up, the input and output registers will be reset to a logic LOW state. The output state will depend on how the array is programmed.

## Military/Industrial AC Characteristics (continued)

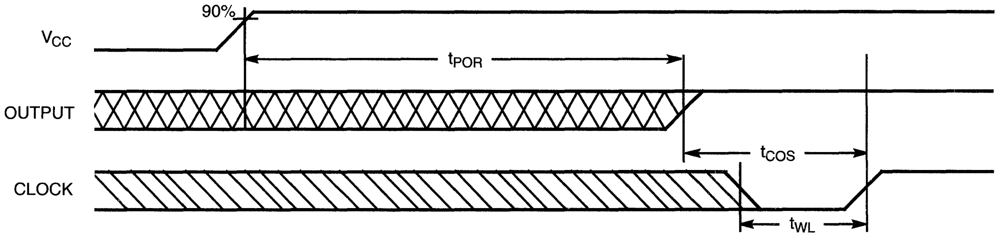
| Parameter | Description | 7C335–83 | | 7C335–66 | | Unit |
|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | |
| $t_{OH}$ | Output Data Stable Time from Output Clock | 2 | | 2 | | ns |
| $t_{OH2}$ | Output Data Stable Time From Output Clock (Through Memory Array)[5] | 3 | | 3 | | ns |
| $t_{OH2}-t_{IH}$ | Output Data Clock Stable Time From Output Clock Minus Input Register Hold Time[5] | 0 | | 0 | | ns |
| $f_{MAX3}$ | Maximum Frequency with Internal Feedback in Output Registered Mode[5] | 83.3 | | 66.6 | | MHz |
| $f_{MAX4}$ | Maximum Frequency of (2) CY7C335s in Output Registered Mode (Lower of $1/(t_{CO} + t_S)$ & $1/(t_{WL} + t_{WH})$)[5] | 47.6 | | 41.6 | | MHz |
| $f_{MAX5}$ | Maximum Frequency Data Path in Output Registered Mode (Lowest of $1/(t_{CO})$, $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$)[5] | 90.9 | | 83.3 | | MHz |
| $t_{OH} - t_{IH}$ 33x | Output Data Stable from Output Clock Minus Input Register Hold Time for 7C335[6] | 0 | | 0 | | ns |
| **Pipelined Mode Parameters** | | | | | | |
| $t_{COS}$ | Input Clock to Output Clock | 12 | | 15 | | ns |
| $f_{MAX6}$ | Maximum Frequency Pipelined Mode (Lowest of $1/(t_{COS})$, $1/(t_{IS})$, or $1/(t_{CO})$), $1/(t_{IS} + t_{IH})$[5] | 83.3 | | 66.6 | | MHz |
| $f_{MAX7}$ | Maximum Frequency of (2) CY7C335s in Pipelined Mode (Lowest of $1/(t_{CO} + t_{IS})$ or $1/t_{COS}$) | 71.4 | | 66.6 | | MHz |
| **Power-Up Reset Parameters** | | | | | | |
| $t_{POR}$ | Power-Up Reset Time[5, 7] | | 1 | | 1 | µs |

## Switching Waveform



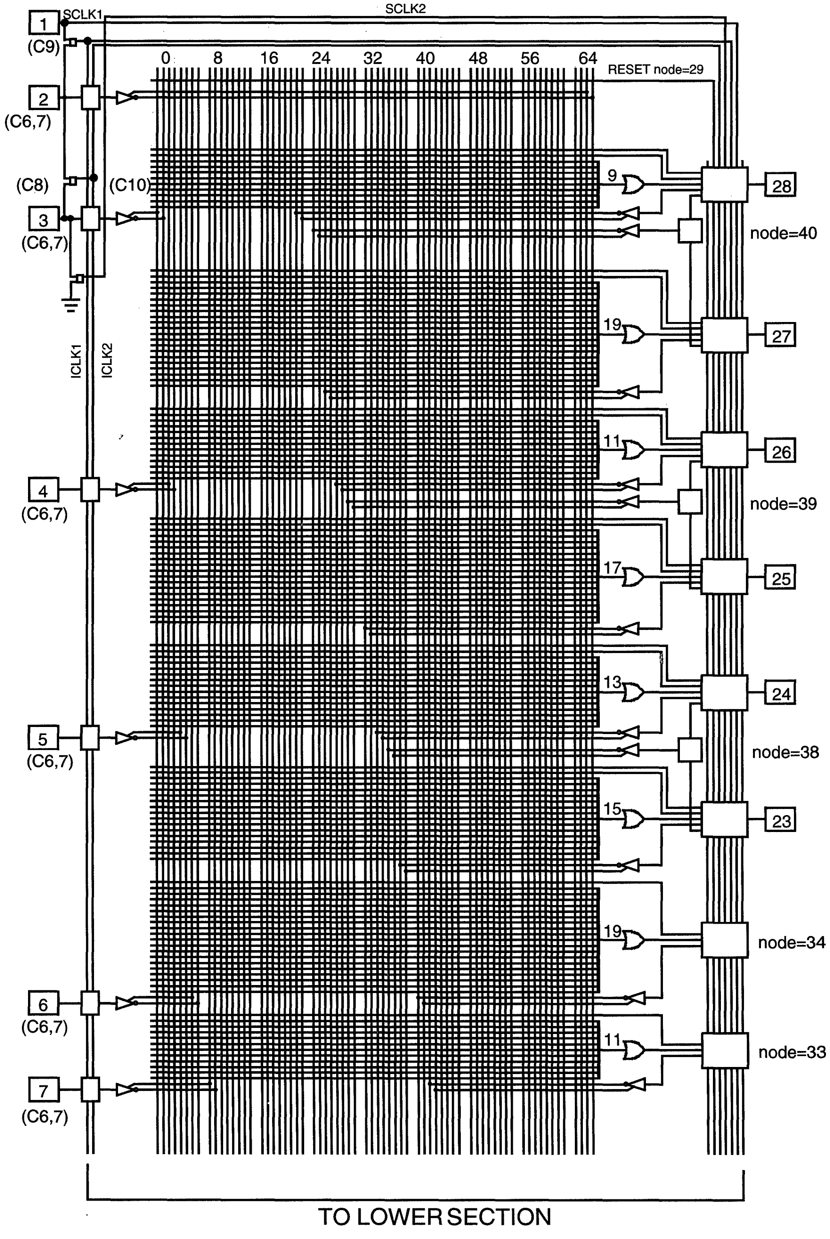
C335–20

### Power-Up Reset Waveform[7]

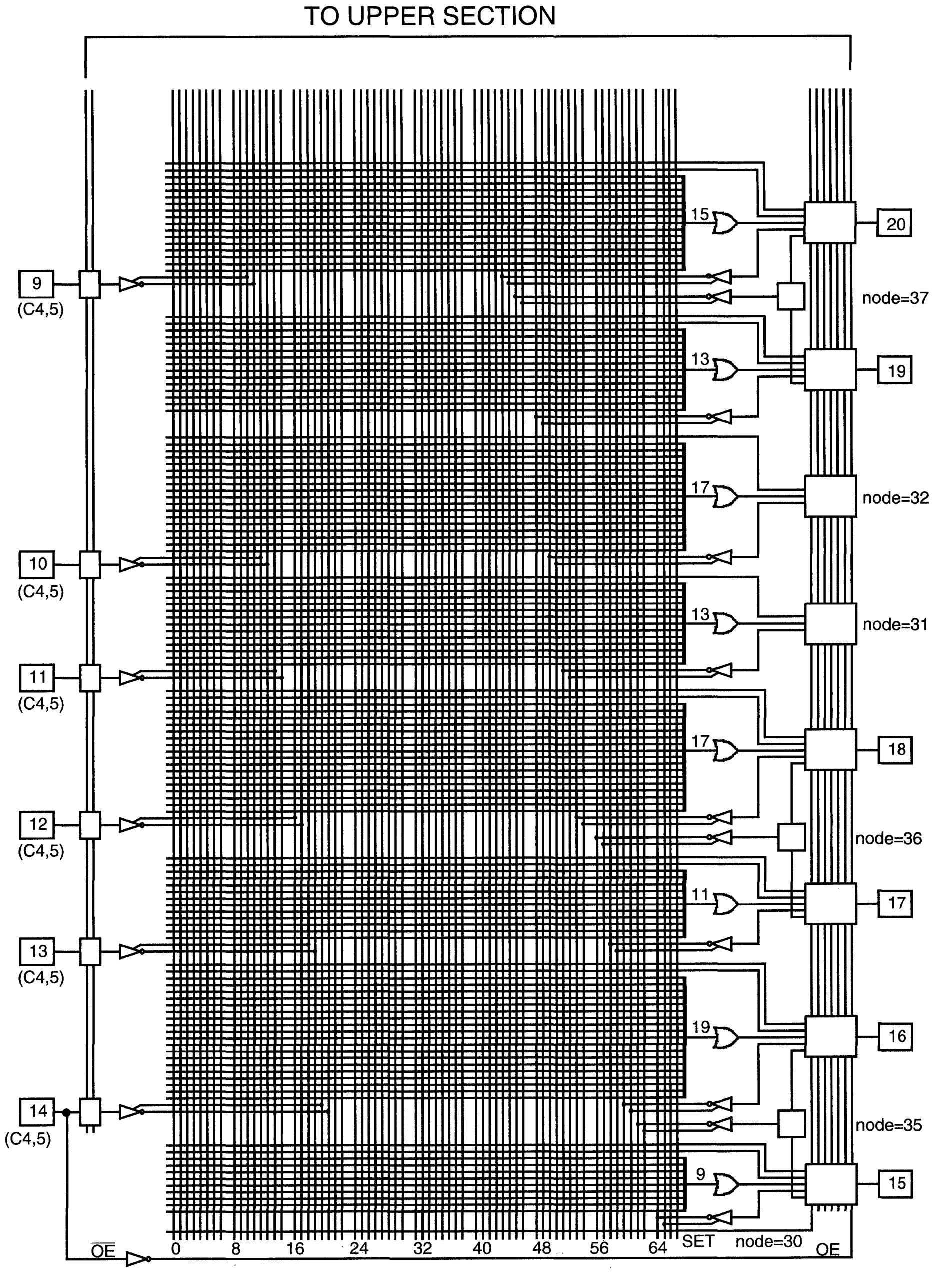


C335–21

## Block Diagram (Page 1 of 2)



**Block Diagram** (Page 2 of 2)



C335–23

## Ordering Information

| $f_{MAX}$ (MHz) | $I_{CC1}$ (mA) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|---|
| 100 | 140 | CY7C335–100WC | W22 | 28-Lead (300-Mil) Windowed CerDIP | Commercial |
| 83.3 | 160 | CY7C335–83LMB | L64 | 28-Square Leadless Chip Carrier | Military |
| | | CY7C335–83QMB | Q64 | 28-Pin Windowed Leadless Chip Carrier | |
| | | CY7C335–83WMB | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| | 140 | CY7C335–83HC | H64 | 28-Pin Windowed Leaded Chip Carrier | Commercial |
| | | CY7C335–83JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | | CY7C335–83WC | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| 66.6 | 160 | CY7C335–66QMB | Q64 | 28-Pin Windowed Leadless Chip Carrier | Military |
| | 140 | CY7C335–66HC | H64 | 28-Pin Windowed Leaded Chip Carrier | Commercial |
| | | CY7C335–66JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | | CY7C335–66PC | P21 | 28-Lead (300-Mil) Molded DIP | |
| | | CY7C335–66WC | W22 | 28-Lead (300-Mil) Windowed CerDIP | |
| 50 | 140 | CY7C335–50JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | | CY7C335–50PC | P21 | 28-Lead (300-Mil) Molded DIP | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{ICO}$ | 9, 10, 11 |
| $t_{IS}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{S}$ | 9, 10, 11 |
| $t_{H}$ | 9, 10, 11 |
| $t_{COS}$ | 9, 10, 11 |

Document #: 38–00186–D

CYPRESS

GENERAL INFORMATION 1

SMALL PLDS 2

***CPLDs*** 3

DEVELOPMENT TOOLS 4

APPLICATION NOTES 5

QUALITY 6

PACKAGES 7

## Table of Contents

## CPLDs (Complex PLDs)

| Device | Description | Page Number |
|---|---|---|
| Ultra37000™ ISR™ CPLD Family | UltraLogic™ High-Performance CPLDs | 3-1 |
| Ultra37128 | UltraLogic™ 128-Macrocell ISR™ CPLD | 3-8 |
| Ultra37128V | UltraLogic™ 3.3V 128-Macrocell ISR™ CPLD | 3-21 |
| Ultra37256 | UltraLogic™ 256-Macrocell ISR™ CPLD | 3-34 |
| Ultra37256V | UltraLogic™ 256-Macrocell 3.3V ISR™ CPLD | 3-47 |
| FLASH370i™ ISR™CPLD Family | UltraLogic™ High-Density Flash CPLDs | 3-60 |
| CY7C371i | UltraLogic™ 32-Macrocell Flash CPLD | 3-68 |
| CY7C372i | UltraLogic™ 64-Macrocell Flash CPLD | 3-77 |
| CY7C373i | UltraLogic™ 64-Macrocell Flash CPLD | 3-86 |
| CY7C374i | UltraLogic™ 128-Macrocell Flash CPLD | 3-96 |
| CY7C375i | UltraLogic™ 128-Macrocell Flash CPLD | 3-107 |
| CPLD Family FLASH370™ | UltraLogic™ High-Density FLASH CPLDs | 3-119 |
| CY7C371 | UltraLogic™ 32-Macrocell Flash CPLD | 3-126 |
| CY7C372 | UltraLogic™ 64-Macrocell Flash CPLD | 3-127 |
| CY7C373 | UltraLogic™ 64-Macrocell Flash CPLD | 3-128 |
| CY7C374 | UltraLogic™ 128-Macrocell Flash CPLD | 3-129 |
| CY7C375 | UltraLogic™ 128-Macrocell Flash CPLD | 3-130 |
| CY7C340 EPLD Family | Multiple Array Matrix High-Density EPLDs | 3-131 |
| CY7C341B | 192-Macrocell MAX® EPLD | 3-137 |
| CY7C341 | 192-Macrocell MAX® EPLD | 3-151 |
| CY7C342B | 128-Macrocell MAX® EPLDs | 3-153 |
| CY7C342 | 128-Macrocell MAX® EPLDs | 3-171 |
| CY7C343/CY7C343B | 64-Macrocell MAX® EPLD | 3-172 |
| CY7C344/CY7C344B | 32-Macrocell MAX® EPLD | 3-189 |
| CY7C346/CY7C346B | 128-Macrocell MAX® EPLDs | 3-203 |

CYPRESS

*PRELIMINARY*

# Ultra37000™ ISR™ CPLD Family

## UltraLogic™ High-Performance CPLDs

## Features

- In-System Reprogrammable (ISR™) CMOS CPLDs
  - JTAG interface for reconfigurability
  - Design changes don't cause pinout changes
  - Design changes don't cause timing changes
- High density
  - 32–512 macrocells
  - 32–256 I/O pins
  - 5–6 Dedicated Inputs including 4 clock pins
- High speed - 167 MHz in-system operation
  - $t_{PD}$ = 6.5 ns
  - $t_S$ = 3.5 ns
  - $t_{CO}$ = 4.5 ns
- Simple timing model
  - No fanout delays
  - No expander delays
  - No dedicated vs. I/O pin delays
  - No additional delay through PIM
  - No penalty for using full 16 product terms
  - No delay for steering or sharing product terms
- 3.3V and 5V versions
- Fully PCI compliant[1]
- Bus Hold capabilities on all I/Os
- Intelligent product term allocator provides:
  - 0–16 product terms to any macrocell
  - Product term steering on an individual basis
  - Product term sharing among local macrocells
- Flexible clocking
  - 4 synchronous clocks per device
  - Product Term clocking
  - Clock polarity control
- Consistent package/pinout offering across all densities
  - Same pinout for 3.3V and 5V devices
  - Simplifies design migration across density
- Security bit and user ID supported
- Packages
  - 44–352 pins in PLCC, TQFP, and BGA packages
- *Warp2*®
  - Low-cost IEEE 1076/1164-compliant VHDL system
  - Available on PC, Sun, and HP platforms for $99
  - Supports all Cypress Programmable Products
- *Warp2*Sim™ adds:
  - Capabilities of *Warp2* and Viewlogic's ViewSim
  - Dynamic timing solutions for all Cypress PLDs
- *Warp3*® CAE development system adds:
  - VHDL input
  - Viewlogic graphical user interface
  - Schematic capture (ViewDraw™)
  - VHDL simulation (SpeedWave™)

## General Description

The Ultra37000™ family of CMOS CPLDs provides a range of high-density programmable logic solutions with unparalleled system performance. The Ultra37000 family is designed to bring the flexibility, ease of use, and performance of the 22V10 to high-density CPLDs. The architecture is based on a number of logic blocks that are connected by a Programmable Interconnect Matrix (PIM). Each logic block features its own product term array, product term allocator, and 16 macrocells. The PIM distributes signals from the logic block outputs and all input pins to the logic block inputs. *Figure 1* shows a block diagram of the Ultra37128.

All of the Ultra37000 devices are electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows, thereby reducing costs. The ISR feature provides the ability to reconfigure the devices without having design changes cause pinout or timing changes. The Cypress ISR function is implemented through a JTAG-compliant serial interface. Data is shifted in and out through the TDI and TDO pins, respectively. Because of the superior routability and simple timing model of the Ultra37000 devices, ISR allows users to change existing logic designs while simultaneously fixing pinout assignments and maintaining system performance.

The entire family features JTAG for ISR and boundary scan, and is fully compliant with the PCI Local Bus specification, meeting all the electrical and timing requirements. The Ultra37000 family features bus-hold capabilities on all I/Os.

## Ultra37000 Selection Guide

| Device | Pins | Macrocells | Dedicated Inputs | I/O Pins | Speed ($t_{PD}$) | Speed ($f_{MAX}$) |
|---|---|---|---|---|---|---|
| Ultra37032/V | 44 | 32 | 6 | 32 | 6.5 | 167 |
| Ultra37064/V | 44/84/100 | 64 | 6 | 32/64 | 6.5 | 167 |
| Ultra37128/V | 84/100/160 | 128 | 5 | 64/128 | 6.5 | 167 |
| Ultra37192/V | 160/208 | 192 | 5 | 120/144 | 7.5 | 154 |
| Ultra37256/V | 208/256 | 256 | 5 | 128/160/192 | 7.5 | 154 |
| Ultra37384/V | 208/256 | 384 | 5 | 160/192 | 10 | 125 |
| Ultra37512/V | 208/352 | 512 | 5 | 160/256 | 10 | 125 |

Shaded area contains advance information.

**Note:**

[1] Due to the 5V tolerant nature of the I/Os, the I/Os are not clamped to $V_{CC}$.

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) consists of a completely global routing matrix for signals from I/O pins and feedbacks from the logic blocks. The PIM provides extremely robust interconnection to avoid fitting and density limitations.

The inputs to the PIM consist of all I/O and dedicated input pins and all macrocell feedbacks from within the logic blocks. The number of PIM inputs increases with pin count and the number of logic blocks. The outputs from the PIM are signals routed to the appropriate logic blocks. Each logic block receives 36 inputs from the PIM and their complements, allowing for 32-bit operations to be implemented in a single pass through the device. The wide number of inputs to the logic block also improves the routing capacity of the Ultra37000 family.

An important feature of the PIM is its simple timing. The propagation delay through the PIM is accounted for in the timing specifications for each device. There is no additional delay for traveling through the PIM. In fact, all inputs travel through the PIM. As a result, there are no route-dependent timing parameters on the Ultra37000 devices. The worst-case PIM delays are incorporated in all appropriate Ultra37000 specifications.

Routing signals through the PIM is completely invisible to the user. All routing is accomplished by software—no hand routing is necessary. *Warp* and third-party development packages automatically route designs for the Ultra37000 family in a matter of minutes. Finally, the rich routing resources of the Ultra37000 family accommodate last minute logic changes while maintaining fixed pin assignments.

### Logic Block

The logic block is the basic building block of the Ultra37000 architecture. It consists of a product term array, an intelligent product-term allocator, 16 macrocells, and a number of I/O cells. The number of I/O cells varies depending on the device used.

*Product Term Array*

Each logic block features a 72 x 87 programmable product term array. This array accepts 36 inputs from the PIM, which originate from macrocell feedbacks and device pins. Active LOW and active HIGH versions of each of these inputs are generated to create the full 72-input field. The 87 product terms in the array can be created from any of the 72 inputs.

Of the 87 product terms, 80 are for general-purpose use for the 16 macrocells in the logic block. Four of the remaining seven product terms in the logic block are output enable (OE) product terms. Each of the OE product terms controls up to eight of the 16 macrocells and is selectable on an individual macrocell basis. In other words, each I/O cell can select between one of two OE product terms to control the output buffer. The first two of these four OE product terms are available to the upper half of the I/O macrocells in a logic block. The other two OE product terms are available to the lower half of the I/O macrocells in a logic block.

## Logic Block Diagram

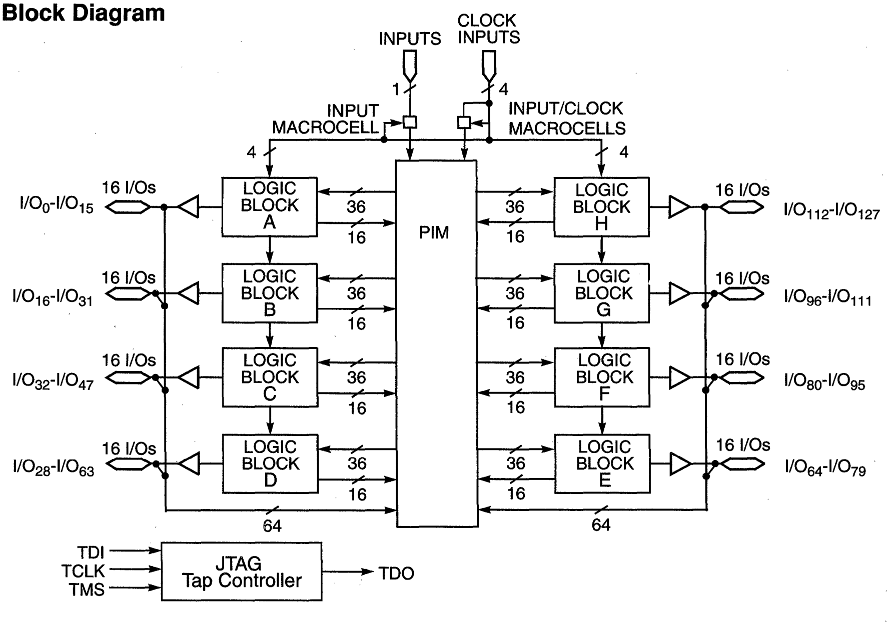
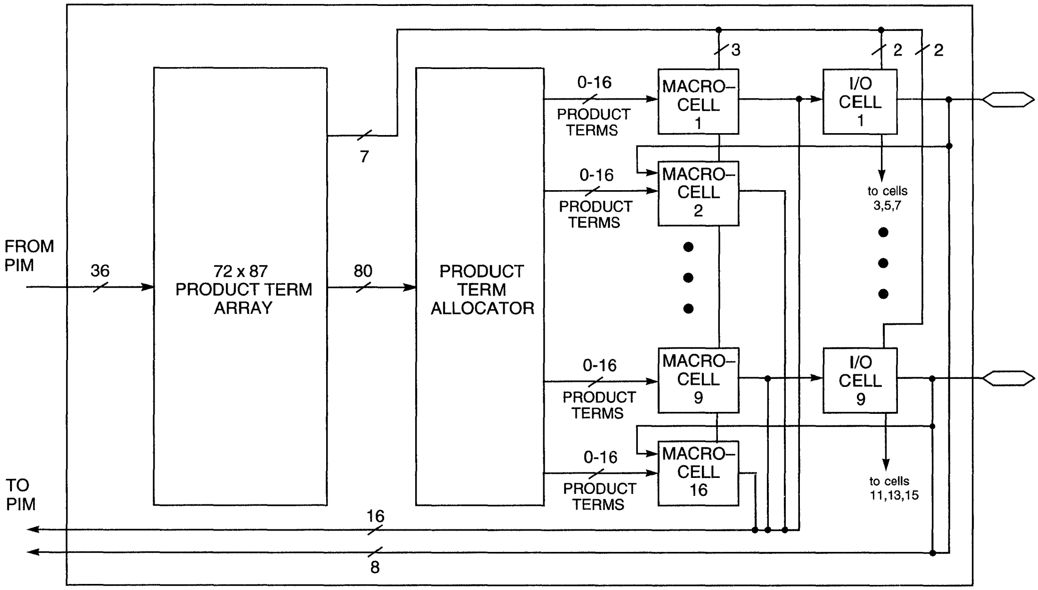


**Figure 1. Ultra37128 Block Diagram**



Ultra37000–3

**Figure 2. Logic Block with 50% Buried Macrocells**

The next two product terms in each logic block are dedicated asynchronous set and asynchronous reset product terms. The final product term is the product term clock. The set, reset, OE and product term clock have polarity control to realize OR functions in a single pass through the array.

### Product Term Allocator

Through the product term allocator, software automatically distributes product terms among the 16 macrocells in the logic block as needed. A total of 80 product terms are available from the local product term array. The product term allocator provides two important capabilities without affecting performance: product term steering and product term sharing.

*Product Term Steering*

Product term steering is the process of assigning product terms to macrocells as needed. For example, if one macrocell requires ten product terms while another needs just three, the product term allocator will "steer" ten product terms to one macrocell and three to the other. On Ultra37000 devices, product terms are steered on an individual basis. Any number between 0 and 16 product terms can be steered to any macrocell. Note that 0 product terms is useful in cases where a particular macrocell is unused or used as an input register.

*Product Term Sharing*

Product term sharing is the process of using the same product term among multiple macrocells. For example, if more than one output has one or more product terms in its equation that are common to other outputs, those product terms are only programmed once. The Ultra37000 product term allocator allows sharing across groups of four output macrocells in a variable fashion. The software automatically takes advantage of this capability—the user does not have to intervene.

Note that neither product term sharing nor product term steering have any effect on the speed of the product. All worst-case steering and sharing configurations have been incorporated in the timing specifications for the Ultra37000 devices.

*Low Power Option*

Each logic block can operate in high speed mode for critical path performance, or in low power mode for power conservation. The logic block mode is set by the user on a logic block by logic block basis.

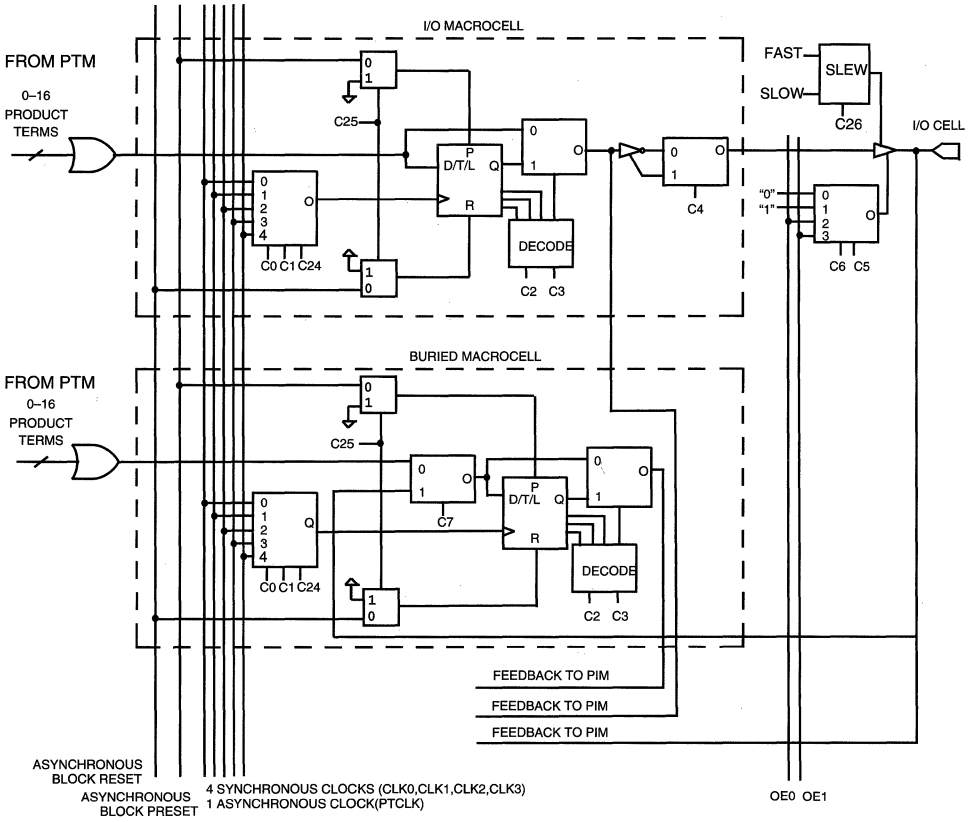


**Figure 3. I/O and Buried Macrocells**

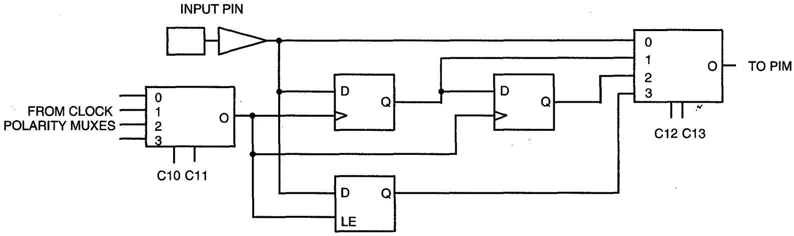


**Figure 4. Input Macrocell**

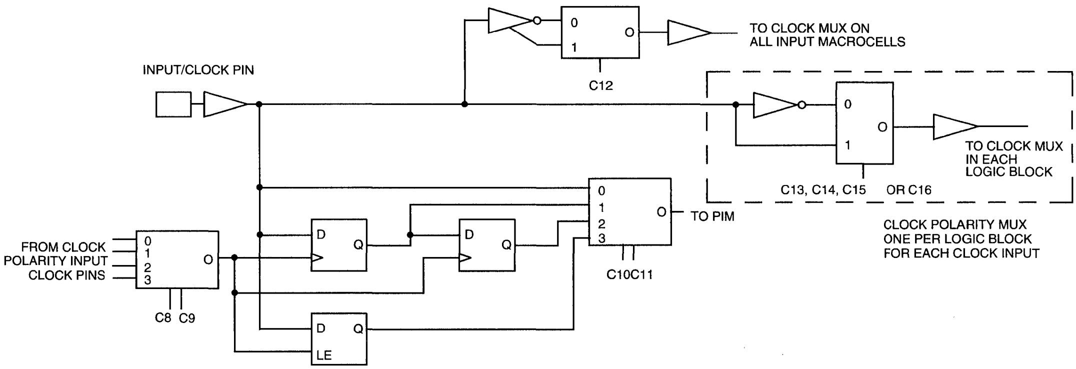


**Figure 5. Input/Clock Macrocell**

## Ultra37000 Macrocell

Within each logic block there are 16 macrocells. Macrocells can either be I/O Macrocells, which include an I/O Cell which is associated with an I/O pin, or buried Macrocells, which do not connect to an I/O. The combination of I/O Macrocells and buried Macrocells varies from device to device.

### *Buried Macrocell*

*Figure 3* displays the architecture of buried macrocells. The buried macrocell features a register that can be configured as combinatorial, a D flip-flop, a T flip-flop, or a level-triggered latch.

The register can be asynchronously set or asynchronously reset at the logic block level with the separate set and reset product terms. Each of these product terms features programmable polarity. This allows the registers to be set or reset based on an AND expression or an OR expression.

Clocking of the register is very flexible. four global synchronous clocks and a product term clock are available to clock the register. Furthermore, each clock features programmable polarity so that registers can be triggered on falling as well as rising edges (see the Clocking section). Clock polarity is chosen at the logic block level.

The buried macrocell also supports input register capability. The buried macrocell can be configured to act as an input register (D-type or latch) whose input comes from the I/O pin associated with the neighboring macrocell. The output of all buried macrocells is sent directly to the PIM regardless of its configuration.

### *I/O Macrocell*

*Figure 3* illustrates the architecture of the I/O macrocell. The I/O Macrocell supports the same functions as the Buried Macrocell with the addition of I/O capability. At the output of the macrocell, a polarity control mux is available to select active LOW or active HIGH signals. This has the added advantage of allowing significant logic reduction to occur in many applications.

The Ultra37000 macrocell features a feedback path to the PIM separate from the I/O pin input path. This means that if the macrocell is buried (fed back internally only), the associated I/O pin can still be used as an input.

### *Bus Hold Capabilities on all I/Os*

Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device's performance. As a latch, bus-hold maintains the last state of a pin when the pin is placed in a high-impedance state, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to VCC or GND.

### *Programmable Slew Rate Control*

Each output has a programmable configuration bit, which sets the output slew rate to fast or slow. For designs concerned with meeting FCC emissions standards the slow edge provides for lower system noise. For designs requiring very high performance the fast edge rate provides maximum system performance.

## Ultra37000 5V devices

The Ultra37000 devices operate with a 5V supply, can support 5V or 3.3V I/O levels. $V_{CCO}$ connections provide the capability of interfacing to either a 5V or 3.3V bus. By connecting the $V_{CCC}$ pins to 5V the user insures 5V TTL levels on the outputs. If $V_{CCC}$ is connected to 3.3V the output levels meet 3.3V JEDEC standard CMOS levels and are 5V tolerant. These devices require 5V ISR programming.

## Ultra37000V 3.3V devices

Devices operating with a 3.3V supply require 3.3V on all $V_{CCC}$ pins, reducing the device's power consumption. These devices support 3.3V JEDEC standard CMOS output levels, and are 5V tolerant. These devices allow 3.3V ISR programming.

## PCI Compliance

5V operation of the Ultra37000 is fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The 3.3V products meet all PCI requirements except for the output 3.3V clamp, which is in direct conflict with 5V tolerance. The Ultra37000 family's simple and predictable timing model ensures compliance with the PCI AC specifications independent of the design.

### IEEE 1149.1 Compliant JTAG

The Ultra37000 family has an IEEE 1149.1 JTAG interface for both Boundary Scan and ISR.

*Boundary Scan*

The Ultra37000 family supports Bypass, Sample/Preload, Extest, Idcode and Usercode boundary scan instructions. The JTAG interface is shown in *Figure 6.*

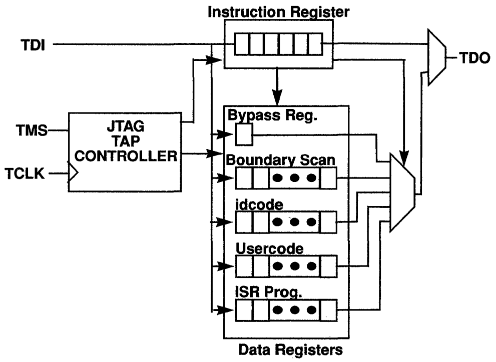


**Figure 6. JTAG Interface**

*In-System Reprogramming (ISR)*

In-System Reprogramming is the combination of the capability to program or reprogram a device on-board, and the ability to support design changes without changing the system timing or device pinout. This combination means design changes during debug or field upgrades do not cause board respins. The Ultra37000 family implements ISR by providing a JTAG compliant interface for on-board programming, robust routing resources for pinout flexability, and a simple timing model for consistent system performance.

### Clocking

Each I/O and buried macrocell has access to 4 synchronous clocks (CLK0, CLK1, CLK2 and CLK3) as well as an asynchronous product term clock PTCLK. Each input macrocell has access to all 4 synchronous clocks.

*Dedicated Inputs/Clocks*

Five pins on each member of the Ultra37000 family are designated as input-only. There are two types of dedicated inputs on Ultra37000 devices: input pins and input/clock pins. *Figure 4* illustrates the architecture for input pins. Four input options are available for the user: combinatorial, registered, double-registered, or latched. If a registered or latched option is selected, any one of the input clocks can be selected for control.

*Figure 5* illustrates the architecture for the input/clock pins. Like the input pins, input/clock pins can be combinatorial, registered, double registered, or latched. In addition, these pins feed the clocking structures throughout the device. The clock path at the input has user-configurable polarity.

*Product Term Clocking*

In addition to the 4 synchronous clocks, the Ultra37000 family also has a product term clock for asynchronous clocking. Each logic block has an independant product term clock which is available to all 16 macrocells. Each product term clock also supports user configurable polarity selection.

### Timing Model

One of the most important features of the Ultra37000 family is the simplicity of its timing. All delays are worst case and system performance is unaffected by the features used. *Figure 7* illustrates the true timing model for the 167 MHz devices in high speed mode. For combinatorial paths, any input to any output incurs an 6.5-ns worst-case delay regardless of the amount of logic used. For synchronous systems, the input set-up time to the output macrocells for any input is 3.5 ns and the clock to output time is also 4.0 ns. These measurements are for any output and synchronous clock, regardless of the logic used.

The Ultra37000 features:

- no fanout delays
- no expander delays
- no dedicated vs. I/O pin delays
- no additional delay through PIM
- no penalty for using 0-16 product terms
- no added delay for steering product terms
- no added delay for sharing product terms
- no routing delays
- no output bypass delays

The simple timing model of the Ultra37000 family eliminates unexpected performance penalties.

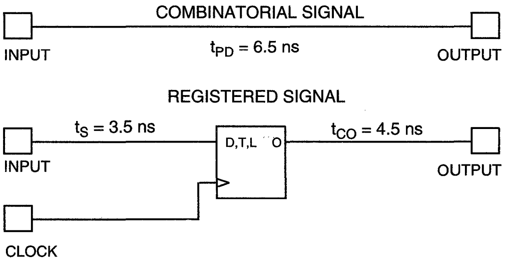


**Figure 7. Timing Model for Ultra37128**

## Development Software Support

### *Warp2*®

*Warp2* is a state-of-the-art VHDL compiler for designing with Cypress programmable logic. *Warp2* utilizes IEEE 1076/1164 VHDL as the Hardware Description Language (HDL) for design entry. VHDL provides a number of significant benefits for design entry. *Warp2* accepts VHDL input, synthesizes and optimizes the entered design, and outputs a JEDEC map for the desired Ultra37000 device. For simulation, *Warp2* provides a graphical waveform simulator as well as VHDL and Verilog Timing Models.

VHDL (VHSIC Hardware Description Language) is an open, powerful, non-proprietary language that is a standard for behavioral design entry and simulation. VHDL allows designers to learn a single language that is useful for all facets of the design process. See the *Warp2* data sheet for further information.

### *Warp3*®

*Warp3* is a sophisticated development system that is based on the latest version of Viewlogic's CAE design environment. *Warp3* features schematic capture (ViewDraw™), VHDL waveform simulation (SpeedWave™), a VHDL debugger, and VHDL synthesis, all integrated in a graphical design environment. *Warp3* is available on PCs using Windows 3.1, Windows95 or Windows NT and on Sun and Hewlett Packard workstations. See the *Warp3* data sheet for further information.

### *Warp2*Sim™

This development system combines the capabilities of *Warp2* and Viewlogic's ViewSim™ package which provides dynamic timing solutions for all Cypress programmable logic devices.

### Third-Party Software

Cypress products are supported in a number of third party design entry and simulation tools. All major third-party software vendors (including ABEL™, Synario, LOG/iC™, CUPL™, Mentor, etc.) provide support for the Ultra37000 family of devices. Refer to the third party software data sheet or contact your local sales office for a list of currently supported third party vendors.

## Programming

There are four programming options available for Ultra37000 devices. The first method is to use a PC with the InSRkit ISR programming cable and software. With this method, the ISR pins of the Ultra37000 devices are routed to a connector at the edge of the printed circuit board. The ISR programming cable is then connected between the parallel port of the PC and this connector. A simple configuration file instructs the ISR software of the programming operations to be performed on each of the Ultra37000 devices in the system. The ISR software then automatically completes all of the necessary data manipulations required to accomplish the programming, reading, verifying, and other ISR functions. For more information on Cypress ISR Interface, see the ISR Programming Kit.

The second method for programming Ultra37000 devices is on automatic test equipment (ATE.) Contact your local sales office for information on availability of this option.

The third programming option for Ultra37000 devices is to utilize the embedded controller or processor that already exists in the system. The Ultra37000 ISR software assists in this method by converting the device JEDEC maps into the ISR serial stream that contains the ISR instruction information and the addresses and data of locations to be programmed. The embedded controller then simply directs this ISR stream to the chain of Ultra37000 devices to complete the desired reconfiguring or diagnostic operations. Contact your local sales office for information on availability of this option.

The fourth method for programming Ultra37000 devices is tc use the same programmer that is currently being used to program FLASH370i devices.

For all pinout, electrical, and timing requirements, refer to device data sheets. For ISR cable and software specifications, refer to the InSRkit data sheets. For a detailed description of ISR capabilities, refer to the Cypress application note, "An Introduction to In System Reprogramming with Ultra37000."

The *Impulse3*™ device programmer from Cypress will program all Cypress programmable logic devices, as well as USB, PROM and EPROM products. This unit is a programmer that connects to any IBM-compatible PC via the printer port. For further information see the *Impulse3* data sheet.

### Third-Party Programmers

As with development software, Cypress support is available on a wide variety third-party programmers. All major third-party programmers (including Data I/O, Logical Devices, Minato, SMS, and Stag) support the Ultra37000 family.

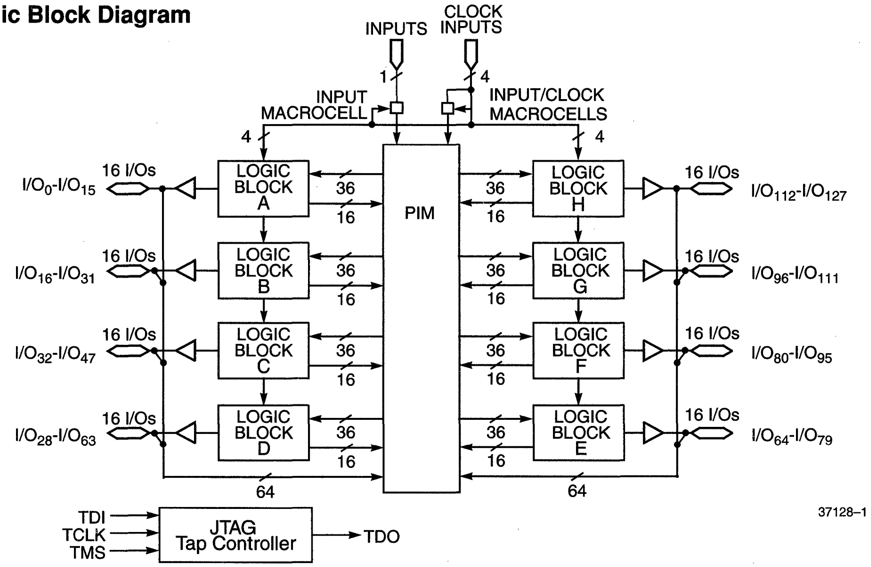
ISR, UltraLogic, Ultra37000, *Warp2*Sim, and *Impulse3* are trademarks of Cypress Semiconductor Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.
SpeedWave, ViewSim and ViewDraw are trademarks of ViewLogic.
ABEL is a trademark of Data I/O Corporation.

# UltraLogic™ 128-Macrocell ISR™ CPLD

## Features

- 128 macrocells in eight logic blocks
- In-System Reprogrammable (ISR™)
  - —JTAG compliant on board programming
  - —Design changes don't cause pinout changes
  - —Design changes don't cause timing changes
- I/O Intensive Version features an I/O for every Macocell
  - —128 I/O Macrocells
  - —128 I/O pins plus 5 dedicated inputs including 4 clock inputs
- Register Intensive Version
  - —64 buried Macrocells and 64 I/O Macrocells
  - —64 I/O pins and 5 or 6 dedicated inputs including 4 clock inputs
- IEEE 1149.1 JTAG boundary scan
- High speed
  - —$f_{MAX}$ = 167 MHz
  - —$t_{PD}$ = 6.5 ns
  - —$t_S$ = 3.5 ns
  - —$t_{CO}$ = 4.5 ns
- Product-term clocking
- Programmable slew rate control on individual I/Os
- Low power option on individual logic block basis
- 5V and 3.3V I/O capability
- Bus Hold capabilities on all I/Os
- Simple Timing Model
- Fully PCI compliant[1]
- Available in 84-pin PLCC, 100-pin TQFP and 160-pin PQFP packages
- Pinout compatible with and functional superset of FLASH374i/5i

## Logic Block Diagram



## Selection Guide

| | Ultra37128–167 | Ultra37128-154 | Ultra37128-125 | Ultra37128-83 |
|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | 6.5 | 7.5 | 10 | 15 |
| Minimum Set-Up, $t_S$ (ns) | 3.5 | 4.0 | 5.5 | 8 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 4.5 | 4.5 | 6.5 | 8 |
| Typical Supply Current, $I_{CC}$ (mA) in Low Power Mode | 75 | 75 | 75 | 75 |

**Note:**

[1] Due to the 5V tolerant nature of the I/Os, the I/Os are not clamped to $V_{CC}$.

## Functional Description

The Ultra37128 is an In-System Reprogrammable (ISR) Complex Programmable Logic Device (CPLD) and is part of the Ultra37000™ family of high-density, high-speed CPLDs. Like all members of the Ultra37000 family, the Ultra37128 is designed to bring the ease of use and high performance of the 22V10 to high-density PLDs.

The 128 macrocell Ultra37128 is available in register intensive and I/O intensive versions. The Ultra37128P84 and Ultra37128P100 feature 64 Buried Macrocells and 64 I/O Macrocells for register intensive designs which require small footprint devices. The Ultra37128P160 I/O intensive device has an I/O pin for each macrocell.

For a more detailed description of the architecture and features of the Ultra37128 see the Ultra37000 family data sheet.

### Fully Routable with 100% Logic Utilization

The Ultra37128 is designed with a robust routing architecture which allows utilization of the entire device, with a fixed pinout. This makes Ultra37000 optimal for implementing on board design changes using ISR without changing pinouts.

### Simple Timing Model

The Ultra37128 features a very simple timing model with predictable delays. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. The timing model allows for design changes with ISR without causing changes to system performance.

### Low Power Operation

Each Logic Block of the Ultra37128 can be configured as either High-Speed (default) or Low-Power. In the Low-Power mode, the logic block consumes 50% less power (9.3 mA max.) and slows down by 5 ns.

### Output Slew Rate Control

Each output can be configured with either a fast edge rate (default) for high performance, or a slow edge rate for added noise reduction. In the fast edge rate mode, outputs switch at 3V/ns max. and in the slow edge rate mode, outputs switch at 1V/ns max. There is a 2ns delay for I/Os using the slow edge rate mode.

### In System Reprogramming

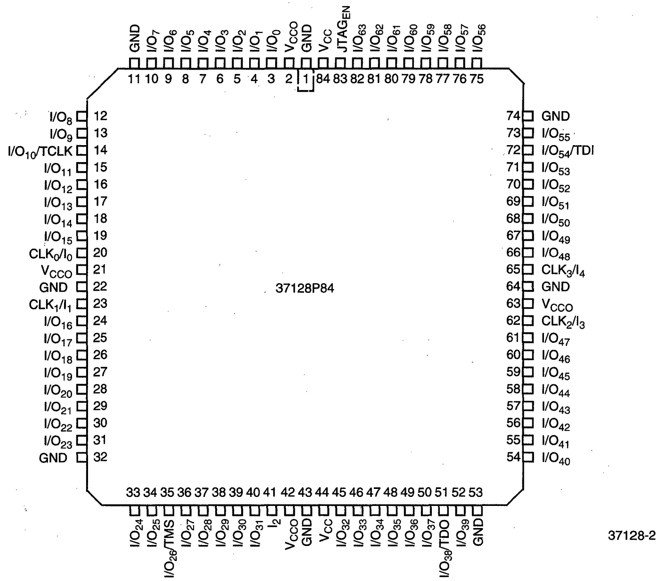
The Ultra37128 can be programmed in system using IEEE 1149.1 complaint JTAG programming protocol. The Ultra37128 can also be programmed on a number of traditional parallel programmers including Cypress's *Impulse3*™ programmer and industry standard third party programmers. For an overview of ISR programming, refer to the Ultra37000 Family data sheet and for ISR cable and software specifications, refer to the InSRkit: ISR programming data sheet (CY3600).

### Design Tools

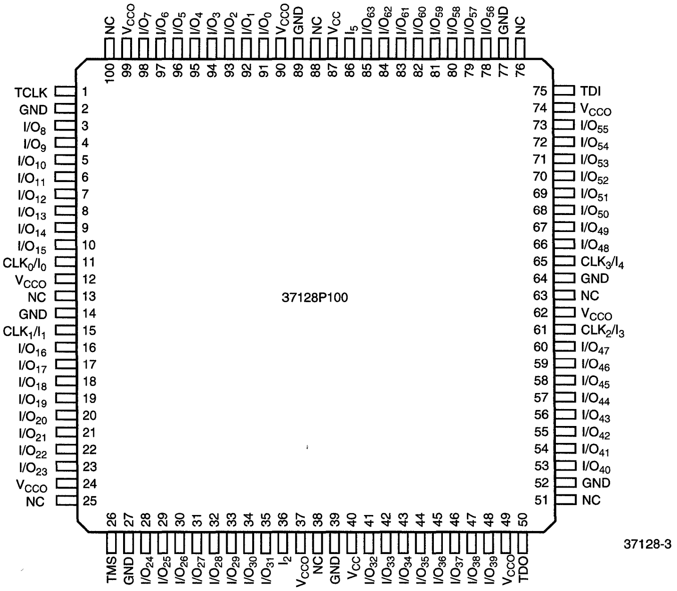
Development software for the Ultra37128 is available from Cypress's *Warp*™ or third party bolt-in software packages as well as a number of third party development packages. Please refer to the *Warp* or third party tool support data sheets for further information.

## Pin Configurations

**84-pin PLCC**
**Top View**

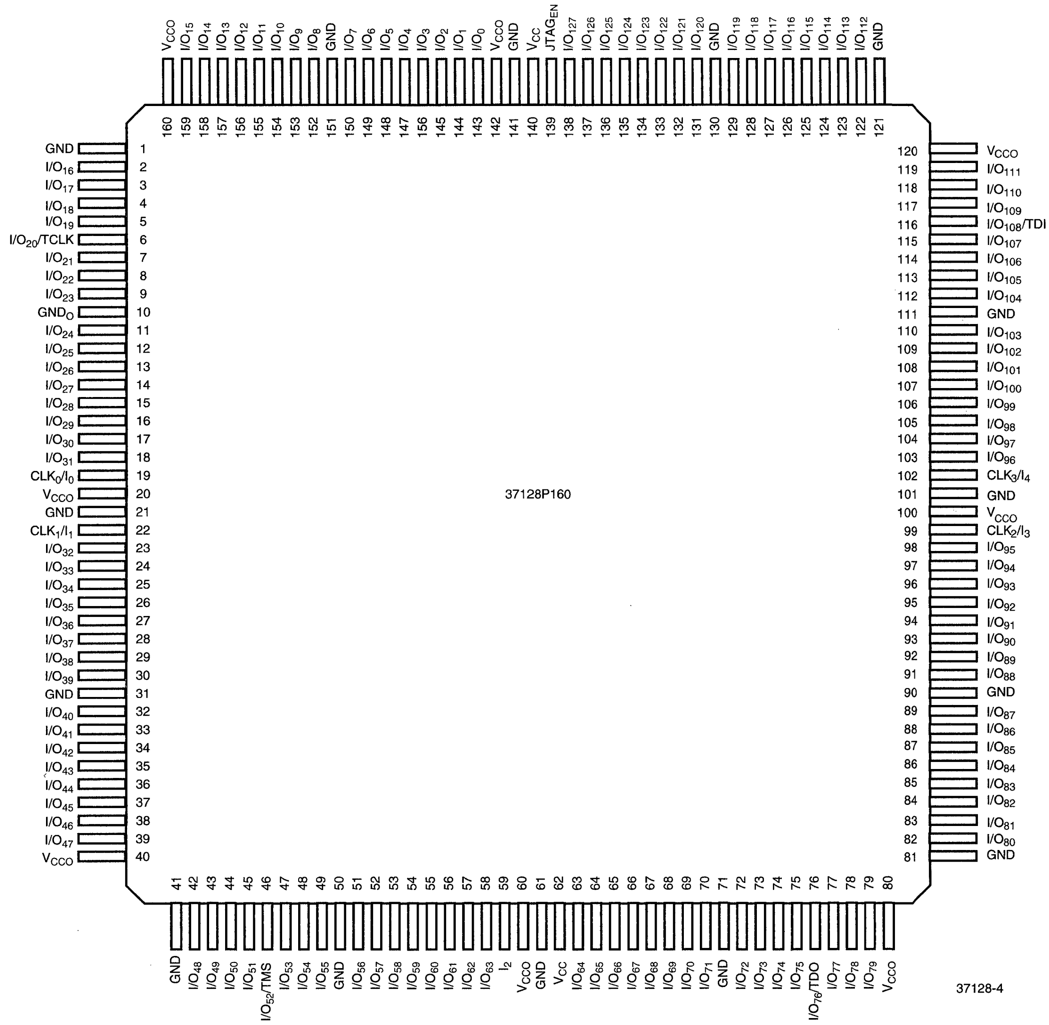


**100-pin TQFP**
**Top View**



## Pin Configurations (continued)

**160-pin TQFP**
**Top View**



37128-4

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | –65°C to +150°C |
| Ambient Temperature with Power Applied | –55°C to +125°C |
| Supply Voltage to Ground Potential | –0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | –0.5V to +7.0V |
| DC Input Voltage | –0.5V to +7.0V |
| DC Program Voltage | 5V±5% |
| Output Current into Outputs | 16 mA |
| Static Discharge Voltage (per MIL-STD-883, Method 3015) | >2001V |
| Latch-Up Current | >200 mA |

## Operating Range[2]

| Range | Ambient Temperature[2] | Output Condition | $V_{CC}$ | $V_{CCO}$ |
|---|---|---|---|---|
| Commercial | 0°C to +70°C | 5V | 5V ± 5% | 5V ± 5% |
| | | 3.3V | 5V ± 5% | 3.3V ± 0.3V |
| Industrial | -40°C to +85°C | 5V | 5V ± 10% | 5V ± 10% |
| | | 3.3V | 5V ± 10% | 3.3V ± 0.3V |
| Military[3] | -55°C to +125°C | 5V | 5V ± 10% | 5V ± 10% |
| | | 3.3V | 5V ± 10% | 3.3V ± 0.3V |

Shaded areas contain advance information.

## Electrical Characteristics Over the Operating Range

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = –3.2 mA[4] | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA[4] | | 0.5 | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH voltage for all inputs[5] | | 2.0 | 5.25 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW voltage for all inputs[5] | | –0.5 | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I$ = $V_{CC}$ | | –10 | 10 | μA |
| $I_{OZ}$ | Output Leakage Current | GND ≤ $V_O$ ≤ $V_{CC}$, Output Disabled | | –50 | 50 | μA |
| $I_{OS}$ | Output Short Circuit Current[6, 7] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | –30 | –160 | mA |
| $I_{CC\text{-}HS}$ | Power Supply Current[8] Per Logic Block - High Speed Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 18.7 | mA |
| $I_{CC\text{-}LP}$ | Power Supply Current[8] Per Logic Block - Low Power Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 9.3 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | μA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | –75 | | μA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | +500 | μA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | –500 | μA |

**Notes:**

[2] Normal Programming Conditions apply across Ambient Temperature Rnage for specified programming methods. For more information on programming the Ultra37000 family devices see the Ultra37000 family data sheet.

[3] $T_A$ is the "instant on" case temperature.

[4] $I_{OH}$ = –2 mA, $I_{OL}$ = 2 mA for SDO.

[5] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[6] Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[7] Tested initially and after any design or process changes that may affect these parameters.

[8] Measured with 16-bit counter programmed into the logic block. Total device power calculated by summing the $I_{CC}$ specifications for the mode of operation of each logic block.

## Inductance[7]

| Parameter | Description | Test Conditions | 84-lead PLCC | 100-lead TQFP | 160-Lead TQFP | Unit |
|---|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | 5 | 9 | nH |

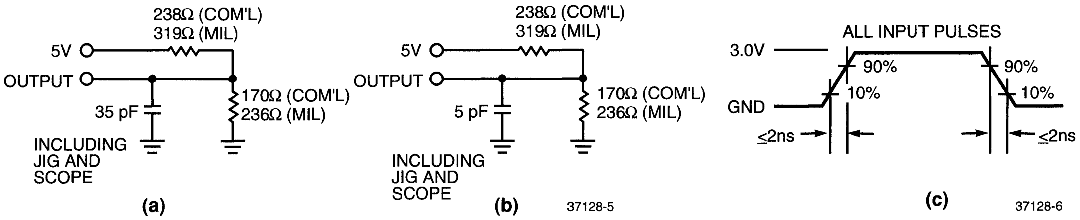
## Capacitance[7]

| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{I/O}$ | Input/Output Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 12 | pF |

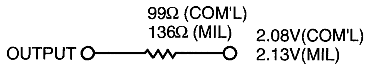
## Endurance Characteristics[7]

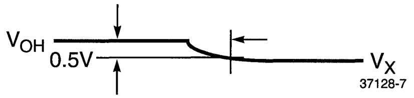
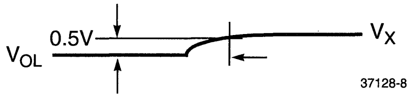
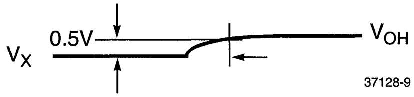
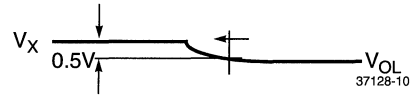
| Parameter | Description | Test Conditions | Min. | Typ. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions[2] | 1,000 | 10,000 | Cycles |

## AC Test Loads and Waveforms



Equivalent to: THÉVENIN EQUIVALENT



| Parameter[9] | $V_X$ | Output Waveform--Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | 1.5V |  |
| $t_{EA(-)}$ | $V_{the}$ |  |

**(d) Test Waveforms**

**Note:**

9. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[10]

| Parameter | Description | 37128-167 | | 37128-154 | | 37128-125 | | 37128-83 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | | | | | |
| $t_{PD}$[11] | Input to Combinatorial Output | | 6.5 | | 7.5 | | 10 | | 15 | ns |
| $t_{PDL}$[11] | Input to Output Through Transparent Input or Output Latch | | 9 | | 10 | | 13 | | 18 | ns |
| $t_{PDLL}$[11] | Input to Output Through Transparent Input and Output Latches | | 11 | | 12 | | 15 | | 19 | ns |
| $t_{EA}$[12] | Input to Output Enable | | 10 | | 11 | | 14 | | 19 | ns |
| $t_{ER}$ | Input to Output Disable | | 10 | | 11 | | 14 | | 19 | ns |
| **Input Register Parameters** | | | | | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[7] | 2.5 | | 2.5 | | 3 | | 4 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[7] | 2.5 | | 2.5 | | 3 | | 4 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 2 | | 2 | | 3 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 2 | | 2 | | 3 | | ns |
| $t_{ICO}$[11] | Input Register Clock or Latch Enable to Combinatorial Output | | 11 | | 11 | | 14 | | 19 | ns |
| $t_{ICOL}$[11] | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 12 | | 12 | | 16 | | 21 | ns |
| **Synchronous Clocking Parameters** | | | | | | | | | | |
| $t_{CO}$[12] | Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output | | 4.5 | | 4.5 | | 6.5 | | 8 | ns |
| $t_S$[11] | Set-Up Time from Input to Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 3.5 | | 4.0 | | 5.5 | | 8 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{CO2}$[11, 12] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Combinatorial Output Delay (Through Logic Array) | | 10 | | 11 | | 14 | | 19 | ns |
| $t_{SCS}$[11] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable (Through Logic Array) | 6 | | 6.5 | | 8 | | 12 | | ns |
| $t_{SL}$[11] | Set-Up Time from Input Through Transparent Latch to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 7.5 | | 8.0 | | 10 | | 15 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 0 | | 0 | | 0 | | 0 | | ns |

**Notes:**

[10] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

[11] Logic Blocks operating in low power mode, add $t_{LP}$ to this spec.

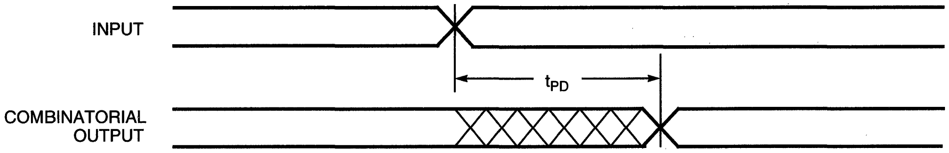
[12] Outputs using Slow Output Slew Rate, add $t_{SLEW}$ to this spec.

## Switching Characteristics Over the Operating Range[10] (continued)

| Parameter | Description | 37128-167 | | 37128-154 | | 37128-125 | | 37128-83 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Product Term Clocking Parameters** | | | | | | | | | | |
| $t_{COPT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output | | 9 | | 10 | | 12 | | 17 | ns |
| $t_{SPT}$[11] | Set-Up Time from Input to register Product Term Clock or Latch Enable (PTCLK) | 2.5 | | 2.5 | | 3 | | 3 | | ns |
| $t_{HPT}$ | Register or Latch Data Hold Time | 2.5 | | 2.5 | | 3 | | 3 | | ns |
| $t_{ISPT}$[11] | Set-Up Time from buried register used as an input register from Input to Product Term Clock or Latch Enable (PT-CLK) | | -2 | | -2 | | -2 | | -2 | ns |
| $t_{IHPT}$ | Buried Register used as an input register or Latch Data Hold Time | | 5.5 | | 6.5 | | 9 | | 14 | ns |
| $t_{CO2PT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output Delay (Through Logic Array) | | 13 | | 14 | | 16 | | 21 | ns |
| **Pipelined Mode Parameters** | | | | | | | | | | |
| $t_{ICS}$[11, 12] | Input Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) | 5.5 | | 6 | | 8 | | 12 | | ns |
| **Operating Frequency Parameters** | | | | | | | | | | |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[7] | 167 | | 154 | | 125 | | 83 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$) | 200 | | 200 | | 153.8 | | 125 | | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ or $1/(t_{WL} + t_{WH})$) | 125 | | 118 | | 83.3 | | 62.5 | | MHz |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 154 | | 154 | | 125 | | 66.6 | | MHz |
| **Reset/Preset Parameters** | | | | | | | | | | |
| $t_{RW}$ | Asynchronous Reset Width[7] | 8 | | 8 | | 10 | | 15 | | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[7] | 10 | | 10 | | 12 | | 17 | | ns |
| $t_{RO}$[11, 12] | Asynchronous Reset to Output | | 14 | | 14 | | 16 | | 21 | ns |
| $t_{PW}$ | Asynchronous Preset Width[7] | 8 | | 8 | | 10 | | 15 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[7] | 10 | | 10 | | 12 | | 17 | | ns |
| $t_{PO}$[11, 12] | Asynchronous Preset to Output | | 14 | | 14 | | 16 | | 21 | ns |
| **User Option Parameters** | | | | | | | | | | |
| $t_{LP}$ | Low Power Adder | 5 | | 5 | | 6 | | 6 | | ns |
| $t_{SLEW}$ | Slow Output Slew Rate Adder | 2 | | 2 | | 2 | | 2 | | ns |
| **Tap Controller Parameter** | | | | | | | | | | |
| $f_{TAP}$ | Tap Controller Frequency | 20 | | | 20 | | 20 | | 20 | MHz |

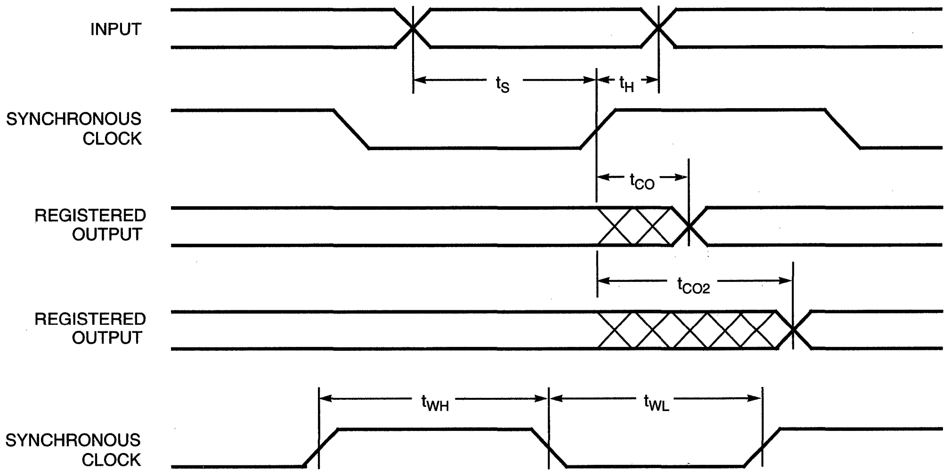
## Switching Waveforms

### Combinatorial Output



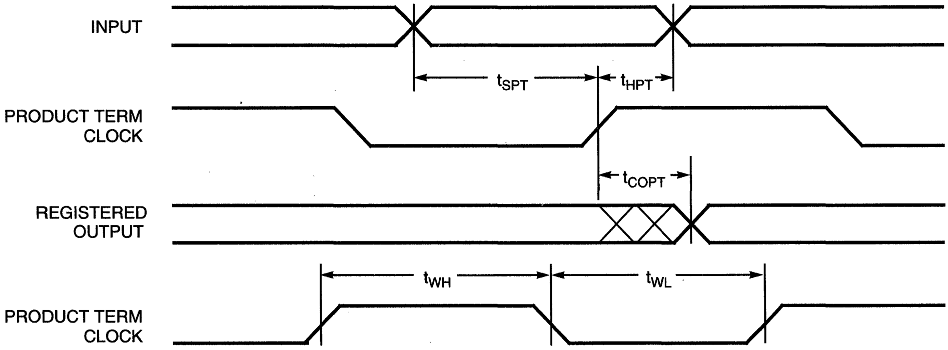
37128-11

### Registered Output with Synchronous Clocking



37128-12

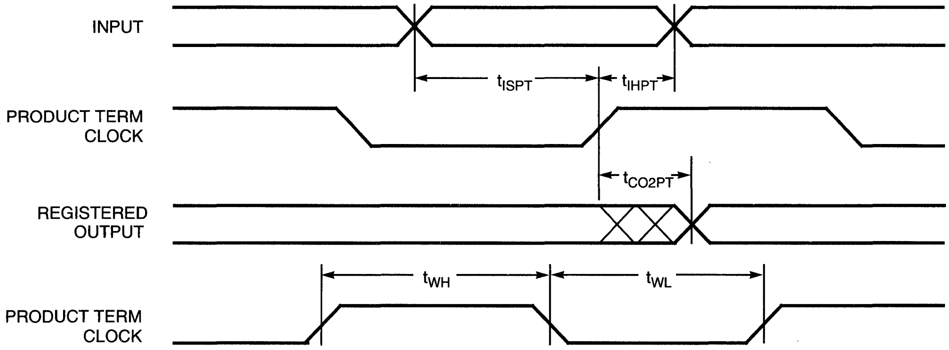
### Registered Output with Product Term Clocking Input going through the Array



37128-13

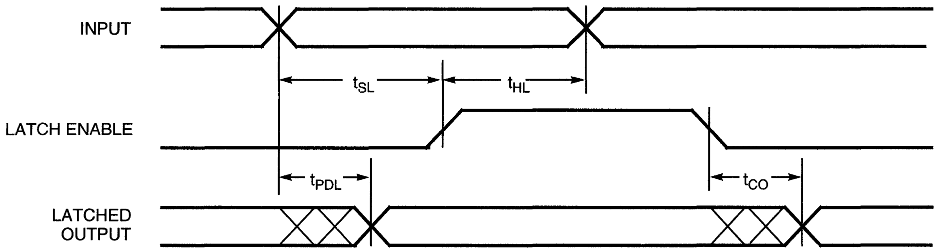
## Switching Waveforms (continued)

### Registered Output with Product Term Clocking Input coming from Adjacent Buried Register



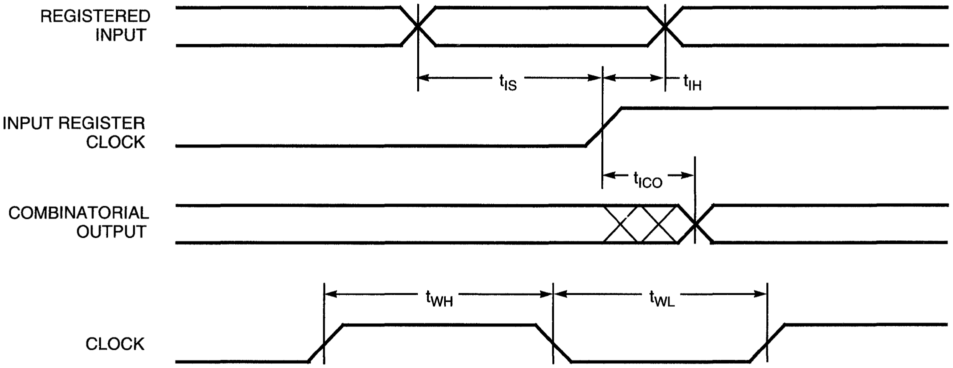
37128-14

### Latched Output



37128-15

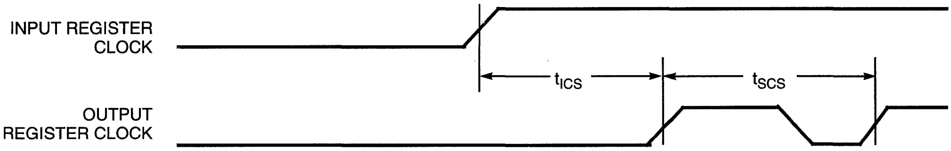
### Registered Input



37128-16

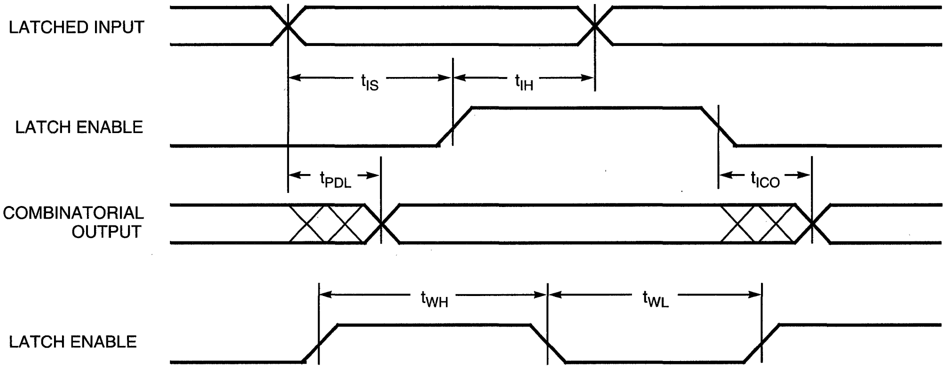
## Switching Waveforms (continued)

### Clock to Clock



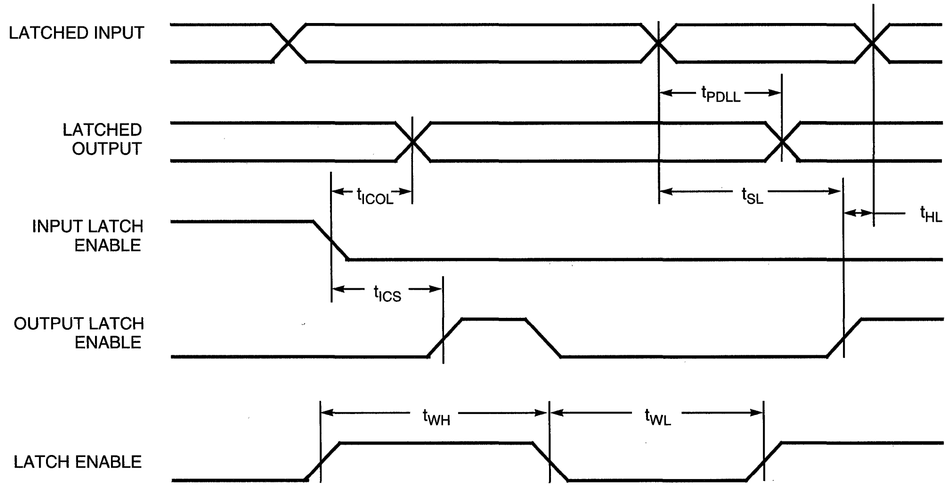
37128-17

### Latched Input



37128-18

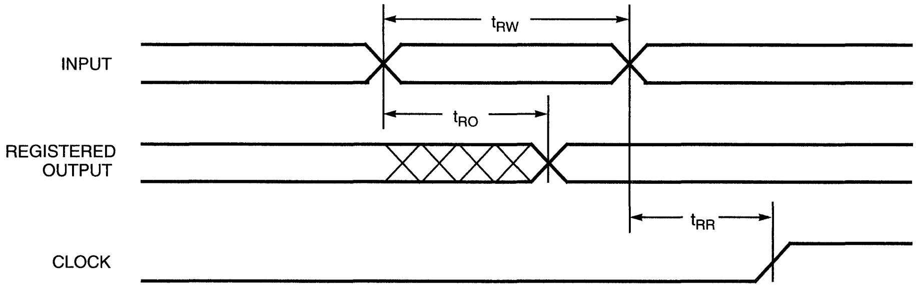
### Latched Input and Output



37128-19

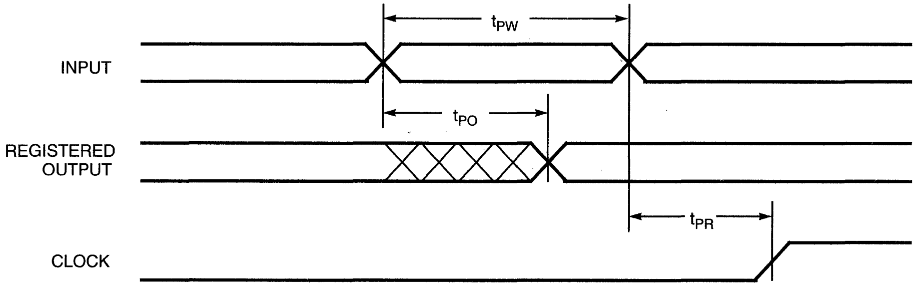
## Switching Waveforms (continued)

### Asynchronous Reset



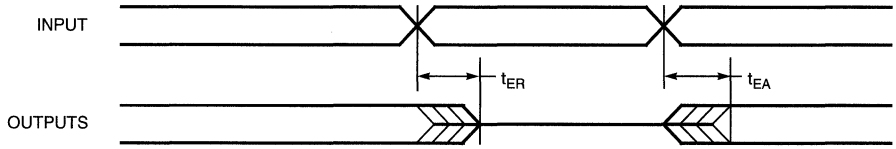
37128-20

### Asynchronous Preset



37128-21

### Output Enable/Disable



37128–22

## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 167 | CY37128P84–167JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128P100–167AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–167AC | A160 | 160-Pin Thin Quad Flatpack | |
| 154 | CY37128P84–154JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128P100–154AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–154AC | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128P84–154JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Industrial |
| | CY37128P100–154AI | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–154AI | A160 | 160-Pin Thin Quad Flatpack | |
| 125 | CY37128P84–125JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128P100–125AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–125AC | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128P84–125JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Indusrtial |
| | CY37128P100–125AI | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–125AI | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128P160–125GMB | G160 | 160-Pin Grid Array | Military |
| | CY37128P160–125UMB | U160 | 160-Pin Ceramic Quad Flatpack | |
| 83 | CY37128P84–83JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128P100–83AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–83AC | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128P84–83JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Indusrtial |
| | CY37128P100–83AI | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128P160–83AI | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128P160-83GMB | G160 | 160-Pin Grid Array | Military |
| | CY37128P160-83UMB | U160 | 160-Pin Ceramic Quad Flatpack | |

Shaded areas contain advance information.

ISR, UltraLogic, FLASH370, Ultra37000, and *Warp2*Sim are trademarks of Cypress Semiconductor Corporation. *Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

Document #: 38-00558

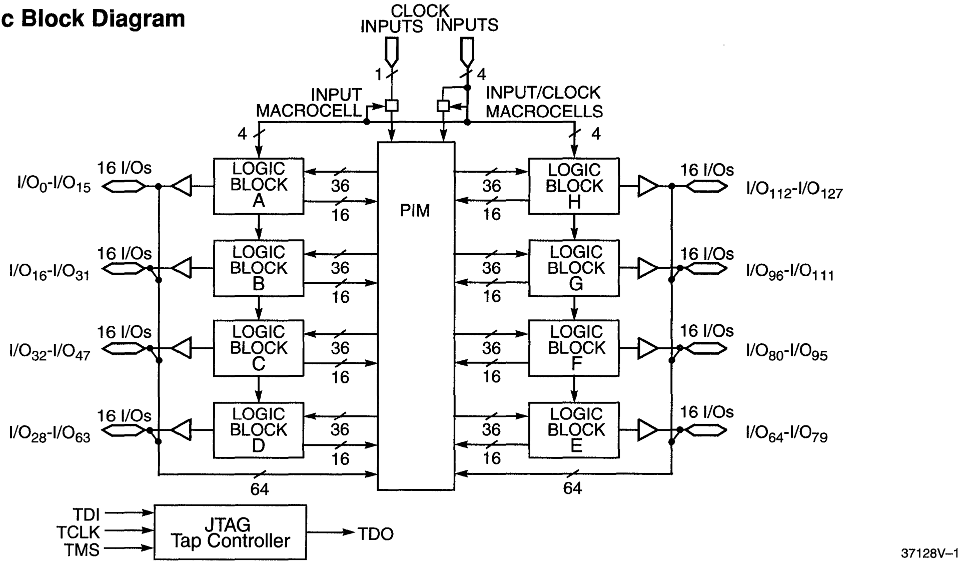
CYPRESS **PRELIMINARY** **Ultra37128V**

# UltraLogic™ 3.3V 128-Macrocell ISR™ CPLD

## Features

- **128 macrocells in eight logic blocks**
- **3.3V In-System Reprogrammable (ISR™)**
  - **—JTAG compliant on board programming**
  - **—Design changes don't cause pinout changes**
  - **—Design changes don't cause timing changes**
- **I/O Intensive Version features an I/O for every Macrocell**
  - **—128 I/O Macrocells**
  - **—128 I/O pins plus 5 dedicated inputs including 4 clock inputs**
- **Register Intensive Version**
  - **—64 buried Macrocells and 64 I/O Macrocells**
  - **—64 I/O pins and 5 dedicated inputs including 4 clock inputs**
- **IEEE 1149.1 JTAG boundary scan**
- **High speed**
  - **—$f_{MAX}$ = 125 MHz**
  - **—$t_{PD}$ = 10 ns**
  - **—$t_S$ = 5.5 ns**
  - **—$t_{CO}$ = 6.5 ns**
- **JEDEC standard 3.3V operation**
  - **—3.3V ISR**
  - **—5V tolerant**
- **Product-term clocking**
- **Programmable slew rate control on individual I/Os**
- **Low power option on individual logic block basis**
- **Bus Hold capabilities on all I/Os**
- **Simple Timing Model**
- **PCI compliant[1]**
- **Available in 84-pin PLCC, 100-pin TQFP and 160-pin PQFP packages**
- **Pinout compatible Ultra37128 5V device**

## Logic Block Diagram



## Selection Guide

| | Ultra37128V-125 | Ultra37128V-83 |
|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | 10 | 15 |
| Minimum Set-Up, $t_S$ (ns) | 5.5 | 8 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 6.5 | 8 |
| Typical Supply Current, $I_{CC}$ (mA) in Low Power Mode | 75 | 75 |

**Note:**

[1] Due to the 5V tolerant nature of the I/Os, the I/Os are not clamped to $V_{CC}$

## Functional Description

The Ultra37128V is an In-System Reprogrammable (ISR) Complex Programmable Logic Device (CPLD) and is part of the Ultra37000™ family of high-density, high-speed CPLDs. Like all members of the Ultra37000 family, the Ultra37128V is designed to bring the ease of use and high performance of the 22V10 to high-density PLDs.

The 128 macrocell Ultra37128V is available in register intensive and I/O intensive versions. The Ultra37128VP84 and Ultra37128VP100 feature 64 Buried Macrocells and 64 I/O Macrocells for register intensive designs which require small footprint devices. The Ultra37128VP160 I/O intensive device has an I/O pin for each macrocell.

For a more detailed description of the architecture and features of the Ultra37128V see the Ultra37000 family data sheet.

### Fully Routable with 100% Logic Utilization

The Ultra37128V is designed with a robust routing architecture which allows utilization of the entire device, with a fixed pinout. This makes Ultra37000 optimal for implementing on board design changes using ISR without changing pinouts.

### Simple Timing Model

The Ultra37128V features a very simple timing model with predictable delays. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. The timing model allows for design changes with ISR without causing changes to system performance.

### Low Power Operation

Each Logic Block of the Ultra37128V can be configured as either High-Speed (default) or Low-Power. In the Low-Power mode, the logic block consumes 50% less power (9.3 mA max.) and slows down by 6 ns.

### Output Slew Rate Control

Each output can be configured with either a fast edge rate (default) for high performance, or a slow edge rate for added noise reduction. In the fast edge rate mode, outputs switch at 3V/ns max. and in the slow edge rate mode, outputs switch at 1V/ns max. There is a 2-ns delay for I/Os using the slow edge rate mode.

### In System Reprogramming

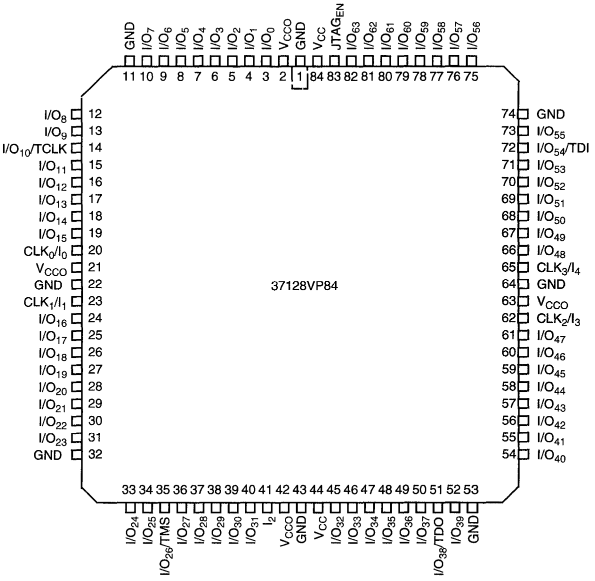
The Ultra37128V can be programmed in system using IEEE 1149.1 compliant JTAG programming protocol. The Ultra37128V can also be programmed on a number of traditional parallel programmers including Cypress's *Impulse3*™ programmer and industry standard third-party programmers. For an overview of ISR programming, refer to the Ultra37000 Family data sheet and for ISR cable and software specifications, refer to InSRkit: ISR programming data sheet (CY3600).

### Design Tools

Development software for the Ultra37128V is available from Cypress's *Warp*™ or third-party bolt-in software packages as well as a number of third-party development packages. Please refer to the *Warp* or third-party tool support data sheets for further information.

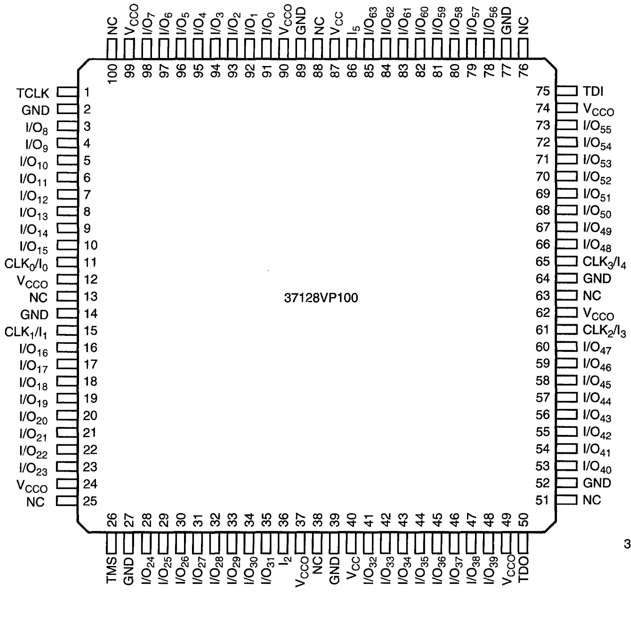
## Pin Configurations

### 84-pin PLCC
### Top View



37128V-2

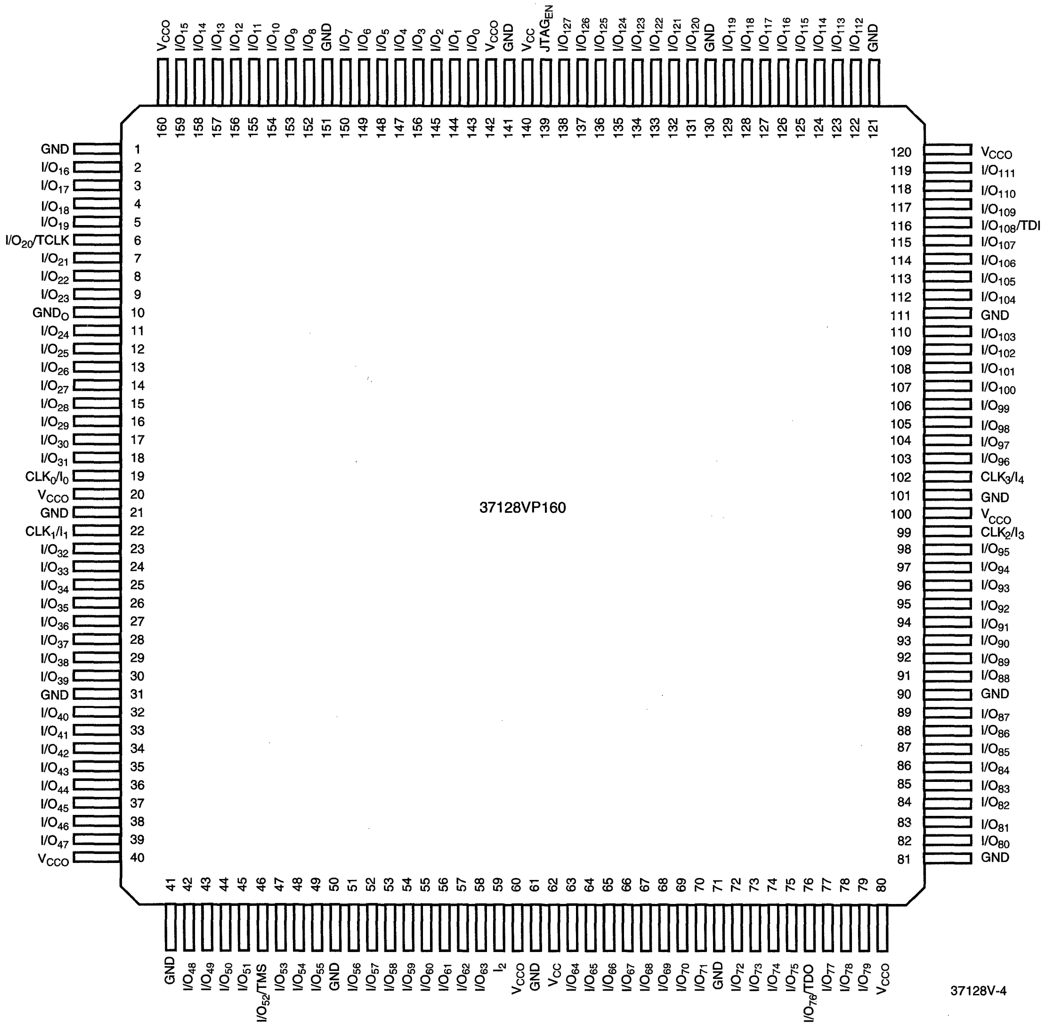
### 100-pin TQFP
### Top View



37128V-3

## Pin Configurations (continued)

**160-pin TQFP**
**Top View**



37128V-4

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................ −65°C to +150°C

Ambient Temperature with
Power Applied ............................................ −55°C to +125°C

Supply Voltage to Ground Potential ............... −0.5V to +7.0V

DC Voltage Applied to Outputs
in High Z State ............................................. −0.5V to +7.0V

DC Input Voltage ........................................... −0.5V to +7.0V

DC Program Voltage ................................................ 5V±5%

Output Current into Outputs ........................................ 8 mA

Static Discharge Voltage .......................................... >2001V
(per MIL-STD-883, Method 3015)

Latch-Up Current ................................................... >200 mA

## Operating Range[2]

| Range | Ambient Temperature[2] | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 3.3V ± 0.3V |
| Industrial | -40°C to +85°C | 3.3V ± 0.3V |
| Military[3] | -55°C to +125°C | 3.3V ± 0.3V |

Shaded areas contain advanced information.

## Electrical Characteristics Over the Operating Range

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = −4 mA[4] (COM'L) | 2.4 | | V |
| | | | $I_{OH}$ = −3 mA[4] (MIL) | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 8 mA[4](COM'L) | | 0.5 | V |
| | | | $I_{OL}$ = 6 mA[4] (MIL) | | | |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH voltage for all inputs[5] | | 2.0 | 5.25 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW voltage for all inputs[5] | | −0.5 | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I$ = $V_{CC}$ | | −10 | 10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = GND or $V_{CC}$, Output Disabled | | −50 | 50 | µA |
| $I_{OS}$ | Output Short Circuit Current[6, 7] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | −30 | −160 | mA |
| $I_{CC-HS}$ | Power Supply Current[8] Per Logic Block - High Speed Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 18.7 | mA |
| $I_{CC-LP}$ | Power Supply Current[8] Per Logic Block - Low Power Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 9.3 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | µA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | −75 | | µA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | +500 | µA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | −500 | µA |

**Notes:**

[2] Normal Programming Conditions apply across Ambient Temperature Range for specified programming methods. For more information on programming the Ultra37000 family devices see the Ultra37000 family data sheet.
[3] $T_A$ is the "instant on" case temperature.
[4] $I_{OH}$ = −2 mA, $I_{OL}$ = 2 mA for SDO.
[5] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.
[6] Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
[7] Tested initially and after any design or process changes that may affect these parameters.
[8] Measured with 16-bit counter programmed into the logic block. Total device power calculated by summing the $I_{CC}$ specifications for the mode of operation of each logic block.

## Inductance[7]

| Parameter | Description | Test Conditions | 84-lead PLCC | 100-lead TQFP | 160-Lead TQFP | Unit |
|---|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | 5 | 9 | nH |

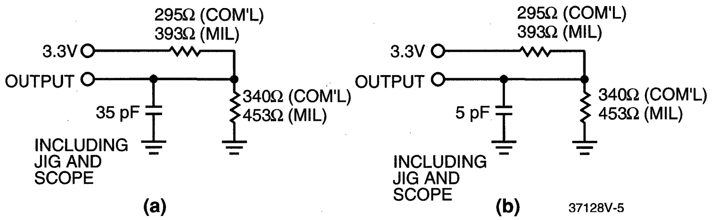
## Capacitance[7]

| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{I/O}$ | Input/Output Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 12 | pF |

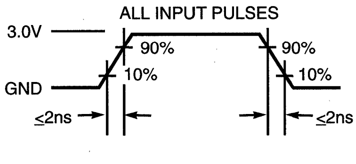
## Endurance Characteristics[7]

| Parameter | Description | Test Conditions | Min. | Typ. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions[2] | 1,000 | 10,000 | Cycles |

## AC Test Loads and Waveforms

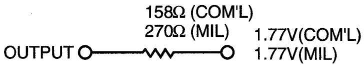


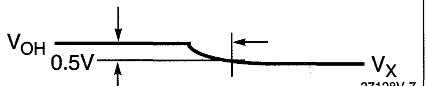
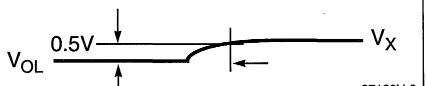
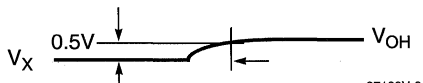
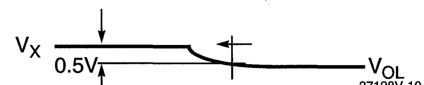
(a) (b) 37128V-5



(c) 37128V-6

Equivalent to: THÉVENIN EQUIVALENT



| Parameter[9] | $V_X$ | Output Waveform--Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  37128V-7 |
| $t_{ER(+)}$ | 2.6V |  37128V-8 |
| $t_{EA(+)}$ | 1.5V |  37128V-9 |
| $t_{EA(-)}$ | $V_{the}$ |  37128V-10 |

**(d) Test Waveforms**

**Note:**

9. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[10]

| Parameter | Description | 37128V-125 | | 37128V-83 | | Unit |
|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | |
| $t_{PD}$[11] | Input to Combinatorial Output | | 10 | | 15 | ns |
| $t_{PDL}$[11] | Input to Output Through Transparent Input or Output Latch | | 13 | | 18 | ns |
| $t_{PDLL}$[11] | Input to Output Through Transparent Input and Output Latches | | 15 | | 19 | ns |
| $t_{EA}$[12] | Input to Output Enable | | 14 | | 19 | ns |
| $t_{ER}$ | Input to Output Disable | | 14 | | 19 | ns |
| **Input Register Parameters** | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[7] | 3 | | 4 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[7] | 3 | | 4 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 3 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 3 | | ns |
| $t_{ICO}$[11] | Input Register Clock or Latch Enable to Combinatorial Output | | 14 | | 19 | ns |
| $t_{ICOL}$[11] | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 16 | | 21 | ns |
| **Synchronous Clocking Parameters** | | | | | | |
| $t_{CO}$[12] | Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output | | 6.5 | | 8 | ns |
| $t_S$[11] | Set-Up Time from Input to Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 5.5 | | 8 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | ns |
| $t_{CO2}$[11, 12] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Combinatorial Output Delay (Through Logic Array) | | 14 | | 19 | ns |
| $t_{SCS}$[11] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable (Through Logic Array) | 8 | | 12 | | ns |
| $t_{SL}$[11] | Set-Up Time from Input Through Transparent Latch to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 10 | | 15 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 0 | | 0 | | ns |
| **Product Term Clocking Parameters** | | | | | | |
| $t_{COPT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output | | 12 | | 17 | ns |
| $t_{SPT}$[11] | Set-Up Time from Input to Product Term Clock or Latch Enable (PTCLK) | 3 | | 3 | | ns |
| $t_{HPT}$ | Register or Latch Data Hold Time | 3 | | 3 | | ns |
| $t_{ISPT}$[11] | Set-Up Time for buried register used as an input register from Input to Product Term Clock or Latch Enable (PTCLK) | | 2 | | 2 | ns |

**Notes:**

[10] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

[11] Logic Blocks operating in low power mode, add $t_{LP}$ to this spec.

[12] Outputs using Slow Output Slew Rate, add $t_{SLEW}$ to this spec.

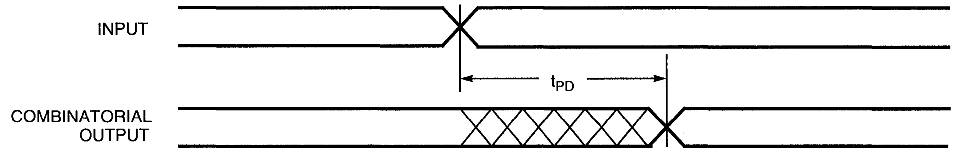
3

## Switching Characteristics Over the Operating Range[10] (continued)

| Parameter | Description | 37128V-125 |  | 37128V-83 |  | Unit |
|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. |  |
| $t_{IHPT}$ | Buried Register used as an input register or Latch Data Hold Time |  | 9 |  | 14 | ns |
| $t_{CO2PT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output Delay (Through Logic Array) |  | 16 |  | 21 | ns |
| **Pipelined Mode Parameters** |  |  |  |  |  |  |
| $t_{ICS}$[11, 12] | Input Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) | 8 |  | 12 |  | ns |
| **Operating Frequency Parameters** |  |  |  |  |  |  |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[7] | 125 |  | 83 |  | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$) | 153.8 |  | 125 |  | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ or $1/(t_{WL} + t_{WH})$) | 83.3 |  | 62.5 |  | MHz |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 125 |  | 66.6 |  | MHz |
| **Reset/Preset Parameters** |  |  |  |  |  |  |
| $t_{RW}$ | Asynchronous Reset Width[7] | 10 |  | 15 |  | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[7] | 12 |  | 17 |  | ns |
| $t_{RO}$[11, 12] | Asynchronous Reset to Output |  | 16 |  | 21 | ns |
| $t_{PW}$ | Asynchronous Preset Width[7] | 10 |  | 15 |  | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[7] | 12 |  | 17 |  | ns |
| $t_{PO}$[11, 12] | Asynchronous Preset to Output |  | 16 |  | 21 | ns |
| **User Option Parameters** |  |  |  |  |  |  |
| $t_{LP}$ | Low Power Adder | 6 |  | 6 |  | ns |
| $t_{SLEW}$ | Slow Output Slew Rate Adder | 2 |  | 2 |  | ns |
| **Tap Controller Parameter** |  |  |  |  |  |  |
| $f_{TAP}$ | Tap Controller Frequency |  | 20 |  | 20 | MHz |

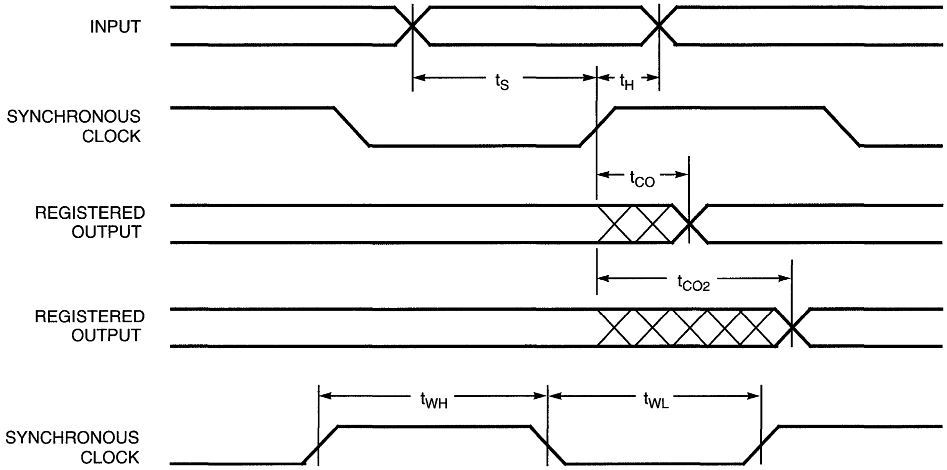
## Switching Waveforms

### Combinatorial Output



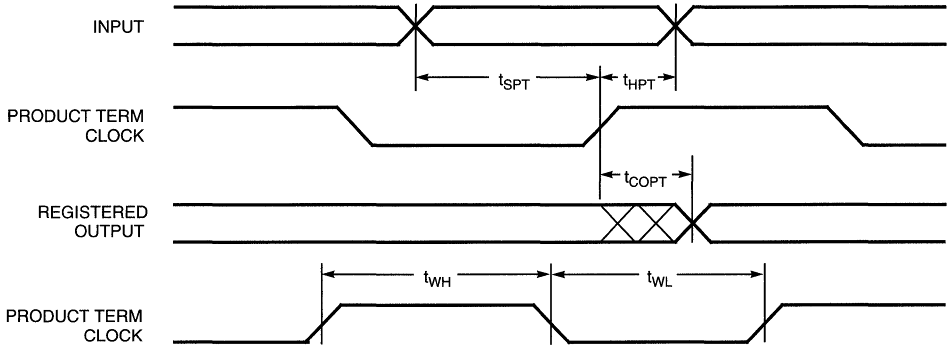
37128VV-11

### Registered Output with Synchronous Clocking



37128VV-12

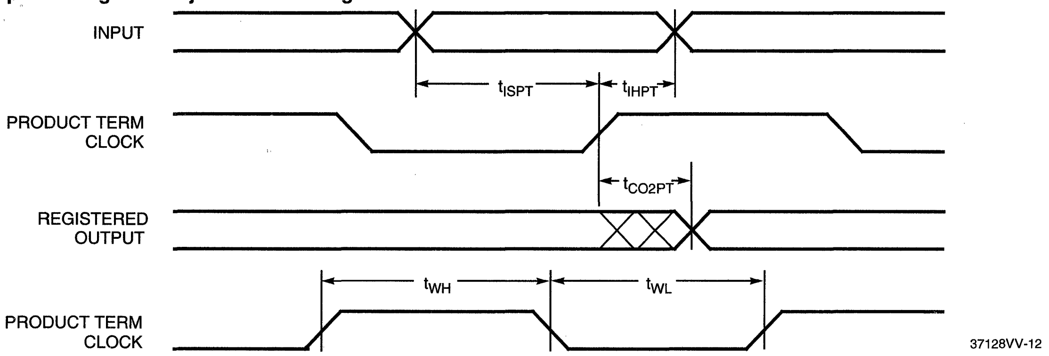
### Registered Output with Product Term Clocking Input going through the Array



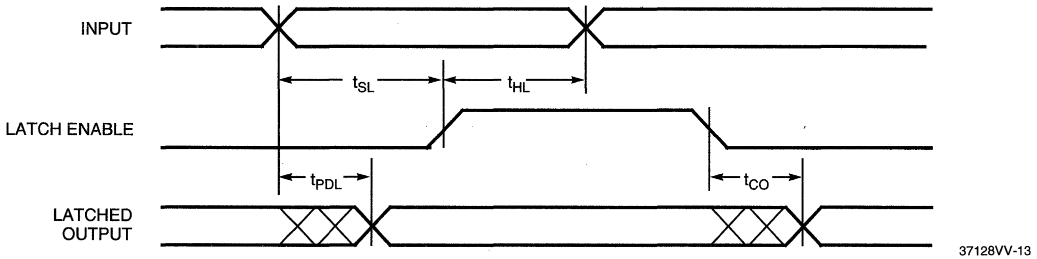
37128VV-12

## Switching Waveforms (continued)

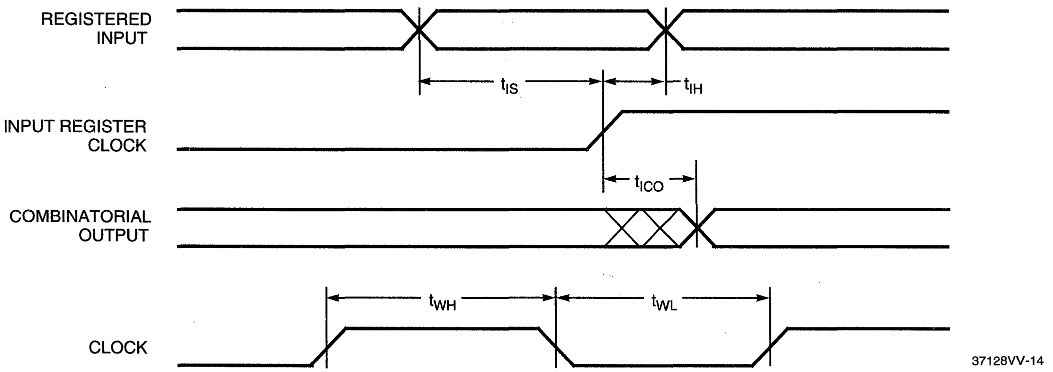
### Registered Output with Product Term Clocking Input coming from Adjacent Buried Register



### Latched Output

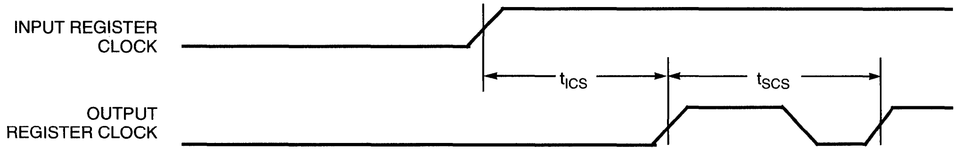


### Registered Input



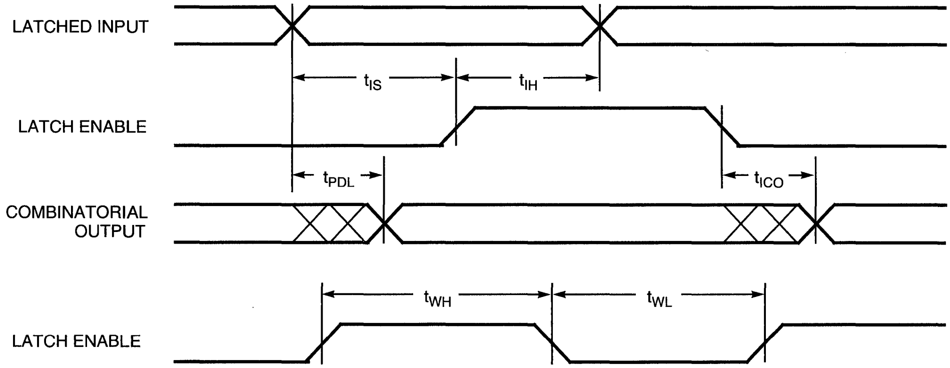
## Switching Waveforms (continued)

### Clock to Clock



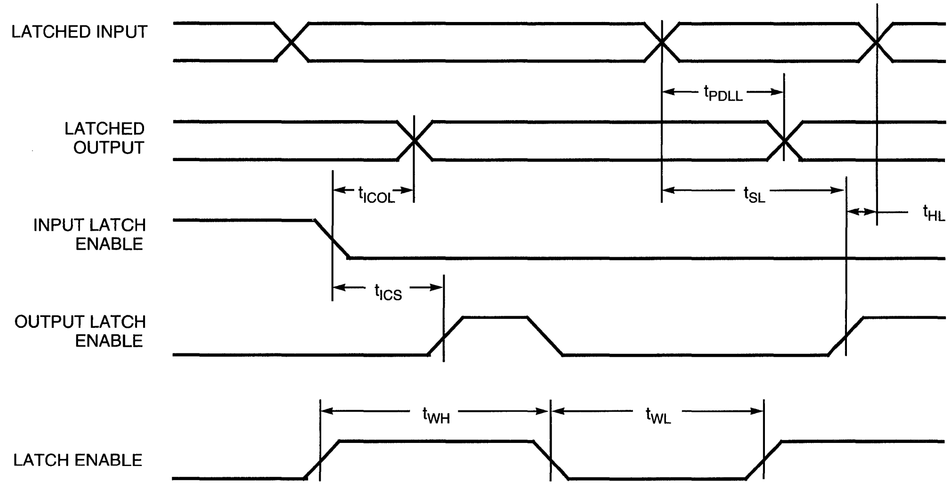
37128VV-15

### Latched Input



37128VV-16

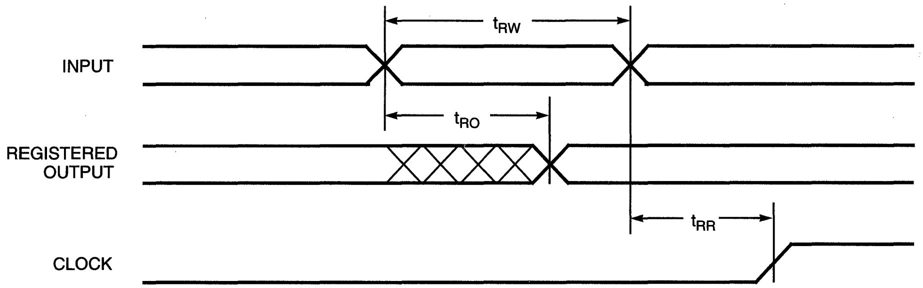
### Latched Input and Output



37128VV-17

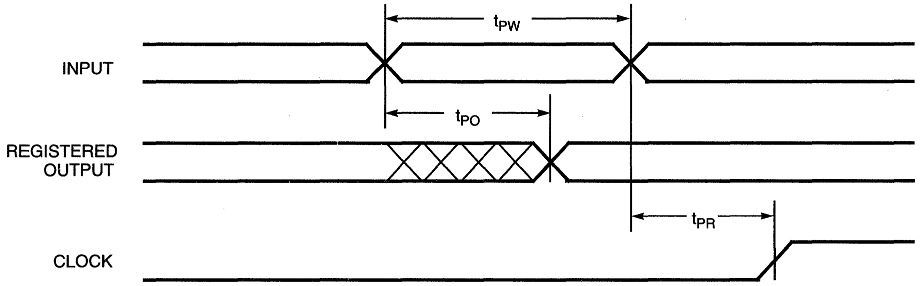
## Switching Waveforms (continued)

### Asynchronous Reset



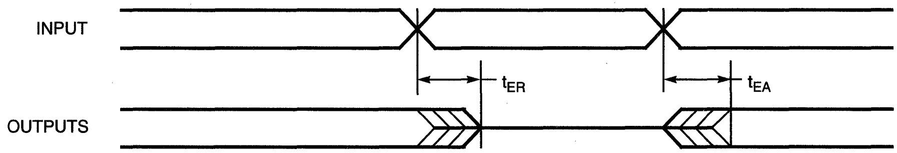
37128VV-18

### Asynchronous Preset



37128VV-19

### Output Enable/Disable



37128VV–20

## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 154 | CY37128VP84–154JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128VP100–154AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128VP160–154AC | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128VP84–154JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Industrial |
| | CY37128VP100–154AI | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128VP160–154AI | A160 | 160-Pin Thin Quad Flatpack | |
| 125 | CY37128VP84–125JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128VP100–125AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128VP160–125AC | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128VP84–125JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Industrial |
| | CY37128VP100–125AI | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128VP160–125AI | A160 | 160-Pin Thin Quad Flatpack | |
| 83 | CY37128VP84–83JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY37128VP100–83AC | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128VP160–83AC | A160 | 160-Pin Thin Quad Flatpack | |
| | CY37128VP84–83JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Industrial |
| | CY37128VP100–83AI | A100 | 100-Pin Thin Quad Flatpack | |
| | CY37128VP160–83AI | A160 | 160-Pin Thin Quad Flatpack | |

Shaded areas contain advanced information.

ISR, UltraLogic, FLASH370, Ultra37000, and *Warp2*Sim are trademarks of Cypress Semiconductor Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.
ABEL is a trademark of Data I/O Corporation.
LOG/iC is a trademark of Isdata Corporation.
CUPL is a trademark of Logical Devices Incorporated.

Document #: 38-00621

3

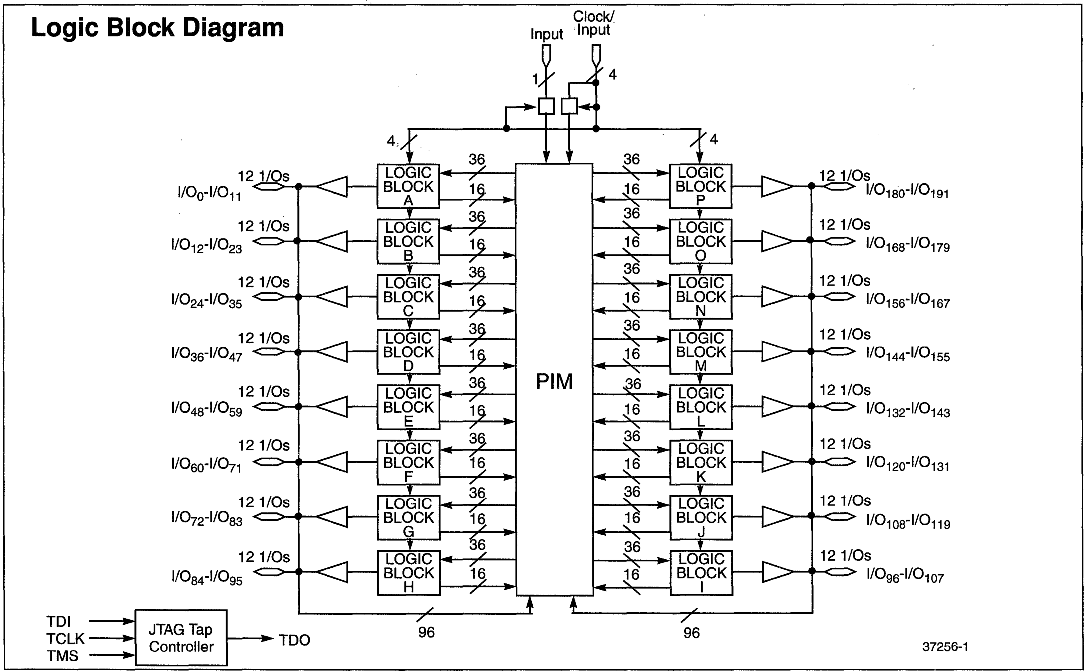
CYPRESS *PRELIMINARY* **Ultra37256**

# UltraLogic™ 256-Macrocell ISR™ CPLD

## Features

- 256 macrocells in sixteen logic blocks
- In-System Reprogrammable (ISR™)
  - —JTAG compliant on board programming
  - —Design changes don't cause pinout changes
  - —Design changes don't cause timing changes
- Up to 192 I/Os
  - —plus 5 dedicated inputs including 4 clock inputs
- High speed
  - —$f_{MAX}$ = 154 MHz
  - —$t_{PD}$ = 7.5 ns
  - —$t_S$ = 4.5 ns
  - —$t_{CO}$ = 5.0 ns
- Product-term clocking
- IEEE1149.1 JTAG boundary scan
- Programmable slew rate control on individual I/Os
- Low power option on individual logic block basis
- 5V and 3.3V I/O capability
- Bus Hold capabilities on all I/Os
- Simple Timing Model
- Fully PCI compliant[1]
- Available in 160-pin TQFP, 208-pin PQFP and 256-lead BGA packages
- Pinout compatible with all other Ultra37000 family CPLDs

## Logic Block Diagram



## Selection Guide

| | Ultra37256-154 | Ultra37256-125 | Ultra37256-83 |
|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | 7.5 | 10 | 15 |
| Minimum Set-Up, $t_S$ (ns) | 4.5 | 5.5 | 8 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 5.0 | 6.5 | 8 |
| Typical Supply Current, $I_{CC}$ (mA) in Low Power Mode | 120 | 120 | 120 |

**Note:**

[1] Due to the 5V tolerant nature of the I/Os, the I/Os are not clamped to Vcc.

## Functional Description

The Ultra37256 is an In-System Reprogrammable (ISR) Complex Programmable Logic Device (CPLD) and is part of the Ultra37000™ family of high-density, high-speed CPLDs. Like all members of the Ultra37000 family, the Ultra37256 is designed to bring the ease of use and high performance of the 22V10 to high-density PLDs.

The 256 macrocell Ultra37256 is available in register intensive and I/O intensive versions. The Ultra37256P160 features 128 Buried and 128 I/O Macrocells while the Ultra37256P208 features 96 Buried Macrocells and 160 I/O Macrocells, for register intensive designs which require small footprint devices. The Ultra37256P256 I/O intensive device has 192 I/O macrocells and 64 Buried macrocells.

For a more detailed description of the architecture and features of the Ultra37256 see the Ultra37000 family data sheet.

### Fully Routable with 100% Logic Utilization

The Ultra37256 is designed with a robust routing architecture which allows utilization of the entire device with a fixed pinout. This makes Ultra37000 optimal for implementing on board design changes using ISR without changing pinouts.

### Simple Timing Model

The Ultra37256 features a very simple timing model with predictable delays. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. The timing model allows for design changes with ISR without causing changes to system performance.

### Low Power Operation

Each Logic Block of the Ultra37256 can be configured as either High-Speed (default) or Low-Power. In the Low-Power mode, the logic block consumes 50% less power (9.3 mA max.) and slows down by 5 ns.

### Output Slew Rate Control

Each output can be configured with either a fast edge rate (default) for high performance, or a slow edge rate for added noise reduction. In the fast edge rate mode, outputs switch at 3V/ns max. and in the slow edge rate mode, outputs switch at 1V/ns max. There is a 2-ns delay for I/Os using the slow edge rate mode.

### In System Reprogramming

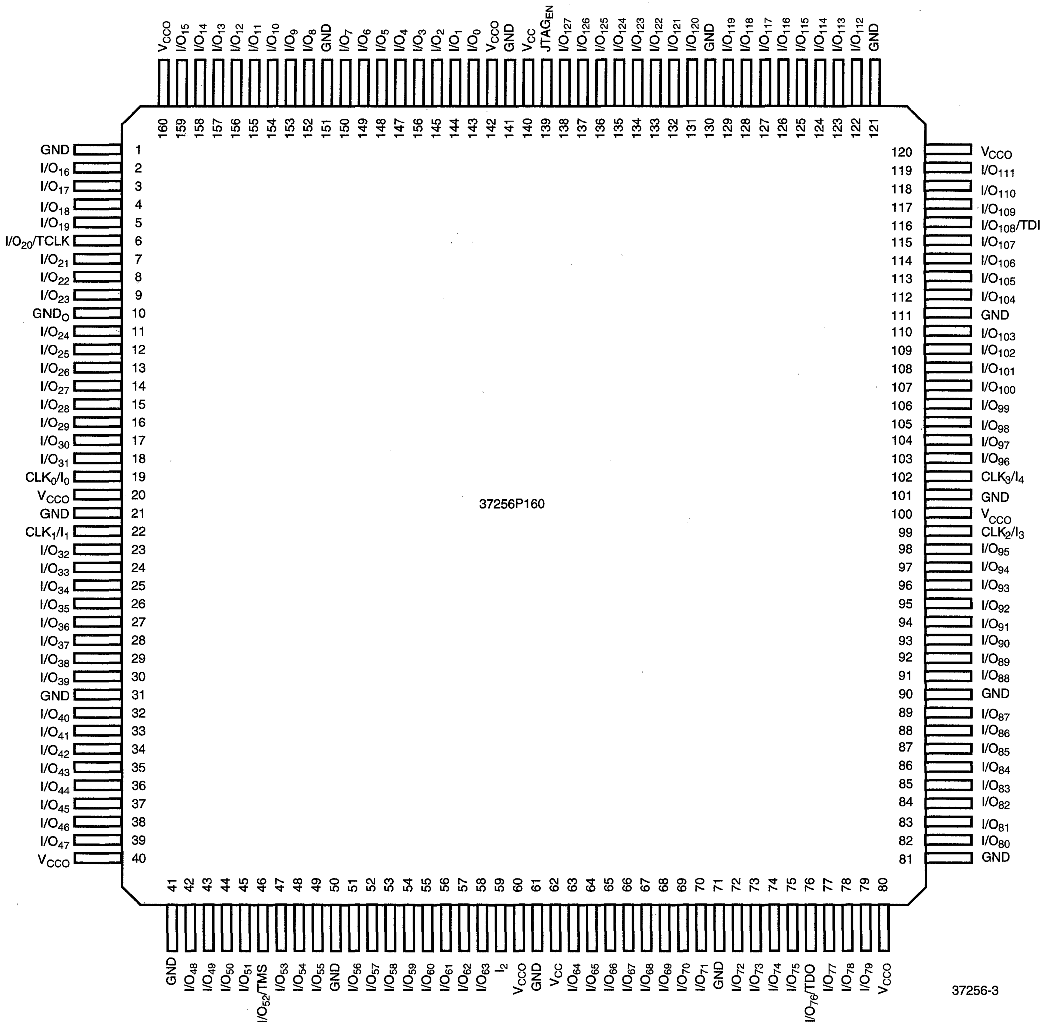
The Ultra37256 can be programmed in system using IEEE 1149.1 complaint JTAG programming protocol. The Ultra37256 can also be programmed on a number of traditional parallel programmers including Cypress's *Impulse3*™ programmer and industry standard third-party programmers. For an overview of ISR programming, refer to the Ultra37000 Family data sheet and for ISR cable and software specifications, refer to InSRkit: ISR programming data sheet (CY3600).

### Design Tools

Development software for the Ultra37256 is available from Cypress's *Warp*™ or third-party bolt-in software packages as well as a number of third-party development packages. Please refer to the *Warp* or third-party tool support data sheets for further information.

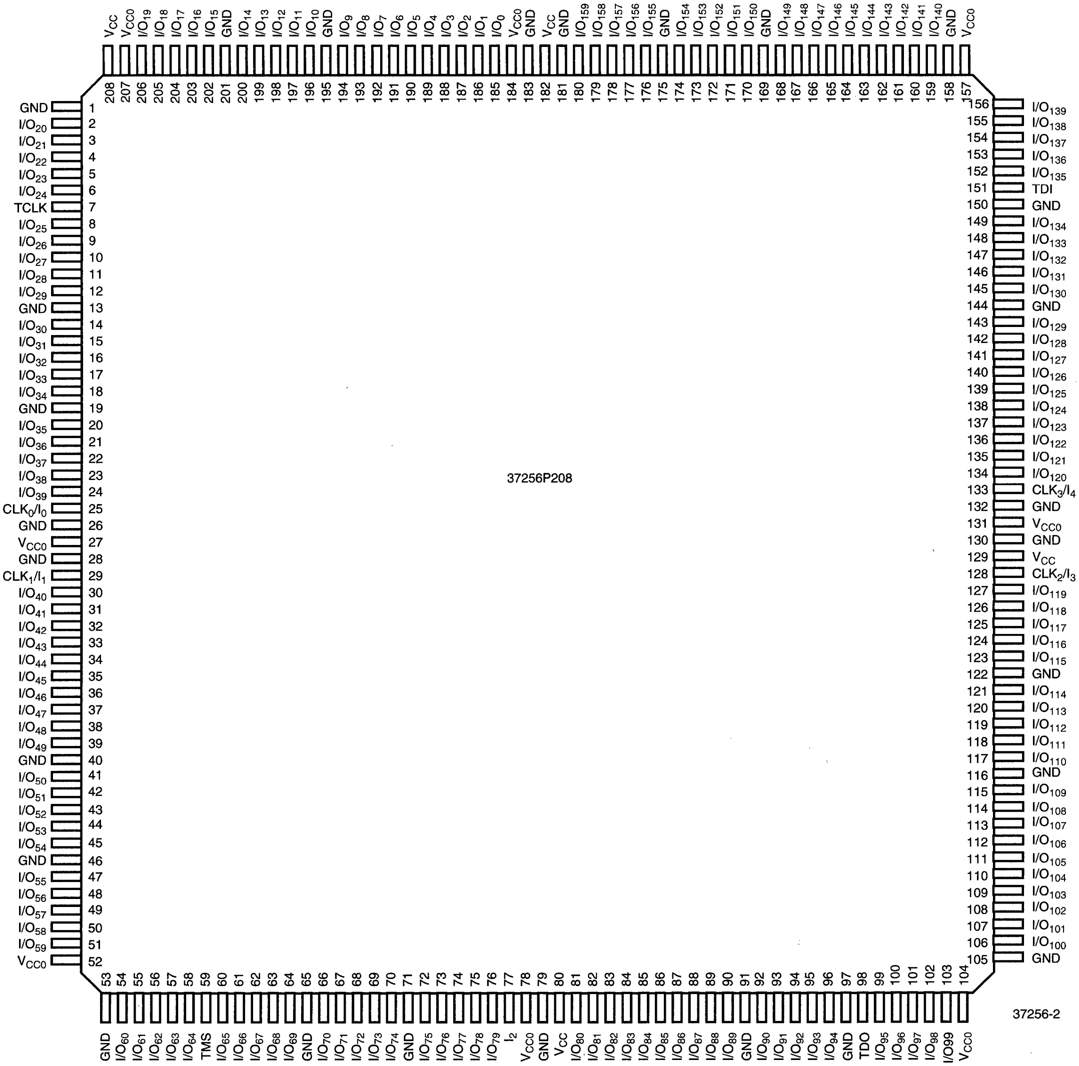
## Pin Configurations

**160-pin TQFP**
**Top View**



## Pin Configurations (continued)

208-pin PQFP
Top View



37256-2

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | –65°C to +150°C |
| Ambient Temperature with Power Applied | –55°C to +125°C |
| Supply Voltage to Ground Potential | –0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | –0.5V to +7.0V |
| DC Input Voltage | –0.5V to +7.0V |
| DC Program Voltage | 5V±5% |
| Output Current into Outputs | 16 mA |
| Static Discharge Voltage (per MIL-STD-883, Method 3015) | >2001V |
| Latch-Up Current | >200 mA |

## Operating Range[2]

| Range | Ambient Temperature[2] | Output Condition | $V_{CC}$ | $V_{CCO}$ |
|---|---|---|---|---|
| Commercial | 0°C to +70°C | 5V | 5V ± 5% | 5V ± 5% |
| | | 3.3V | 5V ± 5% | 3.3V ± 0.3V |
| Industrial | –40°C to +85°C | 5V | 5V ± 10% | 5V ± 10% |
| | | 3.3V | 5V ± 10% | 3.3V ± 0.3V |
| Military[3] | –55°C to +125°C | 5V | 5V ± 10% | 5V ± 10% |
| | | 3.3V | 5V ± 10% | 3.3V ± 0.3V |

Shaded areas contain advance information.

## Electrical Characteristics Over the Operating Range

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ =–3.2 mA[4] | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA[4] | | 0.5 | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH voltage for all inputs[5] | | 2.0 | $V_{CCmax}$ | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW voltage for all inputs[5] | | –0.5 | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I$ = $V_{CC}$ | | –10 | 10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = GND or $V_{CC}$, Output Disabled | | –50 | 50 | µA |
| $I_{OS}$ | Output Short Circuit Current[6, 7] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | –30 | –160 | mA |
| $I_{CC\text{-}HS}$ | Power Supply Current[8] Per Logic Block - High Speed Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA,f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 18.7 | mA |
| $I_{CC\text{-}LP}$ | Power Supply Current[8] Per Logic Block - Low Power Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA,f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 9.3 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | µA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | –75 | | µA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | +500 | µA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | –500 | µA |

**Notes:**

[2] Normal Programming Conditions apply across Ambient Temperature Range for specified programming methods. For more information on programming the Ultra37000 family devices see the Ultra37000 family data sheet.

[3] $T_A$ is the "instant on" case temperature.

[4] $I_{OH}$ = –2 mA, $I_{OL}$ = 2 mA for SDO.

[5] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[6] Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[7] Tested initially and after any design or process changes that may affect these parameters.

[8] Measured with 16-bit counter programmed into the logic block. Total device power calculated by summing the $I_{CC}$ specifications for the mode of operation of each logic block.

## Inductance[7]

| Parameter | Description | Test Conditions | 160-lead TQFP | 208-lead PQFP | Unit |
|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f = 1 MHz | 9 | 11 | nH |

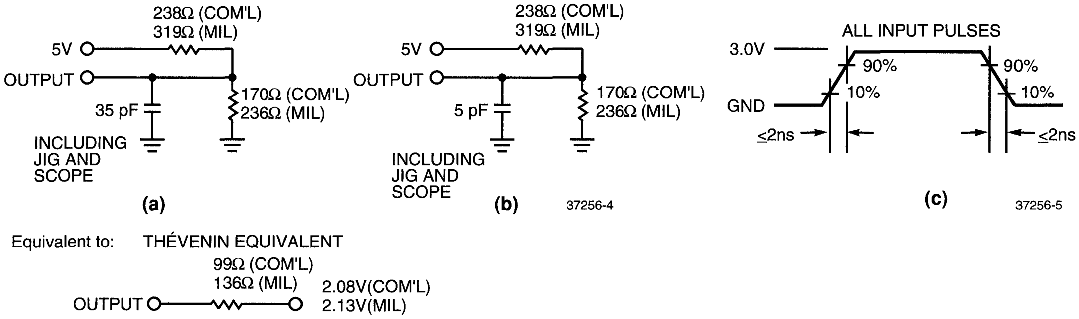
## Capacitance[7]

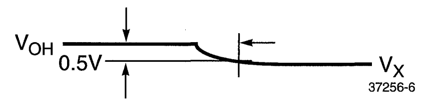
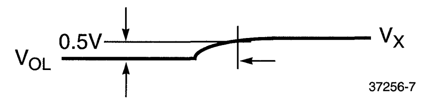
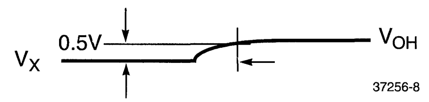
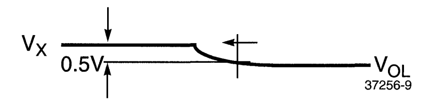
| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{I/O}$ | Input/Output Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 12 | pF |

## Endurance Characteristics[7]

| Parameter | Description | Test Conditions | Min. | Typ. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions[2] | 1,000 | 10,000 | Cycles |

## AC Test Loads and Waveforms



| Parameter[9] | $V_X$ | Output Waveform–Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | 1.5V |  |
| $t_{EA(-)}$ | $V_{the}$ |  |

**(d) Test Waveforms**

**Note:**

9. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[10]

| Parameter | Description | 37256-154 | | 37256-125 | | 37256-83 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | | | |
| $t_{PD}$[11] | Input to Combinatorial Output | | 7.5 | | 10 | | 15 | ns |
| $t_{PDL}$[11] | Input to Output Through Transparent Input or Output Latch | | 10 | | 13 | | 18 | ns |
| $t_{PDLL}$[11] | Input to Output Through Transparent Input and Output Latches | | 12 | | 15 | | 19 | ns |
| $t_{EA}$[12] | Input to Output Enable | | 11 | | 14 | | 19 | ns |
| $t_{ER}$ | Input to Output Disable | | 11 | | 14 | | 19 | ns |
| **Input Register Parameters** | | | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[7] | 2.5 | | 3 | | 4 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[7] | 2.5 | | 3 | | 4 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 2 | | 3 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 2 | | 3 | | ns |
| $t_{ICO}$[11] | Input Register Clock or Latch Enable to Combinatorial Output | | 11 | | 14 | | 19 | ns |
| $t_{ICOL}$[11] | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 12 | | 16 | | 21 | ns |
| **Synchronous Clocking Parameters** | | | | | | | | |
| $t_{CO}$[12] | Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output | | 5.0 | | 6.5 | | 8 | ns |
| $t_S$[11] | Set-Up Time from Input to Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 4.5 | | 5.5 | | 8 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | 0 | | ns |
| $t_{CO2}$[11, 12] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Combinatorial Output Delay (Through Logic Array) | | 11 | | 14 | | 19 | ns |
| $t_{SCS}$[11] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable (Through Logic Array) | 6.5 | | 8 | | 12 | | ns |
| $t_{SL}$[11] | Set-Up Time from Input Through Transparent Latch to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 8.0 | | 10 | | 15 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 0 | | 0 | | 0 | | ns |
| **Product Term Clocking Parameters** | | | | | | | | |
| $t_{COPT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output | | 10 | | 12 | | 17 | ns |
| $t_{SPT}$[11] | Set-Up Time from Input to Product Term Clock or Latch Enable (PTCLK) | 2.5 | | 3 | | 3 | | ns |
| $t_{HPT}$ | Register or Latch Data Hold Time | 2.5 | | 3 | | 3 | | ns |
| $t_{ISPT}$[11] | Set-Up Time for buried register used as an input register from Input to Product Term Clock or Latch Enable (PTCLK) | | –2 | | –2 | | –2 | ns |
| $t_{IHPT}$ | Buried Register used as an input register or Latch Data Hold Time | | 6.5 | | 9 | | 14 | ns |

**Notes:**

[10] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

[11] Logic Blocks operating in low power mode, add $t_{LP}$ to this spec.

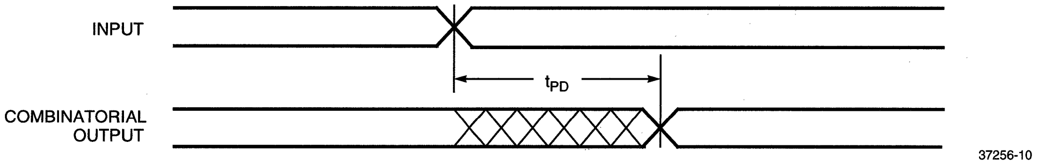
[12] Outputs using Slow Output Slew Rate, add $t_{SLEW}$ to this spec.

## Switching Characteristics Over the Operating Range[10] (continued)

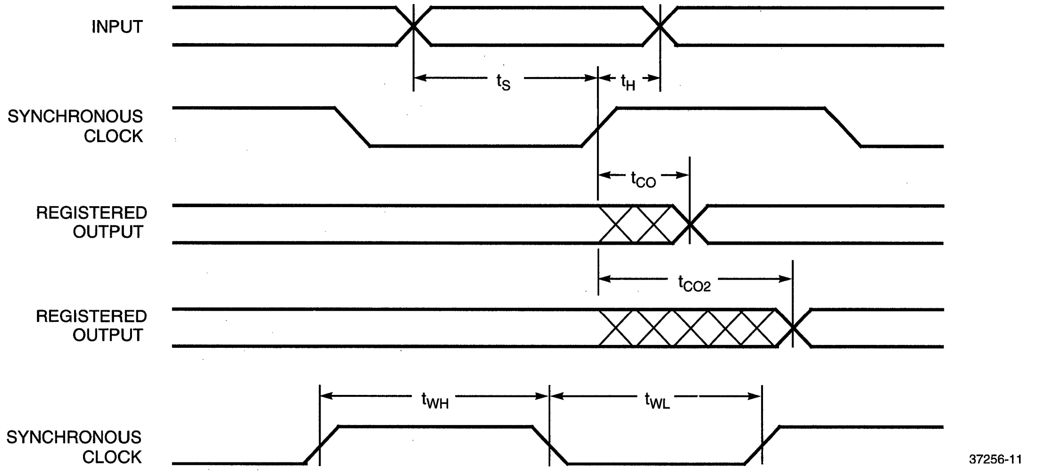
| Parameter | Description | 37256-154 |  | 37256-125 |  | 37256-83 |  | Unit |
|---|---|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. | Min. | Max. |  |
| $t_{CO2PT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output Delay (Through Logic Array) |  | 14 |  | 16 |  | 21 | ns |
| **Pipelined Mode Parameters** |  |  |  |  |  |  |  |  |
| $t_{ICS}$[11, 12] | Input Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) | 6 |  | 8 |  | 12 |  | ns |
| **Operating Frequency Parameters** |  |  |  |  |  |  |  |  |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[7] | 154 |  | 125 |  | 83 |  | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$) | 200 |  | 153.8 |  | 125 |  | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ or $1/(t_{WL} + t_{WH})$) | 118 |  | 83.3 |  | 62.5 |  | MHz |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 154 |  | 125 |  | 66.6 |  | MHz |
| **Reset/Preset Parameters** |  |  |  |  |  |  |  |  |
| $t_{RW}$ | Asynchronous Reset Width[7] | 8 |  | 10 |  | 15 |  | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[7] | 10 |  | 12 |  | 17 |  | ns |
| $t_{RO}$[11, 12] | Asynchronous Reset to Output |  | 14 |  | 16 |  | 21 | ns |
| $t_{PW}$ | Asynchronous Preset Width[7] | 8 |  | 10 |  | 15 |  | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[7] | 10 |  | 12 |  | 17 |  | ns |
| $t_{PO}$[11, 12] | Asynchronous Preset to Output |  | 14 |  | 16 |  | 21 | ns |
| **User Option Parameters** |  |  |  |  |  |  |  |  |
| $t_{LP}$ | Low Power Adder | 5 |  | 6 |  | 6 |  | ns |
| $t_{SLEW}$ | Slow Output Slew Rate Adder | 2 |  | 2 |  | 2 |  | ns |
| **Tap Controller Parameter** |  |  |  |  |  |  |  |  |
| $f_{TAP}$ | Tap Controller Frequency |  | 20 |  | 20 |  | 20 | MHz |

## Switching Waveforms

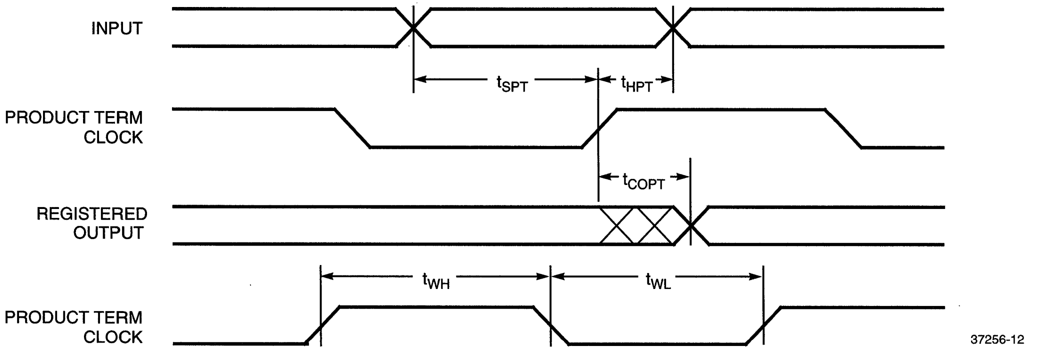
### Combinatorial Output



### Registered Output with Synchronous Clocking

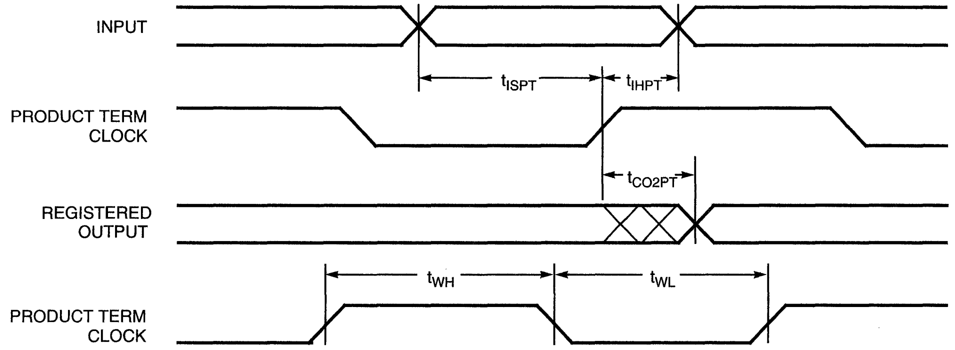


### Registered Output with Product Term Clocking Input going through the Array



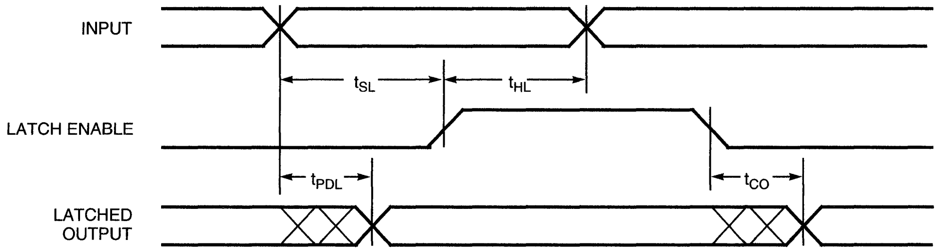
## Switching Waveforms (continued)

### Registered Output with Product Term Clocking Input coming from Adjacent Buried Register



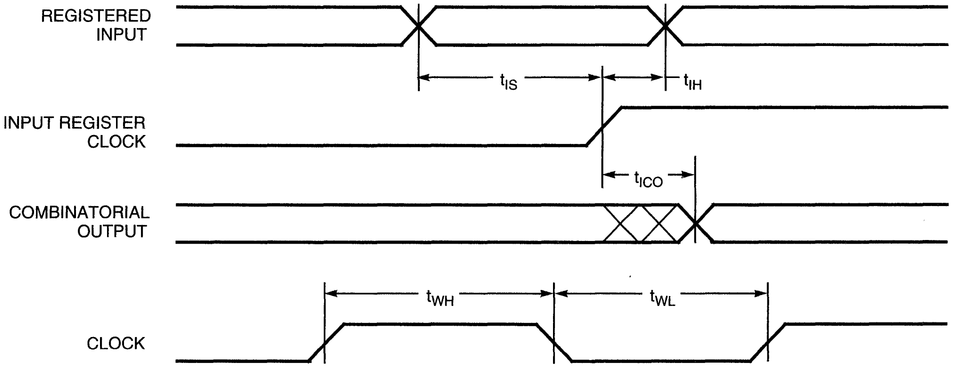
37256-13

### Latched Output



37256-14

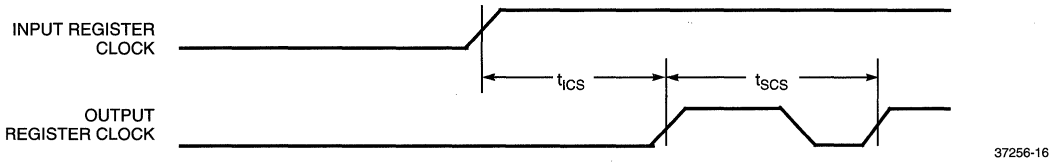
### Registered Input



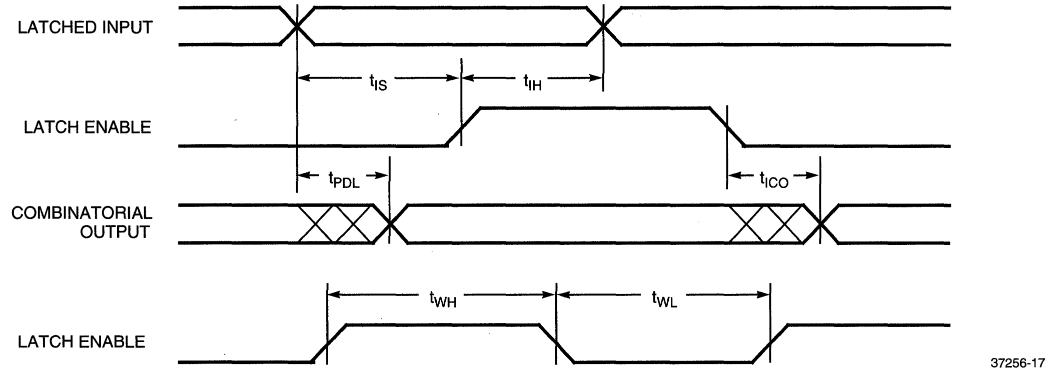
37256-15

## Switching Waveforms (continued)

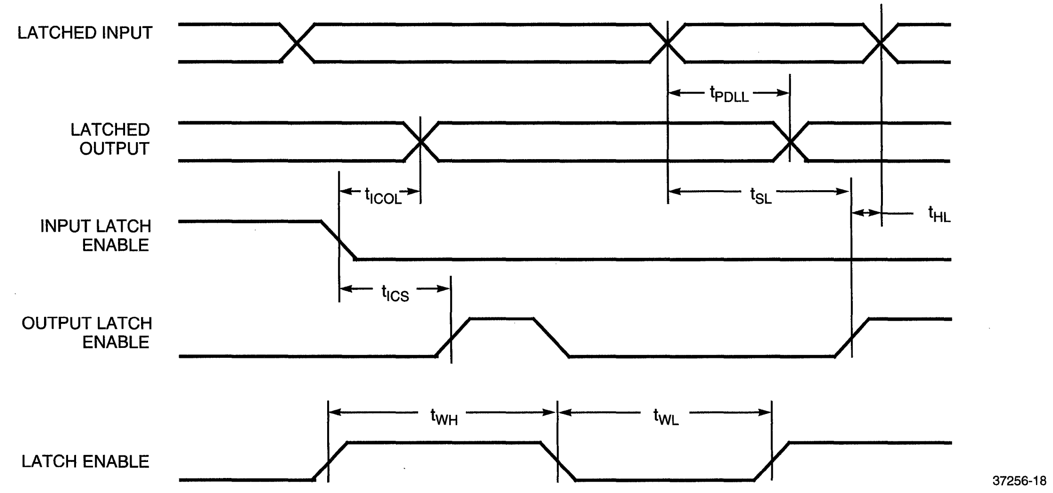
### Clock to Clock



### Latched Input

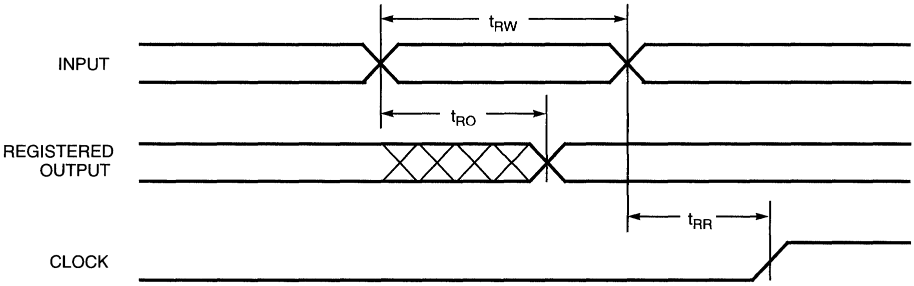


### Latched Input and Output



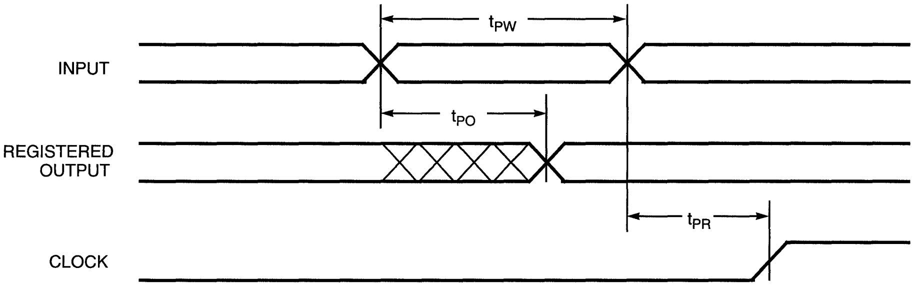
## Switching Waveforms (continued)

### Asynchronous Reset



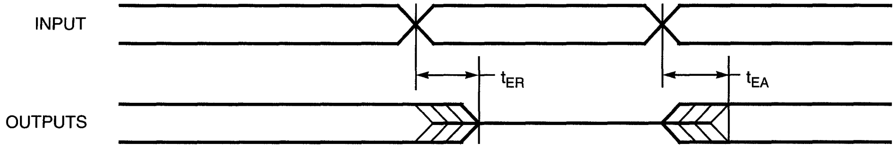
37256-19

### Asynchronous Preset



37256-20

### Output Enable/Disable



37256–21

## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 154 | CY37256P160-154AC | A160 | 160-Pin Thin Quad Flatpack | Commercial |
| | CY37256P208-154NC | N208 | 208-Pin Plastic Quad Flatpack | |
| | CY37265P256-154BGC | BG256 | 256-Lead Ball Grid Array | |
| 125 | CY37256P160-125AC | A160 | 160-Pin Thin Quad Flatpack | Commercial |
| | CY37256P208-125NC | N208 | 208-Pin Plastic Quad Flatpack | |
| | CY37265P256-125BGC | BG256 | 256-Lead Ball Grid Array | |
| | CY37256P160-125AI | A160 | 160-Pin Thin Quad Flatpack | Indusrtial |
| | CY37256P208-125NI | N208 | 208-Pin Plastic Quad Flatpack | |
| 83 | CY37256P160-83AC | A160 | 160-Pin Thin Quad Flatpack | Commercial |
| | CY37256P208-83NC | N208 | 208-Pin Plastic Quad Flatpack | |
| | CY37265P256-83BGC | BG256 | 256-Lead Ball Grid Array | |
| | CY37256P160-83AI | A160 | 160-Pin Thin Quad Flatpack | Indusrtial |
| | CY37256P208-83NI | N208 | 208-Pin Plastic Quad Flatpack | |

Shaded areas contain advance Information.

ISR, UltraLogic, FLASH370, Ultra37000, InSRkit, and *Warp2*Sim are trademarks of Cypress Semiconductor Corporation. *Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

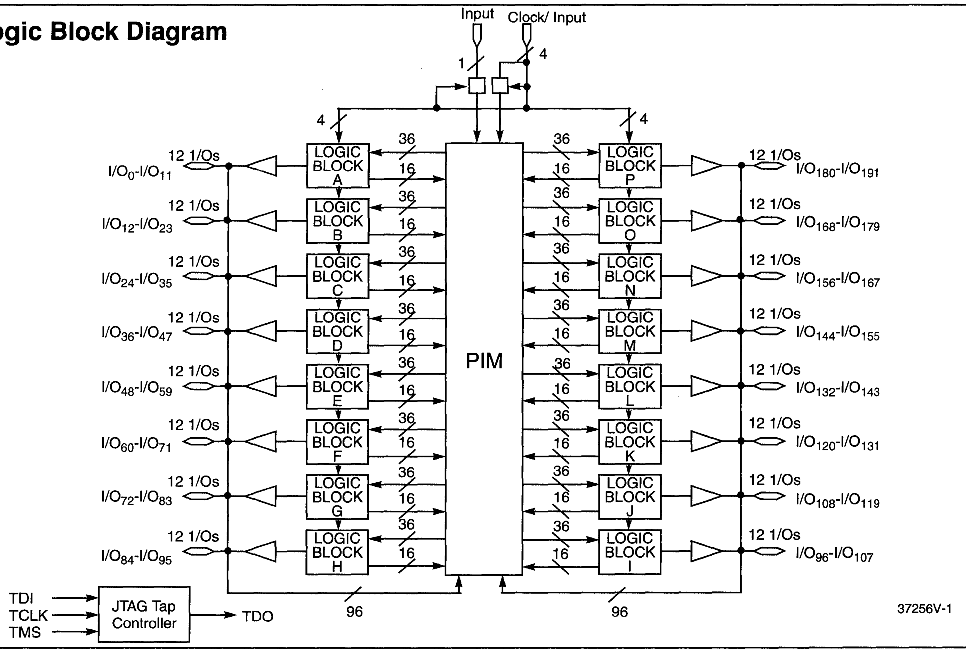
Document #: 38-00474-A

# UltraLogic™ 256-Macrocell 3.3V ISR™ CPLD

## Features

- 256 macrocells in sixteen logic blocks
- IEEE standard 3.3V operation
  - —3.3V ISR
  - —5V tolerant
- 3.3V In-System Reprogrammable (ISR™)
  - —JTAG compliant on board programming
  - —Design changes don't cause pinout changes
  - —Design changes don't cause timing changes
- High speed
  - —$f_{MAX}$ = 125 MHz
  - —$t_{PD}$ = 10 ns
  - —$t_S$ = 5.5 ns
  - —$t_{CO}$ = 6.5 ns
- Up to 192 I/Os
  - —plus 5 dedicated inputs including 4 clock inputs
- Product-term clocking
- IEEE1149.1 JTAG boundary scan
- Programmable slew rate control on individual I/Os
- Low power option on individual logic block basis
- Bus Hold capabilities on all I/Os
- Simple Timing Model
- Fully PCI compliant[1]
- Available in 160-pin TQFP, 208-pin PQFP and 256-lead BGA packages
- Pinout compatible with all other Ultra37000 family CPLDs

## Logic Block Diagram



## Selection Guide

| | Ultra37256V-125 | Ultra37256V-83 |
|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | 10 | 15 |
| Minimum Set-Up, $t_S$ (ns) | 5.5 | 8 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 6.5 | 8 |
| Typical Supply Current, $I_{CC}$ (mA) in Low Power Mode | 120 | 120 |

**Note:**

[1] Due to the 5V tolerant nature of the I/Os, the I/Os are not clamped to Vcc.

## Functional Description

The Ultra37256V is an In-System Reprogrammable (ISR) Complex Programmable Logic Device (CPLD) and is part of the Ultra37000™ family of high-density, high-speed CPLDs. Like all members of the Ultra37000 family, the Ultra37256V is designed to bring the ease of use and high performance of the 22V10 to high-density PLDs.

The 256 macrocell Ultra37256V is available in register intensive and I/O intensive versions. The Ultra37256VP160 features 96 Buried and 160 I/O Macrocells while the Ultra37256VP208 features 64 Buried Macrocells and 192 I/O Macrocells, for register intensive designs which require small footprint devices. The Ultra37256VP256 I/O intensive device has a I/O pin for each macrocell.

For a more detailed description of the architecture and features of the Ultra37256V see the Ultra37000 family data sheet.

### Fully Routable with 100% Logic Utilization

The Ultra37256V is designed with a robust routing architecture which allows utilization of the entire device with a fixed pinout. This makes Ultra37000 optimal for implementing on board design changes using ISR without changing pinouts.

### Simple Timing Model

The Ultra37256V features a very simple timing model with predictable delays. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. The timing model allows for design changes with ISR without causing changes to system performance.

### Low Power Operation

Each Logic Block of the Ultra37256V can be configured as either High-Speed (default) or Low-Power. In the Low-Power mode, the logic block consumes 50% less power (9.3 mA max.) and slows down by 6 ns.

### Output Slew Rate Control

Each output can be configured with either a fast edge rate (default) for high performance, or a slow edge rate for added noise reduction. In the fast edge rate mode, outputs switch at 3V/ns max. and in the slow edge rate mode, Outputs switch at 1V/ns max. There is a 2-ns adder for I/Os using the slow edge rate mode.

### In System Reprogramming

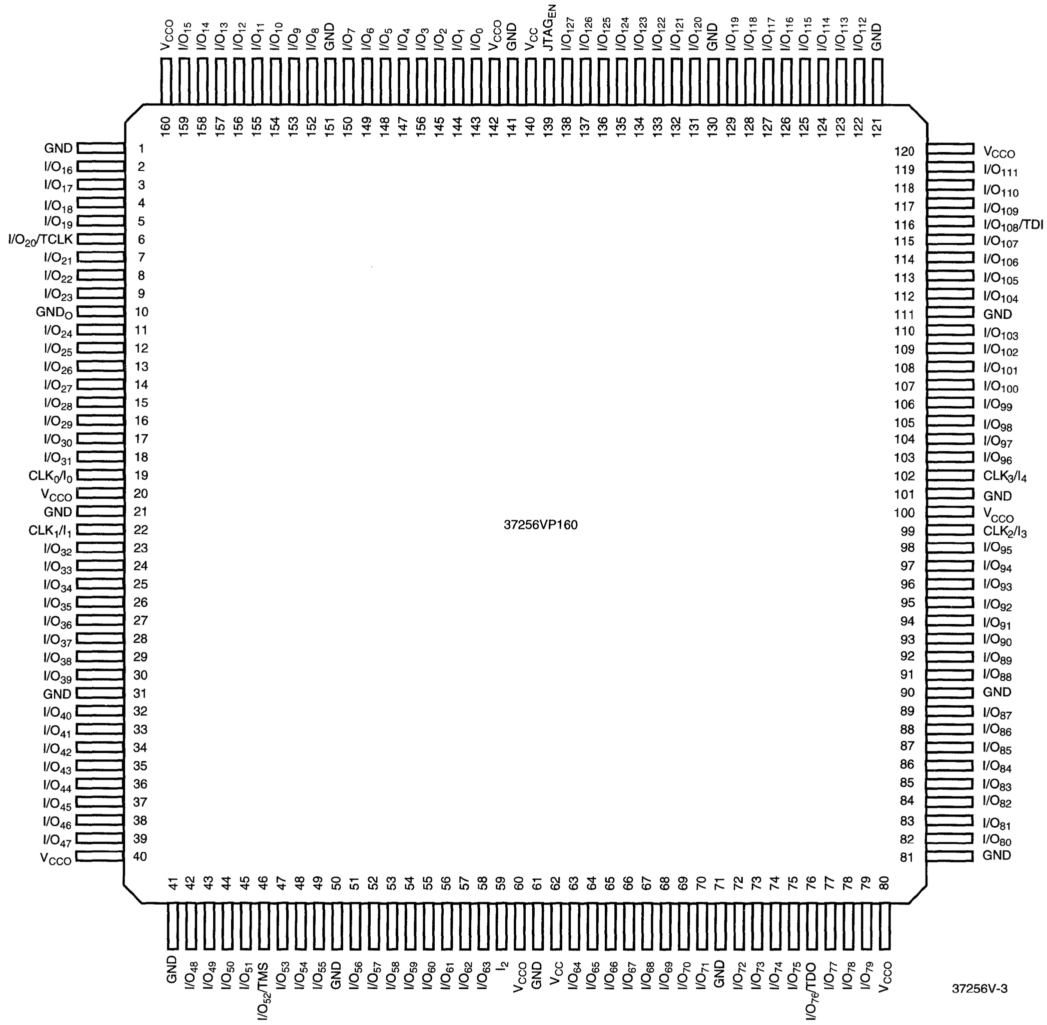
The Ultra37256V can be programmed in system using IEEE 1149.1 complaint JTAG programming protocol. The Ultra37256V can also be programmed on a number of traditional parallel programmers including Cypress's *Impulse3*™ programmer and industry standard third-party programmers. For an overview of ISR programming, refer to the Ultra37000 Family data sheet and for ISR cable and software specifications, refer to InSRkit: ISR programming data sheet (CY3600).

### Design Tools

Development software for the Ultra37256V is available from Cypress's *Warp*™ or third-party bolt-in software packages as well as a number of third-party development packages. Please refer to the *Warp* or third-party tool support data sheets for further information.

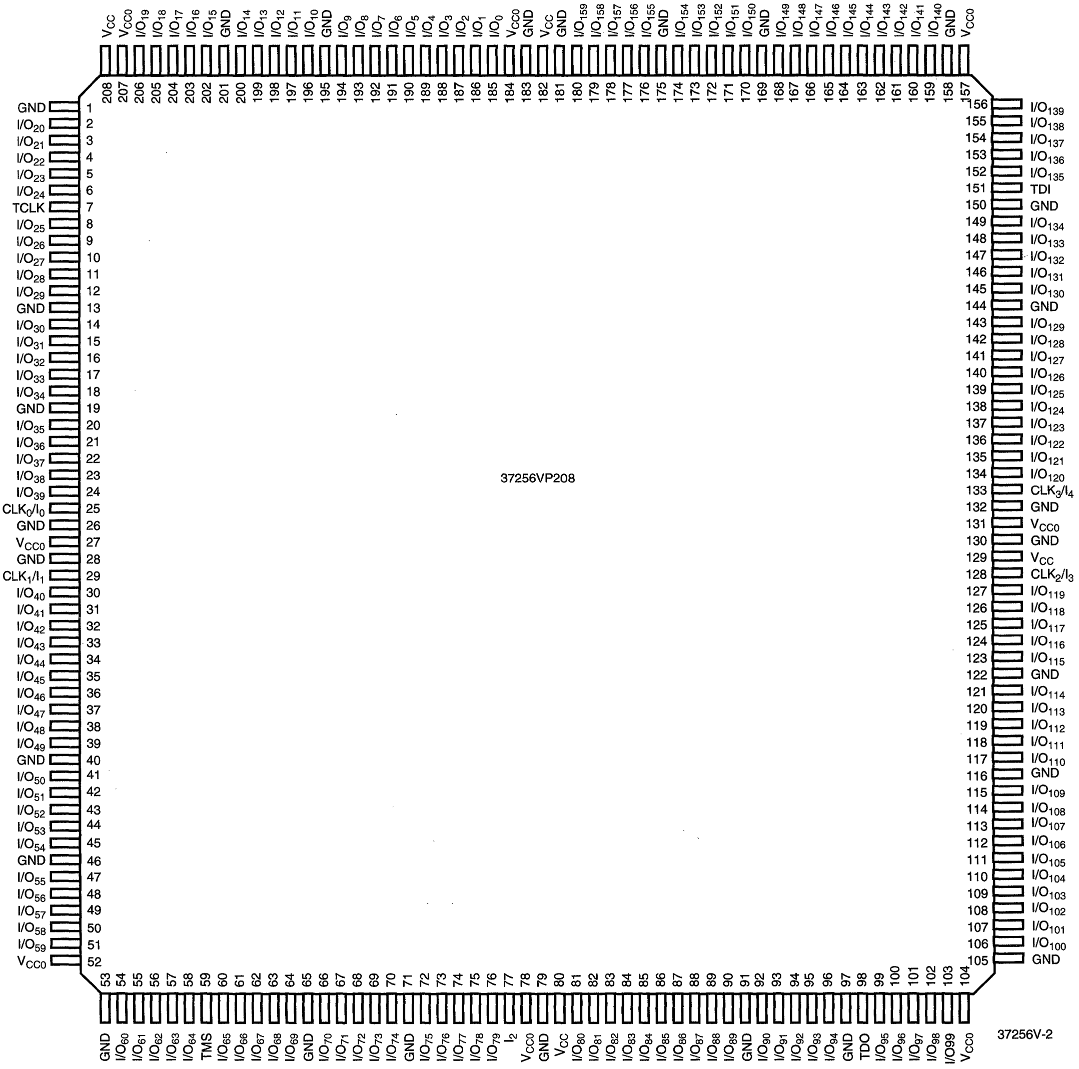
## Pin Configurations

**160-pin TQFP**
**Top View**



## Pin Configurations (continued)

**208-pin PQFP**
**Top View**



## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................–65°C to +150°C

Ambient Temperature with
Power Applied ...........................................–55°C to +125°C

Supply Voltage to Ground Potential ...............–0.5V to +7.0V

DC Voltage Applied to Outputs
in High Z State ..............................................–0.5V to +7.0V

DC Input Voltage...........................................–0.5V to +7.0V

DC Program Voltage ............................................ 3.3V±0.3V

Output Current into Outputs .........................................8 mA

Static Discharge Voltage...........................................>2001V
(per MIL-STD-883, Method 3015)

Latch-Up Current ...................................................>200 mA

## Operating Range[2]

| Range | Ambient Temperature[2] | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 3.3V ± 0.3V |
| Industrial | –40°C to +85°C | 3.3V ± 0.3V |
| Military[3] | –55°C to +125°C | 3.3V ± 0.3V |

Shaded areas contain advance information.

3

## Electrical Characteristics Over the Operating Range

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = –4 mA (Com'l)[4] | 2.4 | | V |
| | | | $I_{OH}$ = –3 mA (Mil)[4] | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 8 mA (Com'l)[4] | | 0.5 | V |
| | | | $I_{OL}$ = 6 mA (Mil)[4] | | | |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH voltage for all inputs[5] | | 2.0 | $V_{CCmax}$ | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW voltage for all inputs[5] | | –0.5 | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I$ = $V_{CC}$ | | –10 | 10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = GND or $V_{CC}$, Output Disabled | | –50 | 50 | µA |
| $I_{OS}$ | Output Short Circuit Current[6, 7] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | –30 | –160 | mA |
| $I_{CC\text{-}HS}$ | Power Supply Current[8] Per Logic Block - High Speed Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA,f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 18.7 | mA |
| $I_{CC\text{-}LP}$ | Power Supply Current[8] Per Logic Block - Low Power Mode | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA,f = 1 MHz, $V_{IN}$ = GND or $V_{CC}$ | | | 9.3 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | µA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | –75 | | µA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | +500 | µA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | –500 | µA |

## Inductance[7]

| Parameter | Description | Test Conditions | 160-lead TQFP | 208-lead PQFP | Unit |
|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 3.3V at f = 1 MHz | 9 | 11 | nH |

**Notes:**

[2] Normal Programming Conditions apply across Ambient Temperature Range for specified programming methods. For more information on programming the Ultra37000 family devices see the Ultra37000 family data sheet.

[3] $T_A$ is the "instant on" case temperature.

[4] $I_{OH}$ = –2 mA, $I_{OL}$ = 2 mA for SDO.

[5] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[6] Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[7] Tested initially and after any design or process changes that may affect these parameters.

[8] Measured with 16-bit counter programmed into the logic block. Total device power calculated by summing the $I_{CC}$ specifications for the mode of operation of each logic block.

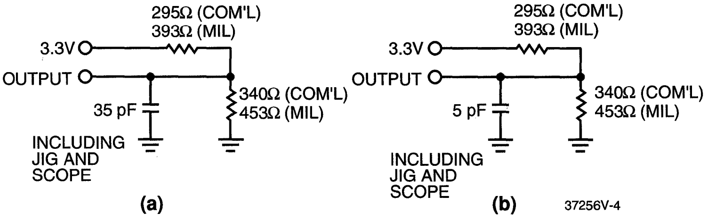
## Capacitance[7]

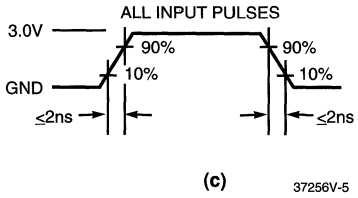
| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{I/O}$ | Input/Output Capacitance | $V_{IN}$ = 3.3V at f = 1 MHz | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 3.3V at f = 1 MHz | 12 | pF |

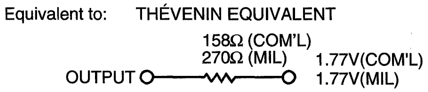
## Endurance Characteristics[7]

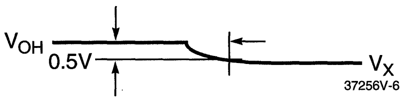
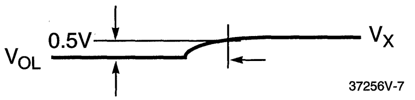
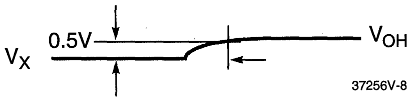
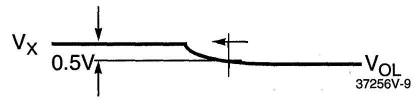
| Parameter | Description | Test Conditions | Min. | Typ. | Unit |
|---|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions[2] | 1,000 | 10,000 | Cycles |

## AC Test Loads and Waveforms







| Parameter[9] | $V_X$ | Output Waveform--Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | 1.5V |  |
| $t_{EA(-)}$ | $V_{the}$ |  |

(d) Test Waveforms

**Note:**

9. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[10]

| Parameter | Description | 37256V-125 | | 37256V-83 | | Unit |
|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | |
| $t_{PD}$[11] | Input to Combinatorial Output | | 10 | | 15 | ns |
| $t_{PDL}$[11] | Input to Output Through Transparent Input or Output Latch | | 13 | | 18 | ns |
| $t_{PDLL}$[11] | Input to Output Through Transparent Input and Output Latches | | 15 | | 19 | ns |
| $t_{EA}$[12] | Input to Output Enable | | 14 | | 19 | ns |
| $t_{ER}$ | Input to Output Disable | | 14 | | 19 | ns |
| **Input Register Parameters** | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[7] | 3 | | 4 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[7] | 3 | | 4 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 3 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 3 | | ns |
| $t_{ICO}$[11] | Input Register Clock or Latch Enable to Combinatorial Output | | 14 | | 19 | ns |
| $t_{ICOL}$[11] | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 16 | | 21 | ns |
| **Synchronous Clocking Parameters** | | | | | | |
| $t_{CO}$[12] | Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output | | 6.5 | | 8 | ns |
| $t_S$[11] | Set-Up Time from Input to Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 5.5 | | 8 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | ns |
| $t_{CO2}$[11, 12] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Combinatorial Output Delay (Through Logic Array) | | 14 | | 19 | ns |
| $t_{SCS}$[11] | Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable to Output Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable (Through Logic Array) | 8 | | 12 | | ns |
| $t_{SL}$[11] | Set-Up Time from Input Through Transparent Latch to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 10 | | 15 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) or Latch Enable | 0 | | 0 | | ns |
| **Product Term Clocking Parameters** | | | | | | |
| $t_{COPT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output | | 12 | | 17 | ns |
| $t_{SPT}$[11] | Set-Up Time from Input to Product Term Clock or Latch Enable (PTCLK) | 3 | | 3 | | ns |
| $t_{HPT}$ | Register or Latch Data Hold Time | 3 | | 3 | | ns |
| $t_{ISPT}$[11] | Set-Up Time for buried register used as an input register from Input to Product Term Clock or Latch Enable (PTCLK) | | –2 | | –2 | ns |
| $t_{IHPT}$ | Buried Register used as an input register or Latch Data Hold Time | | 9 | | 14 | ns |

**Notes:**

[10] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

[11] Logic Blocks operating in low power mode, add $t_{LP}$ to this spec.

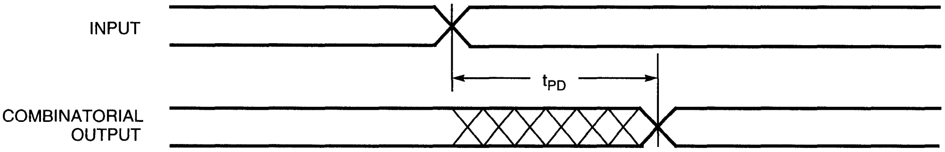
[12] Outputs using Slow Output Slew Rate, add $t_{SLEW}$ to this spec.

## Switching Characteristics Over the Operating Range[10] (continued)

| Parameter | Description | 37256V-125 |  | 37256V-83 |  | Unit |
|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. |  |
| $t_{CO2PT}$[11, 12] | Product Term Clock or Latch Enable (PTCLK) to Output Delay (Through Logic Array) |  | 16 |  | 21 | ns |
| **Pipelined Mode Parameters** |  |  |  |  |  |  |
| $t_{ICS}$[11, 12] | Input Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) to Output Register Synchronous Clock ($CLK_0$, $CLK_1$, $CLK_2$, or $CLK_3$) | 8 |  | 12 |  | ns |
| **Operating Frequency Parameters** |  |  |  |  |  |  |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[7] | 125 |  | 83 |  | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$) | 153.8 |  | 125 |  | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ or $1/(t_{WL} + t_{WH})$) | 83.3 |  | 62.5 |  | MHz |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 125 |  | 66.6 |  | MHz |
| **Reset/Preset Parameters** |  |  |  |  |  |  |
| $t_{RW}$ | Asynchronous Reset Width[7] | 10 |  | 15 |  | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[7] | 12 |  | 17 |  | ns |
| $t_{RO}$[11, 12] | Asynchronous Reset to Output |  | 16 |  | 21 | ns |
| $t_{PW}$ | Asynchronous Preset Width[7] | 10 |  | 15 |  | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[7] | 12 |  | 17 |  | ns |
| $t_{PO}$[11, 12] | Asynchronous Preset to Output |  | 16 |  | 21 | ns |
| **User Option Parameters** |  |  |  |  |  |  |
| $t_{LP}$ | Low Power Adder | 6 |  | 6 |  | ns |
| $t_{SLEW}$ | Slow Output Slew Rate Adder | 2 |  | 2 |  | ns |
| **Tap Controller Parameter** |  |  |  |  |  |  |
| $f_{TAP}$ | Tap Controller Frequency |  | 20 |  | 20 | MHz |

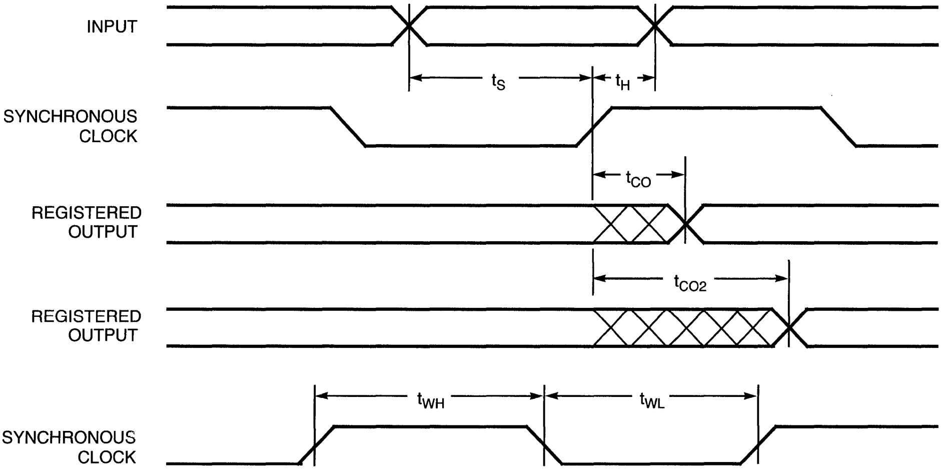
## Switching Waveforms

### Combinatorial Output



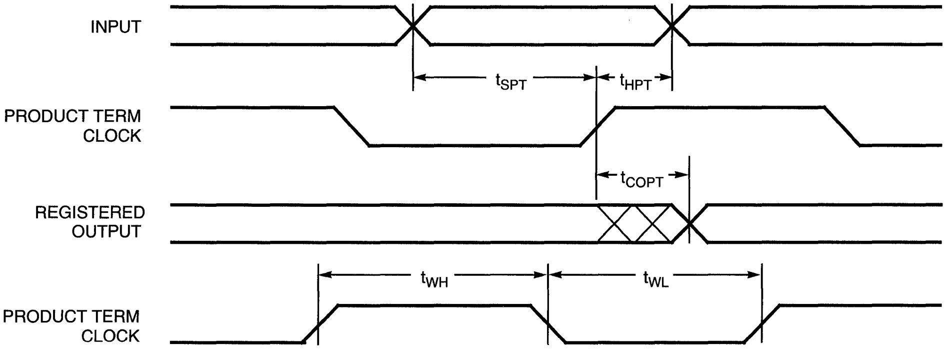
37256V-10

### Registered Output with Synchronous Clocking



37256V-11

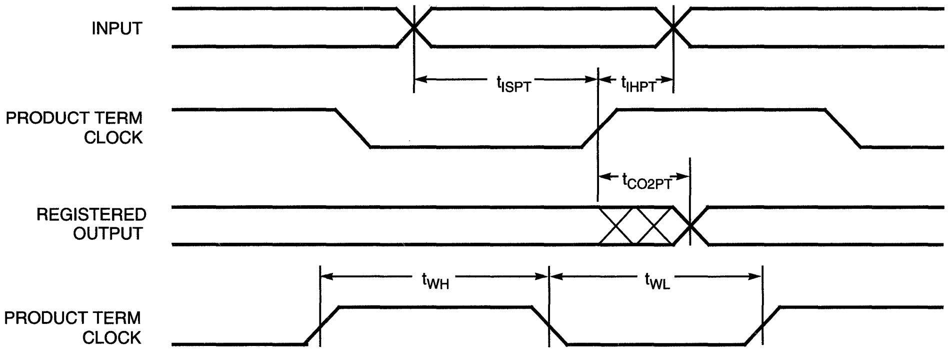
### Registered Output with Product Term Clocking Input going through the Array



37256V-12

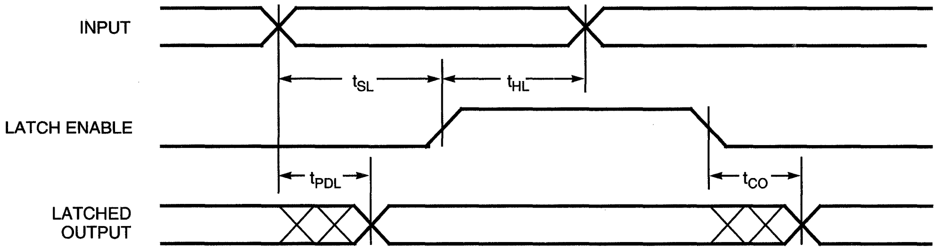
## Switching Waveforms (continued)

### Registered Output with Product Term Clocking Input coming from Adjacent Buried Register



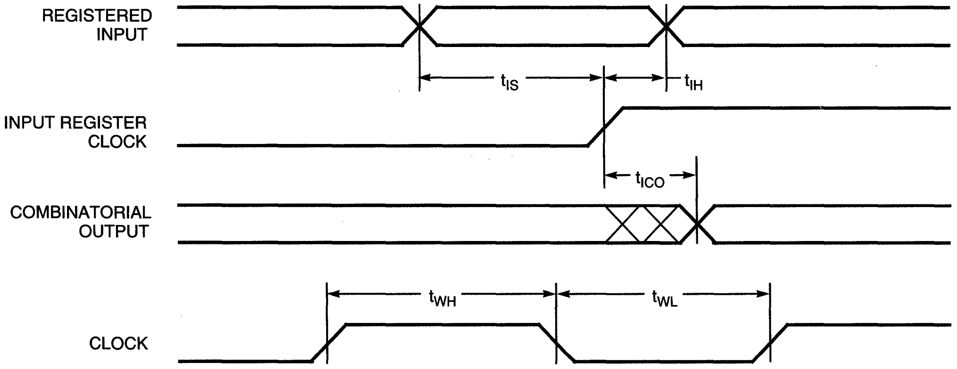
37256V-13

### Latched Output



37256V-14

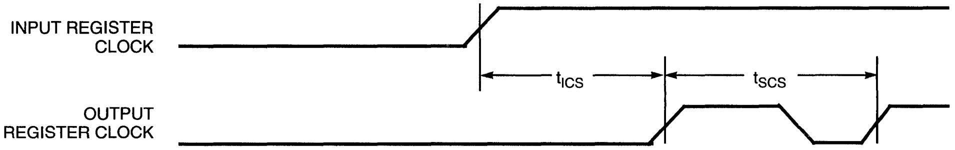
### Registered Input



37256V-15

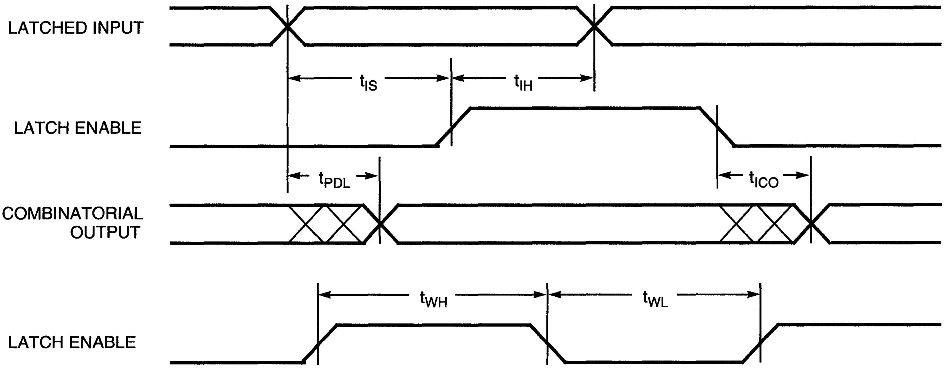
## Switching Waveforms (continued)

### Clock to Clock



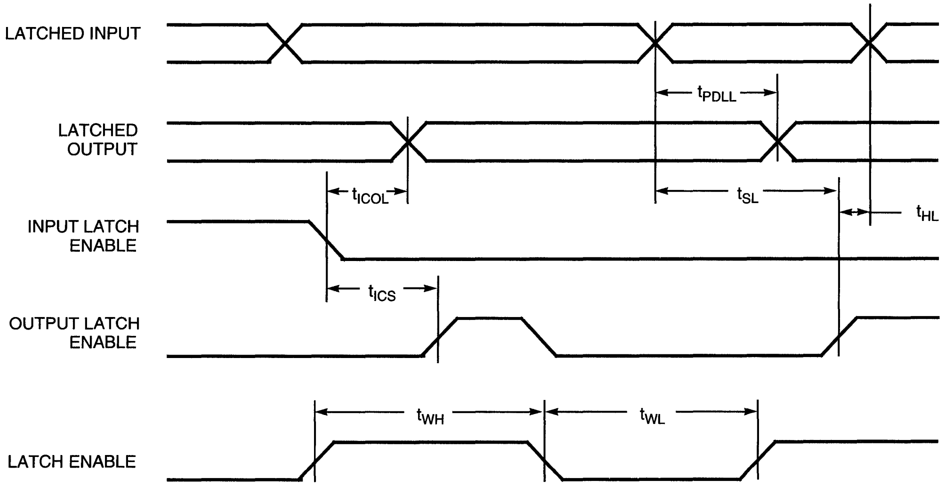
37256V-16

### Latched Input



37256V-17

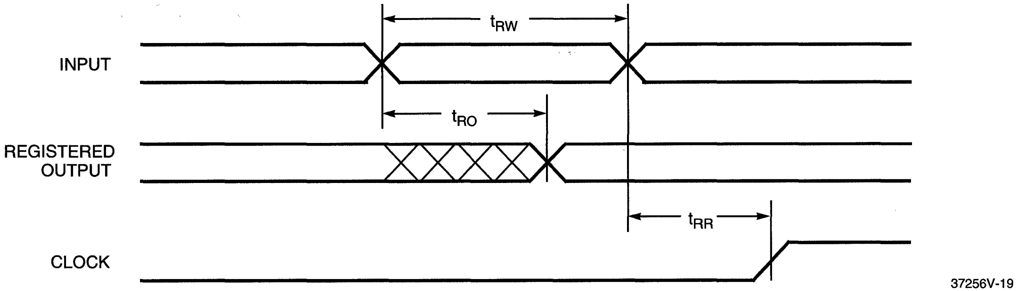
### Latched Input and Output



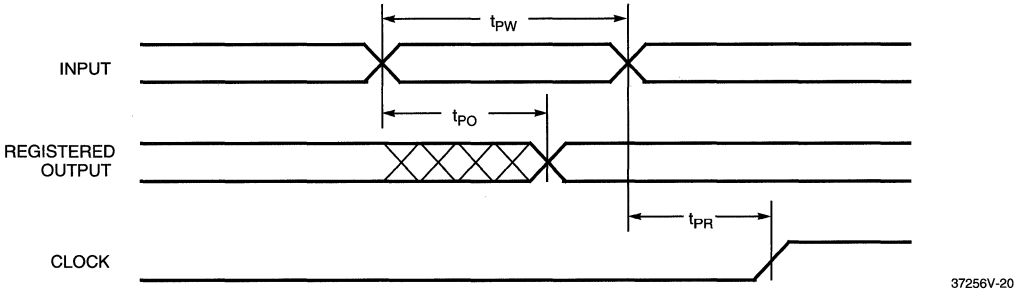
37256V-18

## Switching Waveforms (continued)

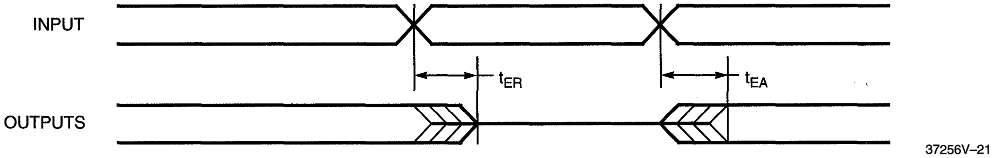
### Asynchronous Reset



### Asynchronous Preset



### Output Enable/Disable



## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 125 | CY37256VP160–125AC | A160 | 160-Pin Thin Quad Flatpack | Commercial |
| | CY37256VP208-125NC | N208 | 208-Pin Plastic Quad Flatpack | |
| | CY37256VP256-125BGC | BG256 | 256-Lead Ball Grid Array | |
| | CY37256VP160–125AI | A160 | 160-Pin Thin Quad Flatpack | Industrial |
| | CY37256VP208-125NI | N208 | 208-Pin Plastic Quad Flatpack | |
| 83 | CY37256VP160–83AC | A160 | 160-Pin Thin Quad Flatpack | Commercial |
| | CY37256VP208-83NC | N208 | 208-Pin Plastic Quad Flatpack | |
| | CY37256VP256-83BGC | BG256 | 256-Lead Ball Grid Array | |
| | CY37256VP160–83AI | A160 | 160-Pin Thin Quad Flatpack | Industrial |
| | CY37256VP208-83NI | N208 | 208-Pin Plastic Quad Flatpack | |

Shaded areas contain advance information.

ISR, UltraLogic, Ultra37000, *Impulse3*, InSRkit, and*Warp2*Sim are trademarks of Cypress Semiconductor Corporation. *Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

Document #: 38-00606

CYPRESS

# FLASH370i™ ISR™ CPLD Family

## UltraLogic™ High-Density Flash CPLDs

## Features

- Flash In-System Reprogrammable (ISR™) CMOS CPLDs
  - —Combines on board reprogramming with pinout flexibility and a simple timing model
  - —Design changes don't cause pinout or timing changes
  - —JTAG interface
- High density
  - —32–128 macrocells
  - —32–128 I/O pins
  - —Multiple clock pins
- Fully PCI compliant
- Bus Hold capabilities on all I/Os and dedicated inputs
- High speed
  - —$t_{PD}$ = 8.5–10 ns
  - —$t_S$ = 5–7 ns
  - —$t_{CO}$ = 6–7 ns
- Fast Programmable Interconnect Matrix (PIM)
  - —Uniform predictable delay, independent of routing
- Intelligent product term allocator
  - —0–16 product terms to any macrocell
  - —Provides product term steering on an individual basis
  - —Provides product term sharing among local macrocells
- Simple timing model
  - —No fanout delays
  - —No expander delays
  - —No dedicated vs. I/O pin delays
  - —No additional delay through PIM
  - —No penalty for using full 16 product terms
  - —No delay for steering or sharing product terms
- Flexible clocking
  - —2–4 clock pins per device
  - —Clock polarity control
- Security bit and user ID supported
- Packages
  - —44–160 pins
  - —PLCC, CLCC, PGA, CQFP, and TQFP packages
- *Warp2*®
  - —Low-cost IEEE 1164-compliant VHDL development system
  - —Available on PC, Sun, and HP platforms
  - —Supports all PLDs, CPLDs, FPGAs
- *Warp2*Sim™
  - —Includes capabilities of *Warp2* and ViewLogic's ViewSim
  - —Provides dynamic timing solutions for all Cypress PLDs, CPLDs, and FPGAs
- *Warp3*® CAE development system
  - —VHDL input
  - —ViewLogic graphical user interface
  - —Schematic capture (ViewDraw™)
  - —VHDL simulation (ViewSim™)
  - —Available on PC, Sun, and HP platforms

## General Description

The FLASH370i™ family of CMOS CPLDs provides a range of high-density programmable logic solutions with unparalleled performance. Each member of the family is designed with Cypress's state-of-the-art 0.65-micron Flash technology.

All of the UltraLogic FLASH370i devices are electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows, thereby reducing costs. Because of the superior routability of the FLASH370i devices, ISR allows users to change existing logic designs without changing pinout assignments or timing. The Cypress ISR function is implemented through a JTAG serial interface. Data is shifted in and out through the SDI and SDO pins, respectively. The ISR interface is enabled from the programming voltage pin ($ISR_{EN}$). The entire family is fully compliant with the PCI Local Bus specification, meeting all the electrical and timing requirements. Also, the entire family features bus-hold capabilities on all I/Os and dedicated inputs. Additionally, the entire family is security bit and user ID supported (when the security bit is programmed, all locations cannot be verified).

## FLASH370i Selection Guide

| Device | Pins | Macrocells | Dedicated Inputs | I/O Pins | Flip-Flops | Speed ($t_{PD}$) | Speed ($f_{MAX}$) |
|---|---|---|---|---|---|---|---|
| 371i | 44 | 32 | 5 | 32 | 44 | 8.5 | 143 |
| 372i | 44 | 64 | 5 | 32 | 76 | 10 | 125 |
| 373i | 84/100 | 64 | 5 | 64 | 76 | 10 | 125 |
| 374i | 84/100 | 128 | 5 | 64 | 140 | 10 | 125 |
| 375i | 160 | 128 | 5 | 128 | 140 | 10 | 125 |

## General Description (continued)

The FLASH370i family is designed to bring the flexibility, ease of use and performance of the 22V10 to high-density CPLDs. The architecture is based on a number of logic blocks that are connected by a Programmable Interconnect Matrix (PIM). Each logic block features its own product term array, product term allocator array, and 16 macrocells. The PIM distributes signals from the logic block outputs and all input pins to the logic block inputs.

The family features a wide variety of densities and pin counts to choose from. At each density there are two packaging options to choose from—one that is I/O intensive and another that is register intensive. For example, the CY7C374i and CY7C375i both feature 128 macrocells. On the CY7C374i, available in 84-pin packages, half of the macrocells are buried. On the CY7C375i, available in 160-pin packages, all of the macrocells are fed to I/O pins. *Figure 1* shows a block diagram of the CY7C374i/5i.

## Functional Description

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) consists of a completely global routing matrix for signals from I/O pins and feedbacks from the logic blocks. The PIM is an extremely robust interconnect that avoids fitting and density limitations. Routing is automatically accomplished by software and the propagation delay through the PIM is transparent to the user. Signals from any pin or any logic block can be routed to any or all logic blocks.

The inputs to the PIM consist of all I/O and dedicated input pins and all macrocell feedbacks from within the logic blocks. The number of PIM inputs increases with pincount and the number of logic blocks. The outputs from the PIM are signals routed to the appropriate logic block(s). Each logic block receives 36 inputs from the PIM and their complements, allowing for 32-bit operations to be implemented in a single pass through the device. The wide number of inputs to the logic block also improves the routing capacity of the FLASH370i family.

An important feature of the PIM is simple timing. The propagation delay through the PIM is accounted for in the timing specifications for each device. There is no additional delay for traveling through the PIM. In fact, all inputs travel through the PIM. Likewise, there are no route-dependent timing parameters on the FLASH370i devices. The worst-case PIM delays are incorporated in all appropriate FLASH370i specifications.

Routing signals through the PIM is completely invisible to the user. All routing is accomplished by software—no hand routing is necessary. *Warp* and third-party development packages automatically route designs for the FLASH370i family in a matter of minutes. Finally, the rich routing resources of the FLASH370i family accommodate last minute logic changes while maintaining fixed pin assignments.

## Logic Block Diagram

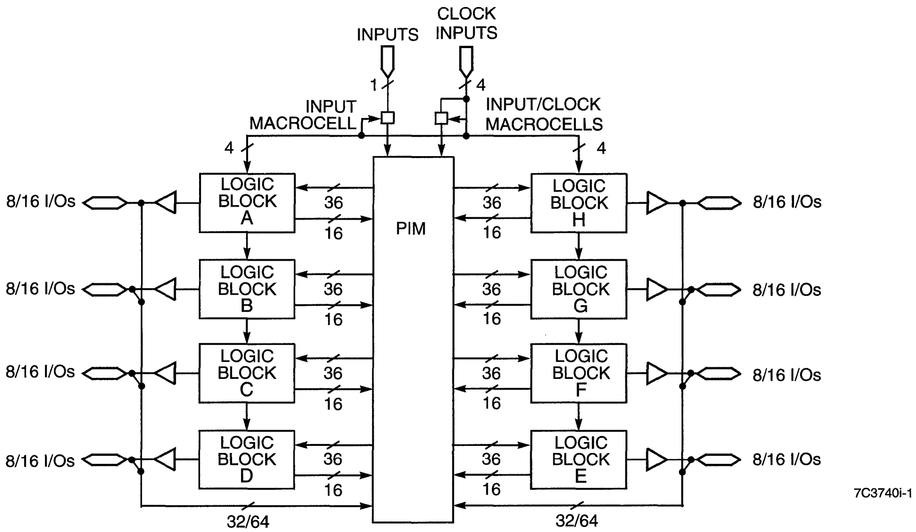
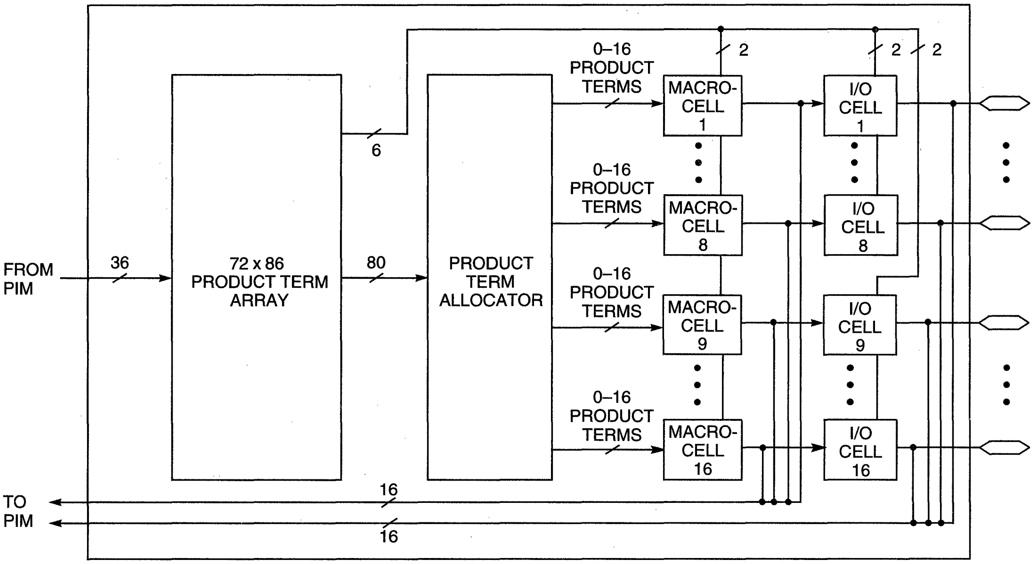
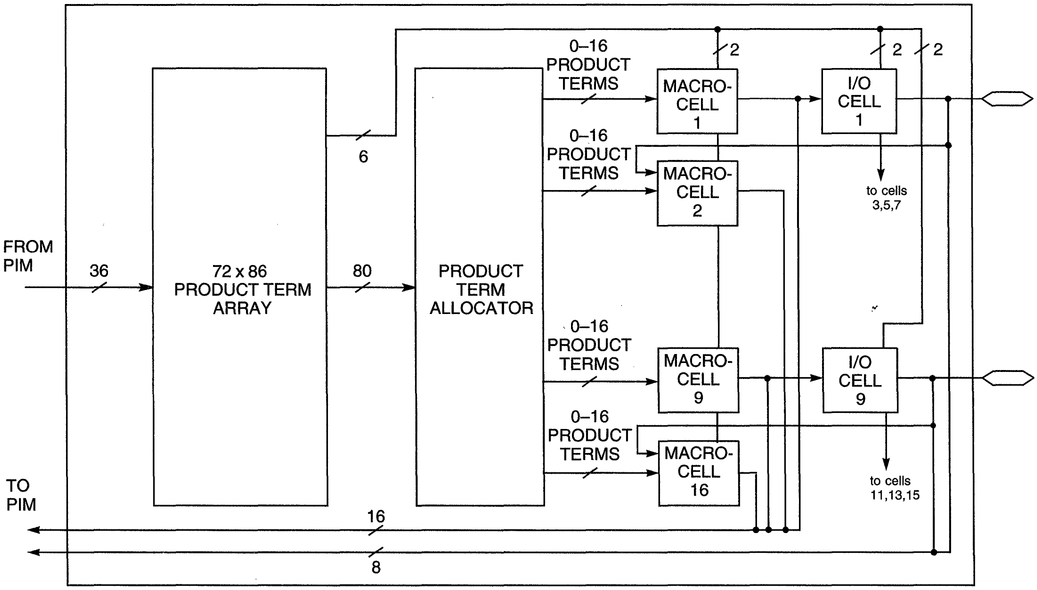


**Figure 1. CY7C374i/5i Block Diagram**



flash370i–2

**Figure 2. Logic Block for CY7C371i, CY7C373i, and CY7C375i (I/O Intensive)**



flash370i–3

**Figure 3. Logic Block for CY7C372i and CY7C374i (Register Intensive)**

### Logic Block

The logic block is the basic building block of the FLASH370i architecture. It consists of a product term array, an intelligent product-term allocator, 16 macrocells, and a number of I/O cells. The number of I/O cells varies depending on the device used.

There are two types of logic blocks in the FLASH370i family. The first type features an equal number (16) of I/O cells and macrocells and is shown in *Figure 2*. This architecture is best for I/O-intensive applications. The second type of logic block features a buried macrocell along with each I/O macrocell. In other words, in each logic block, there are eight macrocells that are connected to I/O cells and eight macrocells that are only internally fed back to the PIM. This organization is designed for register-intensive applications and is displayed in *Figure 3*. Note that at each FLASH370i density (except the 32-macrocell CY7C371i), an I/O intensive and a register-intensive device is available.

### Product Term Array

Each logic block features a 72 x 86 programmable product term array. This array is fed with 36 inputs from the PIM, which originate from macrocell feedbacks and device pins. Active LOW and active HIGH versions of each of these inputs are generated to create the full 72-input field. The 86 product terms in the array can be created from any of the 72 inputs.

Of the 86 product terms, 80 are for general-purpose use for the 16 macrocells in the logic block. Four of the remaining six product terms in the logic block are output enable (OE) product terms. Each of the OE product terms controls up to eight of the 16 macrocells and is selectable on an individual macrocell basis. In other words, each I/O cell can select between one of two OE product terms to control the output buffer. The first two of these four OE product terms are available to the upper half of the I/O macrocells in a logic block. The other two OE product terms are available to the lower half of the I/O macrocells in a logic block. The final two product terms in each logic block are dedicated asynchronous set and asynchronous reset product terms.

### Product Term Allocator

Through the product term allocator, software automatically distributes product terms among the 16 macrocells in the logic block as needed. A total of 80 product terms are available from the local product term array. The product term allocator provides two important capabilities without affecting performance: product term steering and product term sharing.

*Product Term Steering*

Product term steering is the process of assigning product terms to macrocells as needed. For example, if one macrocell requires ten product terms while another needs just three, the product term allocator will "steer" ten product terms to one macrocell and three to the other. On FLASH370i devices, product terms are steered on an individual basis. Any number between 0 and 16 product terms can be steered to any macrocell. Note that 0 product terms is useful in cases where a particular macrocell is unused or used as an input register.

*Product Term Sharing*

Product term sharing is the process of using the same product term among multiple macrocells. For example, if more than one output has one or more product terms in its equation that are common to other outputs, those product terms are only programmed once. The FLASH370i product term allocator allows sharing across groups of four output macrocells in a variable fashion. The software automatically takes advantage of this capability—the user does not have to intervene. Note that greater usable density can often be achieved if the user "floats" the pin assignment. This allows the compiler to group macrocells that have common product terms adjacently.

Note that neither product term sharing nor product term steering have any effect on the speed of the product. All worst-case steering and sharing configurations have been incorporated in the timing specifications for the FLASH370i devices.

### FLASH370i Macrocell

*I/O Macrocell*

Within each logic block there are 8 or 16 I/O macrocells depending on the device used. *Figure 4* illustrates the architecture of the I/O macrocell. The macrocell features a register that can be configured as combinatorial, a D flip-flop, a T flip-flop, or a level-triggered latch.

The register can be asynchronously set or asynchronously reset at the logic block level with the separate set and reset product terms. Each of these product terms features programmable polarity. This allows the registers to be set or reset based on an AND expression or an OR expression.

Clocking of the register is very flexible. Depending on the device, either two or four global synchronous clocks are available to clock the register. Furthermore, each clock features programmable polarity so that registers can be triggered on falling as well as rising edges (see the Dedicated/Clock Inputs section). Clock polarity is chosen at the logic block level.

At the output of the macrocell, a polarity control mux is available to select active LOW or active HIGH signals. This has the added advantage of allowing significant logic reduction to occur in many applications.

The FLASH370i macrocell features a feedback path to the PIM separate from the I/O pin input path. This means that if the macrocell is buried (fed back internally only), the associated I/O pin can still be used as an input.

*Buried Macrocell*

Some of the devices in the FLASH370i family feature additional macrocells that do not feed individual I/O pins *Figure 5.* displays the architecture of the I/O and buried macrocells for these devices. The I/O macrocell is identical to the I/O macrocell for devices without buried macrocells.

The buried macrocell is very similar to the I/O macrocell. Again, it includes a register that can be configured as combinatorial, a D flip-flop, a T flip-flop, or a latch. The clock for this register has the same options as described for the I/O macrocell. The primary difference between the I/O macrocell and the buried macrocell is that the buried macrocell does not have the ability to output data directly to an I/O pin.

One additional difference on the buried macrocell is the addition of input register capability. The buried macrocell can be configured to act as an input register (D-type or latch) whose input comes from the I/O pin associated with the neighboring macrocell. The output of all buried macrocells is sent directly to the PIM regardless of its configuration.

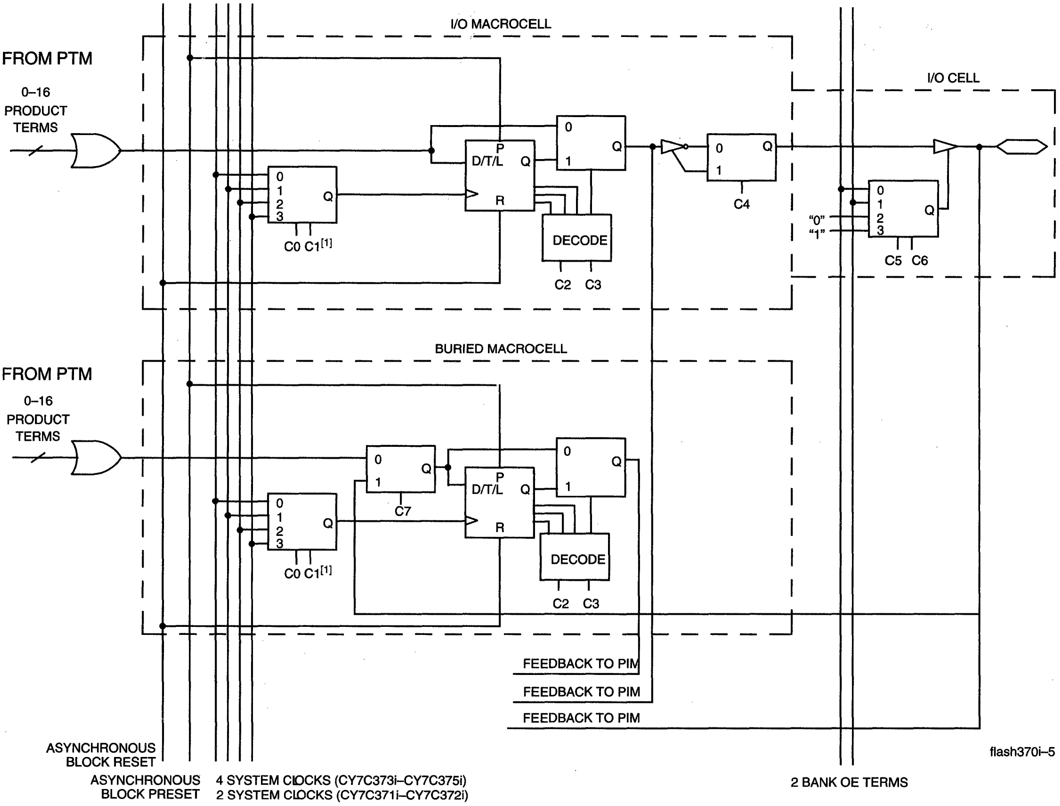


**Figure 4. I/O and Buried Macrocells**

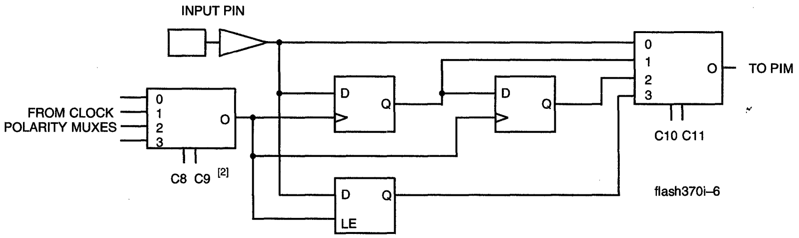


**Figure 5. Input Pins**

**Note:**

1. C1 is not used on the CY7C371i and CY7C372i since the mux size is 2:1.
2. C9 is not used on the CY7C371i and CY7C372i since the mux size is 2:1.
3. C8 and C9 are not included on the CY7C371i and CY7C372i since each input/clock pin has the other input/clock pin on its clock.
4. C15 and C16 are not used on the CY7C371i and CY7C372i since there are two clocks.

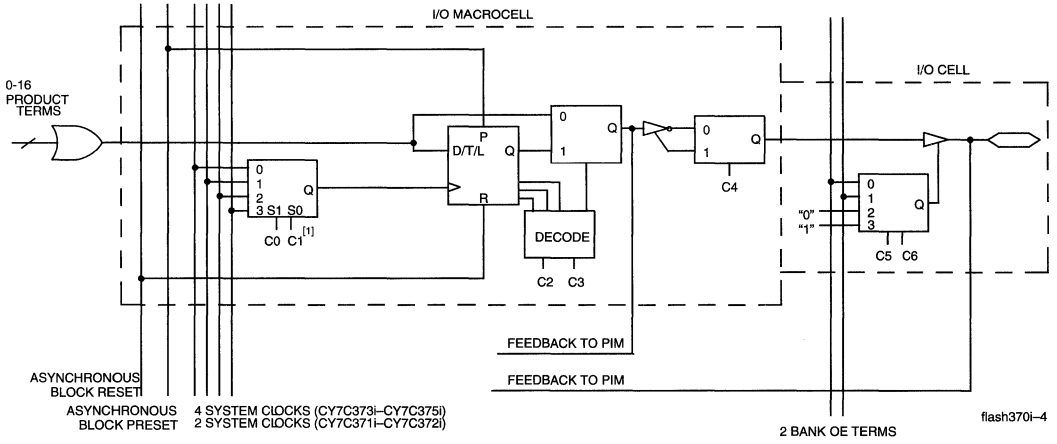


**Figure 6. I/O Macrocell**

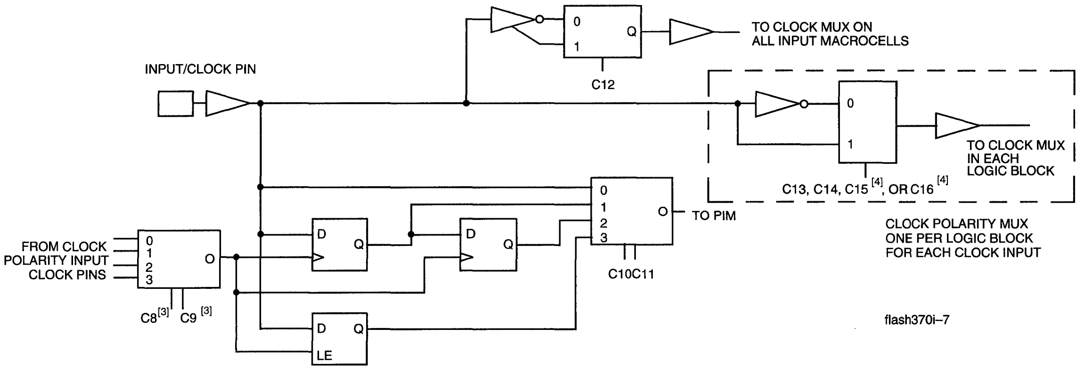


**Figure 7. Input/Clock Pins**

### FLASH370i I/O Cell

The I/O cell on the FLASH370i devices is illustrated along with the I/O macrocell in *Figures 6* and *4*. The user can program the I/O cell to change the way the three-state output buffer is enabled and/or disabled. Each output can be set permanently on (output only), permanently off (input only), or dynamically controlled by one of two OE product terms.

### Dedicated/Clock Inputs

Six pins on each member of the FLASH370i family are designated as input-only. There are two types of dedicated inputs on FLASH370i devices: input pins and input/clock pins. *Figure 5* illustrates the architecture for input pins. Four input options are available for the user: combinatorial, registered, double-registered, or latched. If a registered or latched option is selected, any one of the input clocks can be selected for control. *Figure 7* shows the input/clock macrocell. The CY7C371i and CY7C372i have two input/clock pins while the other devices in the family have four input/clock pins. Like the input pins, input/clock pins can be combinatorial, registered, double registered, or latched. In addition, these pins feed the clocking structures throughout the device. The clock path at the input is user-configurable in polarity. The polarity of the clock signal can also be controlled by the user. Note that this polarity is separately controlled for input registers and output registers.

### Timing Model

One of the most important features of the FLASH370i family is the simplicity of its timing. All delays are worst case and system performance is unaffected by the features used or notused

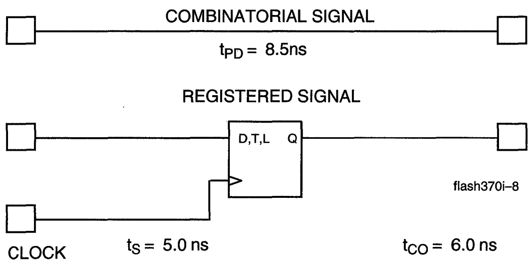


**Figure 8. Timing Model for CY7C371i**

on the parts. *Figure 8* illustrates the true timing model for the 8.5-ns devices. For combinatorial paths, any input to any output incurs an 8.5-ns worst-case delay regardless of the amount of logic used. For synchronous systems, the input set-up time to the output macrocells for any input is 5.0 ns and the clock to output time is also 6.0 ns. These measurements are for any output and clock, regardless of the logic used.

Stated another way, the FLASH370i features:

- no fanout delays
- no expander delays
- no dedicated vs. I/O pin delays
- no additional delay through PIM
- no penalty for using 0–16 product terms
- no added delay for steering product terms
- no added delay for sharing product terms
- no routing delays
- no output bypass delays

The simple timing model of the FLASH370i family eliminates unexpected performance penalties.

### PCI Compliance

The FLASH370i family of CMOS CPLDs are fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The simple and predictable timing model of FLASH370i ensures compliance with the PCI AC specifications independent of the design. On the other hand, in CPLD and FPGA architectures without simple and predictable timing, PCI compliance is dependent upon routing and product term distribution.

### Bus-Hold Capabilities on all I/Os and Dedicated Inputs

In addition to ISR capability, a new feature called bus-hold has been added to all FLASH370i I/Os and dedicated input pins. Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device's performance. As a latch, bus-hold recalls the last state of a pin when it is three-stated, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to $V_{CC}$ or $G_{ND}$.

## Development Software Support

### *Warp2*

*Warp2* is a state-of-the-art VHDL compiler for designing with Cypress PLDs, CPLDs, and FPGAs. *Warp2* utilizes a proper subset of IEEE 1164 VHDL as the Hardware Description Language (HDL) for design entry. VHDL provides a number of significant benefits for design entry. *Warp2* accepts VHDL input, synthesizes and optimizes the entered design, and outputs a JEDEC map for the desired FLASH370i device. For simulation, *Warp2* provides the graphical waveform simulator called Nova as well as VHDL and Verilog Timing Models.

VHDL (VHSIC Hardware Description Language) is an open, powerful, non-proprietary language that is a standard for behavioral design entry and simulation. It is already mandated for use by the Department of Defense and supported by every major vendor of CAE tools. VHDL allows designers to learn a single language that is useful for all facets of the design process. See the *Warp2* data sheet for further information.

### *Warp3*

*Warp3* is a sophisticated development system that is based on the latest version of ViewLogic's CAE design environment. *Warp3* features schematic capture (ViewDraw), VHDL waveform simulation (ViewSim), a VHDL debugger, and VHDL synthesis, all integrated in a graphical design environment. *Warp3* is available on PCs using Windows 3.1 or subsequent versions and on Sun and Hewlett Packard workstations. See the *Warp3* data sheet for further information.

### *Warp2*Sim™

This development system includes the capbilities of *Warp2* and ViewLogic's ViewSim package which provides dynamic timing solutions for all Cypress PLDs, CPLDs, and FPGAs.

### Third-Party Software

Cypress maintains a strong commitment to third-party design software vendors. All major third-party software vendors (including ABEL™, LOG/iC™, CUPL™, and Minc) will provide support for the FLASH370i family of devices. To expedite this support, Cypress supplies vendors with all pertinent architectural information as well as design fitters for our products.

## Programming

There are four programming options available for FLASH370i devices. The first is to use a PC with the FLASH370i ISR programming cable and software. With this method, the ISR pins of the FLASH370i devices are routed to a connector at the edge of the printed circuit board. The ISR programming cable is then connected between the parallel port of the PC and this connector. A simple configuration file instructs the ISR software of the programming operations to be performed on each of the FLASH370i devices in the system. The ISR software then automatically completes all of the necessary data manipulations required to accomplish the programming, reading, verifying, and other ISR functions. For more information on Cypress ISR Interface, see the ISR Programming Kit.

The second method for programming FLASH370i devices is on an ATE (automatic test equipment). If this method is selected, the FLASH370i ISR software generates programming test vectors for the target tester.

The third programming option for FLASH370i devices is to utilize the embedded controller or processor that already exists in the system. The FLASH370i ISR software assists in this method by converting the device JEDEC maps into the ISR serial stream that contains the ISR instruction information and the addresses and data of locations to be programmed. The embedded controller then simply directs this ISR stream to the chain of FLASH370i devices to complete the desired reconfiguring or diagnostic operations.

The fourth method for programming FLASH370i devices is to use the same parallel programmer that is currently being used to program FLASH370 devices. Since the programming algorithms are the same, normal third party programming support for the FLASH370i family is readily available.

For all pinout, electrical, and timing requirements, refer to device data sheets. For ISR cable and software specifications, refer to ISR Kit data sheets. For detailed description of ISR capabilities, refer to the Cypress application note, "An Introduction to In System Reprogramming with FLASH370i."

The *Impulse3*™ device programmer from Cypress will program all Cypress PLDs, CPLDs, FPGAs, and PROMs. This unit is a programmer that connects to any IBM-compatible PC via the printer port. For further information see the *Impulse3* data sheet.

### Third-Party Programmers

As with development software, Cypress supports third-party programmers. All major third-party programmers (including Data I/O, Logical Devices, Minato, SMS, and Stag) will support the FLASH370i family.

Document #: 38-00493-B

ISR, UltraLogic, FLASH370i, *Warp2*Sim, and *Impulse3* are trademarks of Cypress Semiconductor Corporation.
*Warp2, Warp3* are registered trademarks of Cypress Semiconductor Corporation.

ViewSim and ViewDraw are trademarks of ViewLogic.
ABEL is a trademark of Data I/O Corporation.
LOG/iC is a trademark of Isdata Corporation.
CUPL is a trademark of Logical Devices, Inc.

CYPRESS *PRELIMINARY* **CY7C371i**

# UltraLogic™ 32-Macrocell Flash CPLD

## Features

- **32 macrocells in two logic blocks**
- **32 I/O pins**
- **5 dedicated inputs including 2 clock pins**
- **In-System Reprogrammable (ISR™) Flash technology**
  - **—JTAG interface**
- **Bus Hold capabilities on all I/Os and dedicated inputs**
- **No hidden delays**
- **High speed**
  - **—$f_{MAX}$ = 143 MHz**
  - **—$t_{PD}$= 8.5 ns**
  - **—$t_S$ = 5 ns**
  - **—$t_{CO}$ = 6 ns**
- **Fully PCI compliant**
- **Available in 44-pin PLCC, and TQFP packages**
- **Pin compatible with the CY7C372i**

## Functional Description

The CY7C371i is an In-System Reprogrammable Complex Programmable Logic Device (CPLD) and is part of the FLASH370i™ family of high-density, high-speed CPLDs. Like all members of the FLASH370i family, the CY7C371i is designed to bring the ease of use and high performance of the 22V10, as well as PCI Local Bus Specification support, to high-density CPLDs.

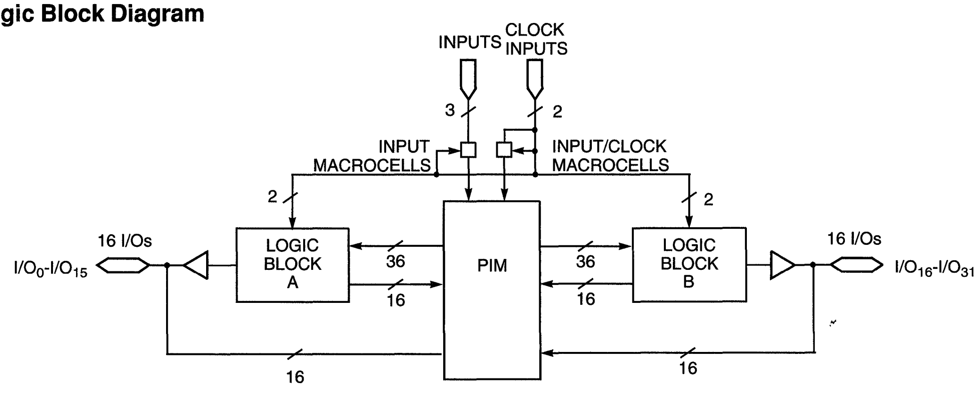
Like all of the UltraLogic™ FLASH370i devices, the CY7C371i is electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows, thereby reducing costs. The Cypress ISR function is implemented through a JTAG serial interface. Data is shifted in and out through the SDI and SDO pins. The ISR interface is enabled using the programming voltage pin ($ISR_{EN}$). Additionally, because of the superior routability of the FLASH370i devices, ISR often allows users to change existing logic designs while simultaneously fixing pinout assignments.

The 32 macrocells in the CY7C371i are divided between two logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370i architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

Like all members of the FLASH370i family, the CY7C371i is rich in I/O resources. Each macrocell in the device features an associated I/O pin, resulting in 32 I/O pins on the CY7C371i. In addition, there are three dedicated inputs and two input/clock pins.

## Logic Block Diagram

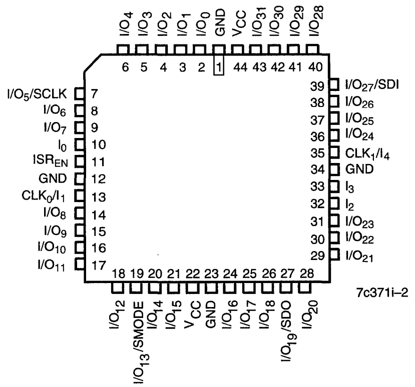


## Selection Guide

| | | 7C371i-143 | 7C371i-110 | 7C371i-83 | 7C371iL-83 | 7C371i-66 | 7C371iL-66 |
|---|---|---|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | | 8.5 | 10 | 12 | 12 | 15 | 15 |
| Minimum Set-Up, $t_S$ (ns) | | 5 | 6 | 8 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | | 6 | 6.5 | 8 | 8 | 10 | 10 |
| Typical Supply Current, $I_{CC}$ (mA) | Comm./Ind. | 75 | 75 | 75 | 45 | 75 | 45 |

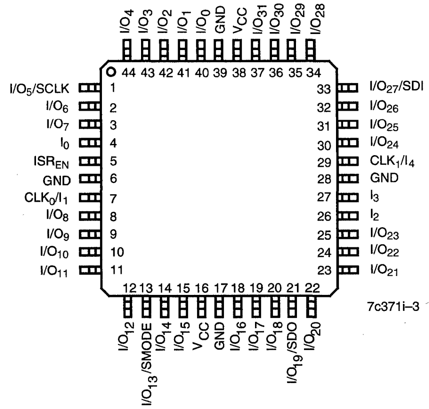
## Pin Configurations

PLCC
TopView



7c371i-2

TQFP
TopView



7c371i-3

## Functional Description (continued)

Finally, the CY7C371i features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used or the type of application, the timing parameters on the CY7C371i remain the same.

### Logic Block

The number of logic blocks distinguishes the members of the FLASH370i family. The CY7C371i includes two logic blocks. Each logic block is constructed of a product term array, a product term allocator, and 16 macrocells.

*Product Term Array*

The product term array in the FLASH370i logic block includes 36 inputs from the PIM and outputs 86 product terms to the product term allocator. The 36 inputs from the PIM are available in both positive and negative polarity, making the overall array size 72 x 86. This large array in each logic block allows for very complex functions to be implemented in a single pass through the device.

*Product Term Allocator*

The product term allocator is a dynamic, configurable resource that shifts product terms to macrocells that require them. Any number of product terms between 0 and 16 inclusive can be assigned to any of the logic block macrocells (this is called product term steering). Furthermore, product terms can be shared among multiple macrocells. This means that product terms that are common to more than one output can be implemented in a single product term. Product term steering and product term sharing help to increase the effective density of the FLASH370i CPLDs. Note that product term allocation is handled by software and is invisible to the user.

*I/O Macrocell*

Each of the macrocells on the CY7C371i has a separate associated I/O pin. The input to the macrocell is the sum of between 0 and 16 product terms from the product term allocator. The macrocell includes a register that can be optionally bypassed. It also has polarity control, and two global clocks to trigger the register. The macrocell also features a separate feedback path to the PIM so that the register can be buried if the I/O pin is used as an input.

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) connects the two logic blocks on the CY7C371i to the inputs and to each other. All inputs (including feedbacks) travel through the PIM. There is no speed penalty incurred by signals traversing the PIM.

### Programming

For an overview of ISR programming, refer to the FLASH370i Family data sheet and for ISR cable and software specifications, refer to ISR data sheets. For a detailed description of ISR capabilities, refer to the Cypress application note, "An Introduction to In System Reprogramming with FLASH370i."

### PCI Compliance

The FLASH370i family of CMOS CPLDs are fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The simple and predictable timing model of FLASH370i ensures compliance with the PCI AC specifications independent of the design. On the other hand, in CPLD and FPGA architectures without simple and predictable timing, PCI compliance is dependent upon routing and product term distribution.

### Bus Hold Capabilities on all I/Os and Dedicated Inputs

In addition to ISR capability, a new feature called bus-hold has been added to all FLASH370i I/Os and dedicated input pins. Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device's performance. As a latch, bus-hold recalls the last state of a pin when it is three-stated, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to $V_{CC}$ or GND.

### Design Tools

Development software for the CY7C371i is available from Cypress's *Warp2*®, *Warp2*Sim™, and *Warp3*® software packages. All of these products are based on the IEEE-standard VHDL language. Cypress also actively supports third-party design tools such as ABEL™, CUPL™, MINC, and LOG/iC™. Please refer to third-party tool support for further information.

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | –65°C to +150°C |
| Ambient Temperature with Power Applied | –55°C to +125°C |
| Supply Voltage to Ground Potential | –0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | –0.5V to +7.0V |
| DC Input Voltage | –0.5V to +7.0V |
| DC Program Voltage | 12.5V |
| Output Current into Outputs (LOW) | 16 mA |
| Static Discharge Voltage (per MIL-STD-883, Method 3015) | >2001V |
| Latch-Up Current | >200 mA |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | –40°C to +85°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[1]

| Parameter | Description | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = –3.2 mA (Com'l/Ind)[2] | 2.4 | | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA (Com'l/Ind)[2] | | | 0.5 | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH Voltage for all inputs[3] | | 2.0 | | 7.0 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW Voltage for all inputs[3] | | –0.5 | | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I$ = $V_{CC}$ | | –10 | | +10 | μA |
| $I_{OZ}$ | Output Leakage Current | GND ≤ $V_O$ ≤ $V_{CC}$, Output Disabled | | –50 | | +50 | μA |
| $I_{OS}$ | Output Short Circuit Current[4,5] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | –30 | | –160 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 mHz, $V_{IN}$ = GND, $V_{CC}$[6] | Com'l/Ind. | | 75 | 125 | mA |
| | | | Com'l "L" –66, –83 | | 45 | 75 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | | μA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | –75 | | | μA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | | +500 | μA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | | –500 | μA |

**Notes:**

[1] See the last page of this specification for Group A subgroup testing information.

[2] $I_{OH}$ = –2 mA, $I_{OL}$ = 2 mA for SDO.

[3] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[4] Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[5] Tested initially and after any design or process changes that may affect these parameters.

[6] Measured with 16-bit counter programmed into each logic block.

## Capacitance[5]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{I/O}$[7] | Input Capacitance | $V_{IN}$ = 5.0V at f=1 MHz | | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 5 | 12 | pF |

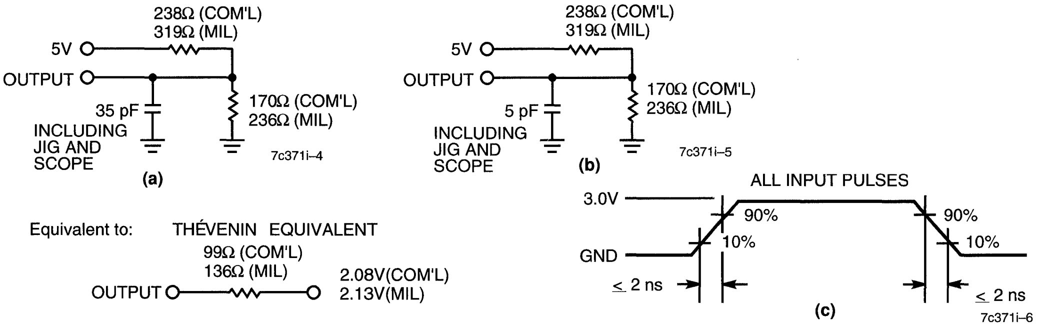
## Inductance[5]

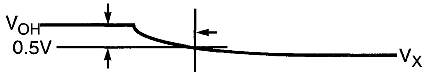
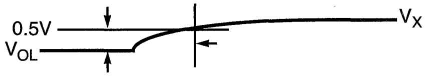
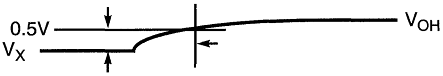
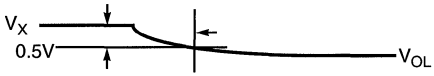
| Parameter | Description | Test Conditions | 44-Lead TQFP | 44-Lead PLCC | Unit |
|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f= 1 MHz | 2 | 5 | nH |

## Endurance Characteristics[5]

| Parameter | Description | Test Conditions | Min. | Unit |
|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | Cycles |

## AC Test Loads and Waveforms



| Parameter[8] | Vx | Output Waveform Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | 1.5V |  |
| $t_{EA(-)}$ | $V_{the}$ |  |

**Note:**

[7] CI/O for $ISR_{EN}$ is 15 pF Max.

[8] $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF ACTest Load.

## Switching Characteristics Over the Operating Range[9]

| Parameter | Description | 7C371i–143 | | 7C371i–110 | | 7C371i–83<br>7C371iL–83 | | 7C371i–66<br>7C371iL–66 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | | | | | |
| $t_{PD}$ | Input to Combinatorial Output | | 8.5 | | 10 | | 12 | | 15 | ns |
| $t_{PDL}$ | Input to Output Through Transparent Input or Output Latch | | 11.5 | | 13 | | 18 | | 22 | ns |
| $t_{PDLL}$ | Input to Output Through Transparent Input and Output Latches | | 13.5 | | 15 | | 20 | | 24 | ns |
| $t_{EA}$ | Input to Output Enable | | 13 | | 14 | | 19 | | 24 | ns |
| $t_{ER}$ | Input to Output Disable | | 13 | | 14 | | 19 | | 24 | ns |
| **Input Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[5] | 2.5 | | 3 | | 4 | | 5 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[5] | 2.5 | | 3 | | 4 | | 5 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{ICO}$ | Input Register Clock or Latch Enable to Combinatorial Output | | 12 | | 14 | | 19 | | 24 | ns |
| $t_{ICOL}$ | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 14 | | 16 | | 21 | | 26 | ns |
| **Output Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{CO}$ | Clock or Latch Enable to Output | | 6 | | 6.5 | | 8 | | 10 | ns |
| $t_S$ | Set-Up Time from Input to Clock or Latch Enable | 5 | | 6 | | 8 | | 10 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{CO2}$ | Output Clock or Latch Enable to Output Delay (Through Memory Array) | | 12 | | 14 | | 19 | | 24 | ns |
| $t_{SCS}$ | Output Clock or Latch Enable to Output Clock or Latch Enable (Through Memory Array) | 7 | | 9 | | 12 | | 15 | | ns |
| $t_{SL}$ | Set-Up Time from Input Through Transparent Latch to Output Register Clock or Latch Enable | 9 | | 10 | | 12 | | 15 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Clock or Latch Enable | 0 | | 0 | | 0 | | 0 | | ns |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[5] | 143 | | 111 | | 83.3 | | 66.6 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$)[5] | 166.7 | | 153.8 | | 100 | | 83.3 | | MHz |
| $f_{MAX3}$ | Maximum Frequency with external feedback (Lesser of $1/(t_{CO} + t_S)$ and $1/(t_{WL} + t_{WH})$)[5] | 91 | | 80 | | 50 | | 41.6 | | MHz |

**Note:**

[9] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

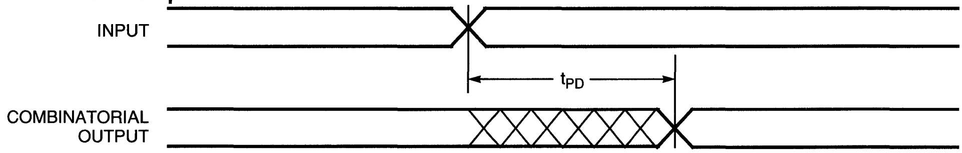
[10] This specification is intended to guarantee interface compatibility of the other members of the CY7C370i family with the CY7C371i. This specification is met for the devices operating at the same ambient temperature and at the same power supply voltage.

## Switching Characteristics Over the Operating Range[9] (continued)

| Parameter | Description | 7C371i–143 |  | 7C371i–110 |  | 7C371i–83<br>7C371iL–83 |  | 7C371i–66<br>7C371iL–66 |  | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. |  |
| $t_{OH}$-$t_{IH}$ 37x | Output Data Stable from Output clock Minus Input Register Hold Time for 7C37x[5,10] | 0 |  | 0 |  | 0 |  | 0 |  | ns |
| **Pipelined Mode Parameters** |  |  |  |  |  |  |  |  |  |  |
| $t_{ICS}$ | Input Register Clock to Output Register Clock | 7 |  | 9 |  | 12 |  | 15 |  | ns |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 125 |  | 111 |  | 76.9 |  | 62.5 |  | MHz |
| **Reset/Preset Parameters** |  |  |  |  |  |  |  |  |  |  |
| $t_{RW}$ | Asynchronous Reset Width[5] | 8 |  | 10 |  | 15 |  | 20 |  | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[5] | 10 |  | 12 |  | 17 |  | 22 |  | ns |
| $t_{RO}$ | Asynchronous Reset to Output |  | 14 |  | 16 |  | 21 |  | 26 | ns |
| $t_{PW}$ | Asynchronous Preset Width[5] | 8 |  | 10 |  | 15 |  | 20 |  | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[5] | 10 |  | 12 |  | 17 |  | 22 |  | ns |
| **Tap Controller Parameters** |  |  |  |  |  |  |  |  |  |  |
| $f_{TAP}$ | Tap Controller Frequency | 500 |  | 500 |  | 500 |  | 500 |  | kHz |

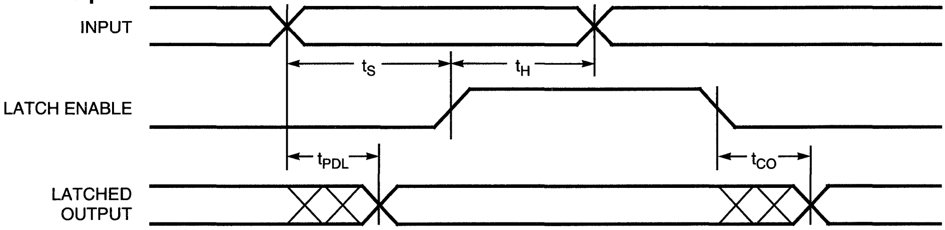
## Switching Waveforms

### Combinatorial Output



7c371i–7

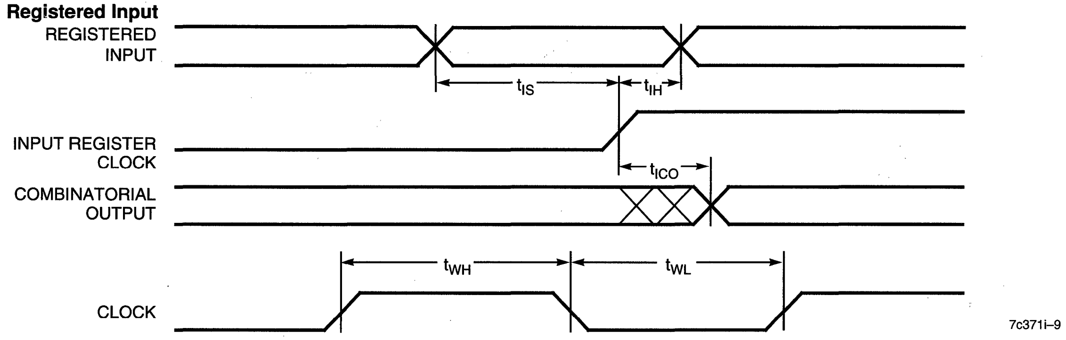
### Latched Output



7c371i–8

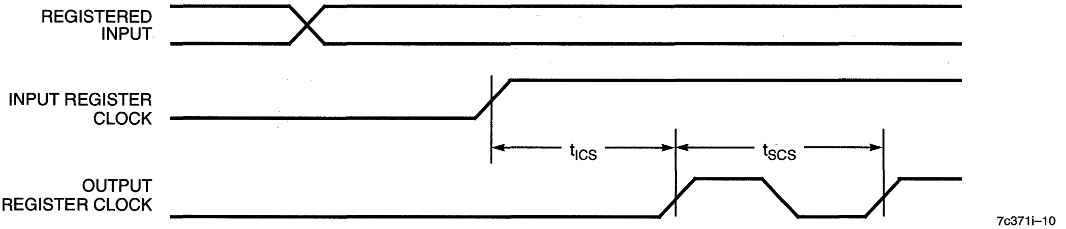
## Switching Waveforms (continued)

### Registered Input



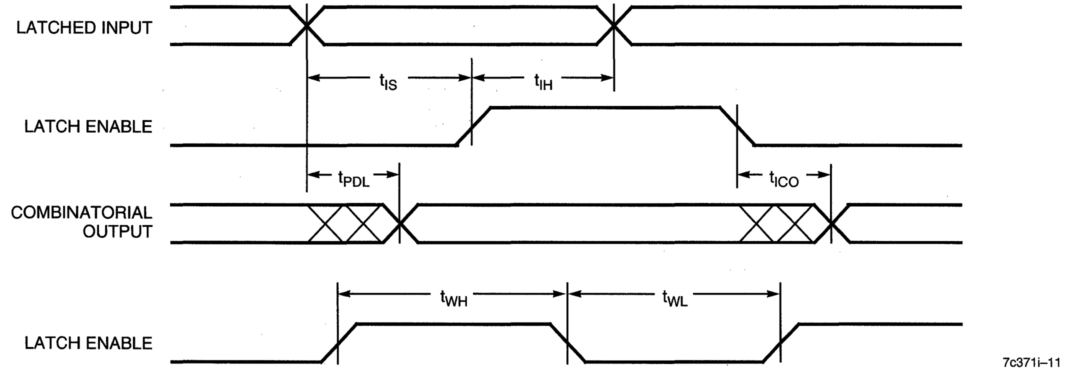
7c371i–9

### Clock to Clock



7c371i–10

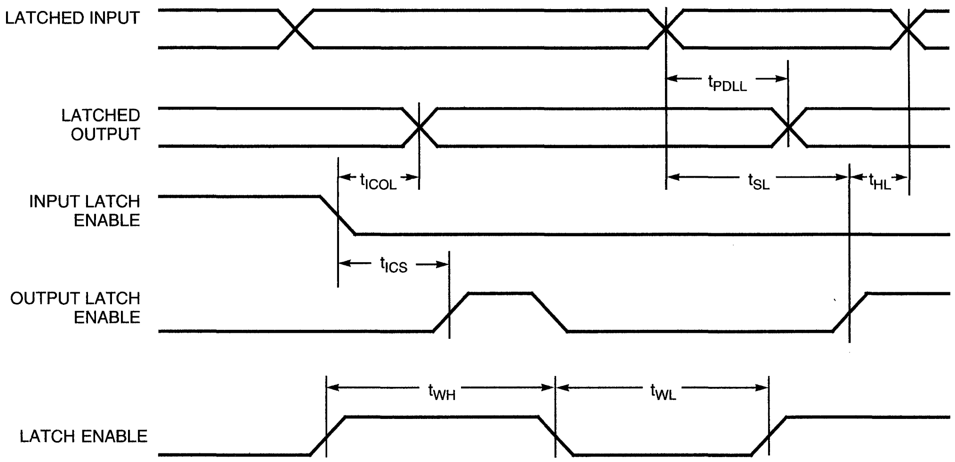
### Latched Input



7c371i–11

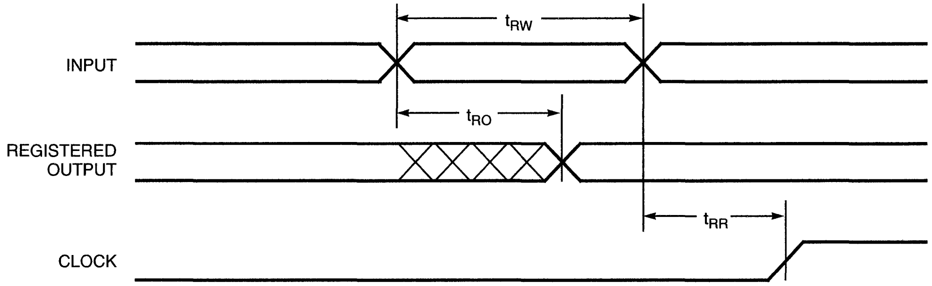
## Switching Waveforms (continued)

### Latched Input and Output



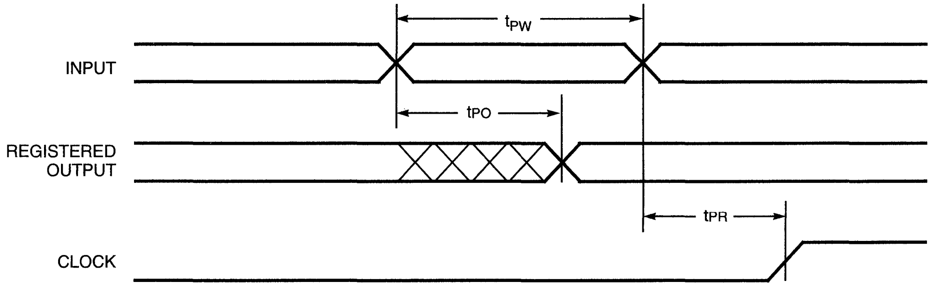
7c371i–12

### Asynchronous Reset



7c371i–13

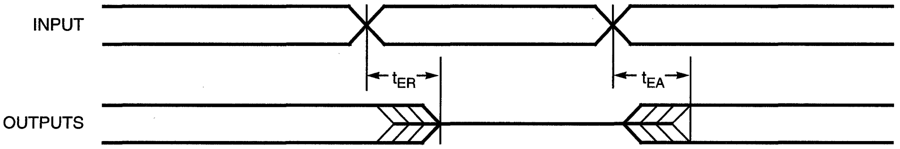
### Asynchronous Preset



7c371i–14

## Switching Waveforms (continued)

### Output Enable/Disable



7c371i–16

## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 143 | CY7C371i–143AC | A44 | 44-Lead Thin Plastic Quad Flat Pack | Commercial |
| | CY7C371i–143JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| 110 | CY7C371i–110AC | A44 | 44-Lead Thin Plastic Quad Flat Pack | Commercial |
| | CY7C371i–110JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| 83 | CY7C371i–83AC | A44 | 44-Lead Thin Plastic Quad Flat Pack | Commercial |
| | CY7C371i–83JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C371i–83AI | A44 | 44-Lead Thin Plastic Quad Flat Pack | Industrial |
| | CY7C371i–83JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C371iL–83AC | A44 | 44-Lead Thin Plastic Quad Flat Pack | Commercial |
| | CY7C371iL–83JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C371iL–83AI | A44 | 44-Lead Thin Plastic Quad Flat Pack | Industrial |
| | CY7C371iL–83JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| 66 | CY7C371i–66AC | A44 | 44-Lead Thin Plastic Quad Flat Pack | Commercial |
| | CY7C371i–66JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C371i–66AI | A44 | 44-Lead Thin Plastic Quad Flat Pack | Industrial |
| | CY7C371i–66JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C371iL–66AC | A44 | 44-Lead Thin Plastic Quad Flat Pack | Commercial |
| | CY7C371iL–66JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C371iL–66AI | A44 | 44-Lead Thin Plastic Quad Flat Pack | Industrial |
| | CY7C371iL–66JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |

Document #: 38–00497–B

*Warp2*Sim, FLASH370, FLASH370i, ISR, and UltraLogic are trademarks of Cypress Semiconductor Corporation
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.
ABEL is a trademark of Data I/O Corporation.
LOG/iC is a trademark of Isdata Corporation.
CUPL is a trademark of Logical Devices Incorporated.

CYPRESS

# CY7C372i

# UltraLogic™ 64-Macrocell Flash CPLD

## Features

- 64 macrocells in four logic blocks
- 32 I/O pins
- 5 dedicated inputs including 2 clock pins
- In-System Reprogrammable (ISR™) Flash technology
  - —JTAG interface
- Bus Hold capabilities on all I/Os and dedicated inputs
- No hidden delays
- High speed
  - —$f_{MAX}$ = 125 MHz
  - —$t_{PD}$ = 10 ns
  - —$t_S$ = 5.5 ns
  - —$t_{CO}$ = 6.5 ns
- Fully PCI compliant
- Available in 44-pin PLCC and CLCC packages
- Pin compatible with the CY7C371i

## Functional Description

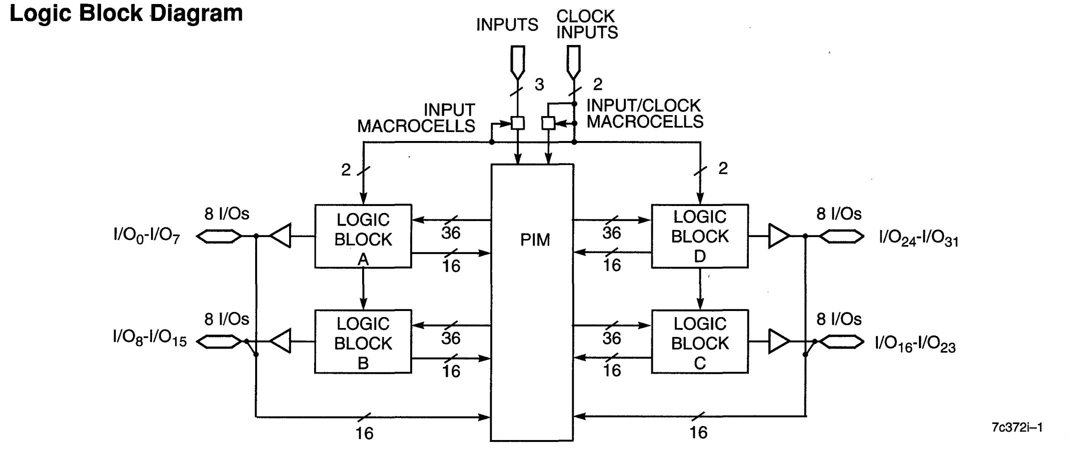
The CY7C372i is an In-System Reprogrammable Complex Programmable Logic Device (CPLD) and is part of the FLASH370i™ family of high-density, high-speed CPLDs. Like all members of the FLASH370i family, the CY7C372i is designed to bring the ease of use and high performance of the 22V10, as well as PCI Local Bus Specification support, to high-density CPLDs.

Like all of the UltraLogic FLASH370i devices, the CY7C372i is electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows, thereby reducing costs. The Cypress ISR function is implemented through a JTAG serial interface. Data is shifted in and out through the SDI and SDO pins. The ISR interface is enabled using the programming voltage pin ($ISR_{EN}$). Additionally, because of the superior routability of the FLASH370i devices, ISR often allows users to change existing logic designs while simultaneously fixing pinout assignments.

The 64 macrocells in the CY7C372i are divided between four logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370i architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

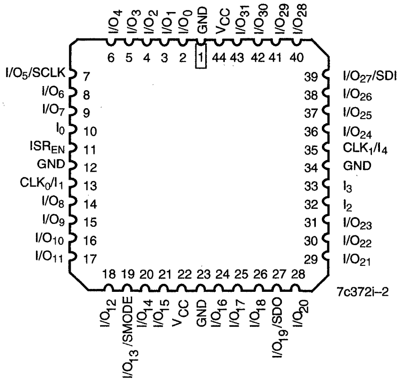
## Logic Block Diagram



## Selection Guide

| | 7C372i-125 | 7C372i-100 | 7C372i-83 | 7C372i-66 | 7C372iL-66 |
|---|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | 10 | 12 | 15 | 20 | 20 |
| Minimum Set-up, $t_S$ (ns) | 5.5 | 6.0 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 6.5 | 6.5 | 8 | 10 | 10 |
| Typical Supply Current, $I_{CC}$ (mA) | 75 | 75 | 75 | 75 | 45 |

## Pin Configuration (continued)



7c372i–2

## Functional Description (continued)

Like all members of the FLASH370i family, the CY7C372i is rich in I/O resources. Every two macrocells in the device feature an associated I/O pin, resulting in 32 I/O pins on the CY7C372i. In addition, there are three dedicated inputs and two input/clock pins.

Finally, the CY7C372i features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used. or the type of application, the timing parameters on the CY7C372i remain the same.

### Logic Block

The number of logic blocks distinguishes the members of the FLASH370i family. The CY7C372i includes four logic blocks. Each logic block is constructed of a product term array, a product term allocator, and 16 macrocells.

*Product Term Array*

The product term array in the FLASH370i logic block includes 36 inputs from the PIM and outputs 86 product terms to the product term allocator. The 36 inputs from the PIM are available in both positive and negative polarity, making the overall array size 72 x 86. This large array in each logic block allows for very complex functions to be implemented in a single pass through the device.

*Product Term Allocator*

The product term allocator is a dynamic, configurable resource that shifts product terms to macrocells that require them. Any number of product terms between 0 and 16 inclusive can be assigned to any of the logic block macrocells (this is called product term steering). Furthermore, product terms can be shared among multiple macrocells. This means that product terms that are common to more than one output can be implemented in a single product term. Product term steering and product term sharing help to increase the effective density of the FLASH370 PLDs. Note that product term allocation is handled by software and is invisible to the user.

*I/O Macrocell*

Half of the macrocells on the CY7C372i have separate I/O pins associated with them. In other words, each I/O pin is shared by two macrocells. The input to the macrocell is the sum of between 0 and 16 product terms from the product term allocator. The macrocell includes a register that can be optionally bypassed. It also has polarity control, and two global clocks to trigger the register. The I/O macrocell also features a separate feedback path to the PIM so that the register can be buried if the I/O pin is used as an input.

*Buried Macrocell*

The buried macrocell is very similar to the I/O macrocell. Again, it includes a register that can be configured as combinatorial, as a D flip-flop, a T flip-flop, or a latch. The clock for this register has the same options as described for the I/O macrocell. One difference on the buried macrocell is the addition of input register capability. The user can program the buried macrocell to act as an input register (D-type or latch) whose input comes from the I/O pin associated with the neighboring macrocell. The output of all buried macrocells is sent directly to the PIM regardless of its configuration.

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) connects the four logic blocks on the CY7C372i to the inputs and to each other. All inputs (including feedbacks) travel through the PIM. There is no speed penalty incurred by signals traversing the PIM.

### Programming

For an overview of ISR programming, refer to the FLASH370i Family data sheet and for ISR cable and software specifications, refer to ISR data sheets. For a detailed description of ISR capabilities, refer to the Cypress application note, "An Introduction to In System Reprogramming with FLASH370i."

### PCI Compliance

The FLASH370i family of CMOS CPLDs are fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The simple and predictable timing model of FLASH370i ensures compliance with the PCI AC specifications independent of the design. On the other hand, in CPLD and FPGA architectures without simple and predictable timing, PCI compliance is dependent upon routing and product term distribution.

### Bus Hold Capabilities on all I/Os and Dedicated Inputs

In addition to ISR capability, a new feature called bus-hold has been added to all FLASH370i I/Os and dedicated input pins. Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device's performance. As a latch, bus-hold recalls the last state of a pin when it is three-stated, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to VCC or GND.

### Design Tools

Development software for the CY7C372i is available from Cypress's *Warp2*®, *Warp2*Sim™, and *Warp3*® software packages. Both of these products are based on the IEEE standard VHDL language. Cypress also supports third-party vendors such as ABEL™, CUPL™, and LOG/iC™. Please refer to thrid-party tool support data sheets for further information.

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................–65°C to +150°C

Ambient Temperature with Power Applied ............................................–55°C to +125°C

Supply Voltage to Ground Potential ..............–0.5V to +7.0V

DC Voltage Applied to Outputs in High Z State ..............................................–0.5V to +7.0V

DC Input Voltage..........................................–0.5V to +7.0V

DC Program Voltage....................................................12.5V

Output Current into Outputs .......................................16 mA

Static Discharge Voltage...........................................>2001V (per MIL–STD–883, Method 3015)

Latch-Up Current ...................................................>200 mA

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | –40°C to +85°C | 5V ± 10% |
| Military[1] | –55°C to +125°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = –3.2 mA (Com'l/Ind)[3] | 2.4 | | | V |
| | | | $I_{OH}$ = –2.0 mA (Mil) | 2.4 | | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA (Com'l/Ind)[3] | | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA (Mil) | | | 0.5 | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH Voltage for all Inputs[4] | | 2.0 | | 7.0 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW Voltage for all Inputs[4] | | –0.5 | | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I = V_{CC}$ | | –10 | | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | GND $\leq V_O \leq V_{CC}$, Output Disabled | | –50 | | +50 | µA |
| $I_{OS}$ | Output Short Circuit Current[5, 6] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | –30 | | –160 | mA |
| $I_{CC}$ | Power Supply Current [7] | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 mHz, $V_{IN}$ = GND, $V_{CC}$ | Com'l/ind. | | 75 | 125 | mA |
| | | | Com'l "L" –66 | | 45 | 75 | mA |
| | | | Military | | 75 | 200 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | | µA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | –75 | | | µA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | | +500 | µA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | | –500 | µA |

## Capacitance[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{I/O}$[8, 9] | Input Capacitance | $V_{IN}$ = 5.0V at f=1 MHz | | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 5 | 12 | pF |

**Notes:**

1. $T_A$ is the "instant on" case temperature.
2. See the last page of this specification for Group A subgroup testing information.
3. For SDO: $I_{OH}$ = –2 mA, $I_{OL}$ = 2 mA.
4. These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.
5. Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
6. Tested initially and after any design or process changes that may affect these parameters.
7. Meaured with 16-bit counter programmed into each logic block.
8. $C_{I/O}$ for dedicated Inputs, and for I/O pins with JTAG functionality is 12 pF Max., and for $ISR_{EN}$ is 15pF Max.
9. $C_{I/O}$ for CLCC package is 15 pF Max.

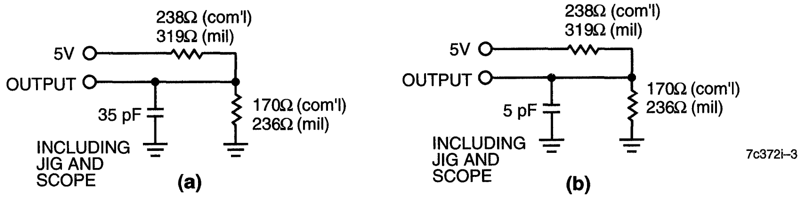
## Inductance[6]

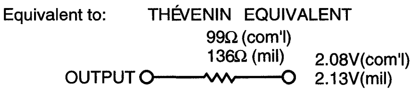
| Parameter | Description | Test Conditions | 44-Lead CLCC | 44-Lead PLCC | Unit |
|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f=1 MHz | 2 | 5 | nH |

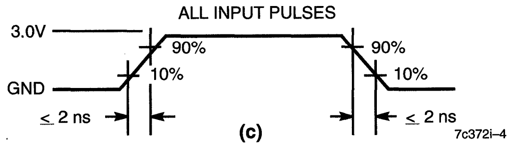
## Endurance Characteristics[6]

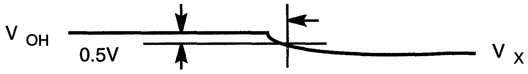
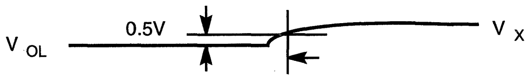
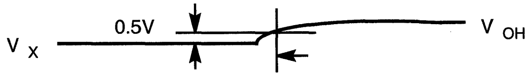
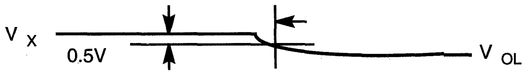
| Parameter | Description | Test Conditions | Min. | Unit |
|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | Cycles |

## AC Test Loads and Waveforms







<table>
<tr><th>Parameter[10]</th><th>$V_x$</th><th>Output Waveform Measurement Level</th></tr>
<tr><td>$t_{ER(-)}$</td><td>1.5V</td><td>
</td></tr>
<tr><td>$t_{ER(+)}$</td><td>2.6V</td><td>
</td></tr>
<tr><td>$t_{EA(+)}$</td><td>1.5V</td><td>
</td></tr>
<tr><td>$t_{EA(-)}$</td><td>$V_{the}$</td><td>
</td></tr>
</table>

**(d) Test Waveforms**

**Notes:**

10. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[11]

| | | 7C372i-125 | | 7C372i-100 | | 7C372i-83 | | 7C372i-66<br>7C372iL-66 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| **Combinatorial Mode Parameters** | | | | | | | | | | |
| $t_{PD}$ | Input to Combinatorial Output | | 10 | | 12 | | 15 | | 20 | ns |
| $t_{PDL}$ | Input to Output Through Transparent Input or Output Latch | | 13 | | 15 | | 18 | | 22 | ns |
| $t_{PDLL}$ | Input to Output Through Transparent Input and Output Latches | | 15 | | 16 | | 19 | | 24 | ns |
| $t_{EA}$ | Input to Output Enable | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{ER}$ | Input to Output Disable | | 14 | | 16 | | 19 | | 24 | ns |
| **Input Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[6] | 3 | | 3 | | 4 | | 5 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[6] | 3 | | 3 | | 4 | | 5 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{ICO}$ | Input Register Clock or Latch Enable to Combinatorial Output | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{ICOL}$ | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 16 | | 18 | | 21 | | 26 | ns |
| **Output Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{CO}$ | Clock or Latch Enable to Output | | 6.5 | | 6.5 | | 8 | | 10 | ns |
| $t_S$ | Set-Up Time from Input to Clock or Latch Enable | 5.5 | | 6 | | 8 | | 10 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{CO2}$ | Output Clock or Latch Enable to Output Delay (Through Memory Array) | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{SCS}$ | Output Clock or Latch Enable to Output Clock or Latch Enable (Through Memory Array) | 8 | | 10 | | 12 | | 15 | | ns |
| $t_{SL}$ | Set-Up Time from Input Through Transparent Latch to Output Register Clock or Latch Enable | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Clock or Latch Enable | 0 | | 0 | | 0 | | 0 | | ns |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback in Output Registered Mode (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[6] | 125 | | 100 | | 83 | | 66 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$)[6] | 153.8 | | 153.8 | | 125 | | 100 | | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ and $1/(t_{WL} + t_{WH})$)[6] | 83.3 | | 80 | | 62.5 | | 50 | | MHz |
| $t_{OH}$-$t_{IH}$ 37x | Output Data Stable from Output clock Minus Input Register Hold Time for 7C37x[6, 12] | 0 | | 0 | | 0 | | 0 | | ns |

**Note:**

[11] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

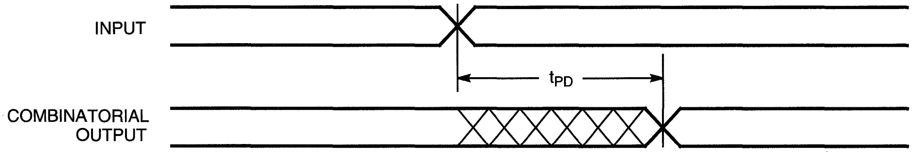
[12] This specification is intended to guarantee interface compatibility of the other members of the CY7C370i family with the CY7C372i. This specification is met for the devices operating at the same ambient temperature and at the same power supply voltage.

## Switching Characteristics Over the Operating Range[11] (continued)

| Parameter | Description | 7C372i-125 | | 7C372i-100 | | 7C372i-83 | | 7C372i-66<br>7C372iL-66 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Pipelined Mode Parameters** | | | | | | | | | | |
| $t_{ICS}$ | Input Register Clock to Output Register Clock | 8 | | 10 | | 12 | | 15 | | ns |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$)[6] | 125 | | 100 | | 83.3 | | 66.6 | | MHz |
| **Reset/Preset Parameters** | | | | | | | | | | |
| $t_{RW}$ | Asynchronous Reset Width[6] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[6] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{RO}$ | Asynchronous Reset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| $t_{PW}$ | Asynchronous Preset Width[6] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[6] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{PO}$ | Asynchronous Preset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| **Tap Controller Parameter** | | | | | | | | | | |
| $f_{TAP}$ | Tap Controller Frequency | 500 | | 500 | | 500 | | 500 | | kHz |

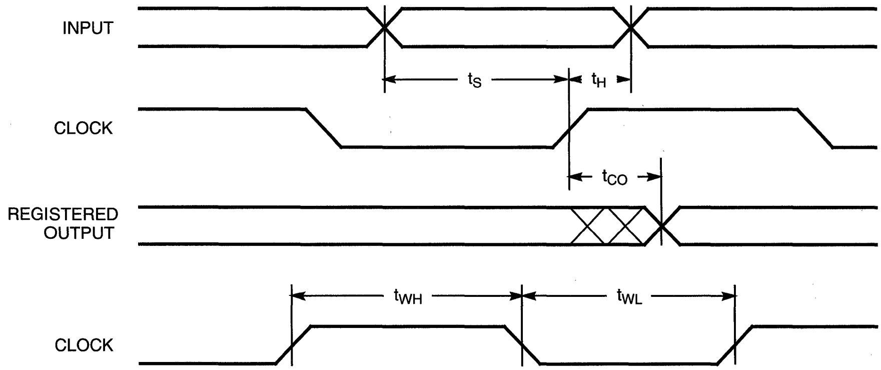
## Switching Waveforms

### Combinatorial Output



7c372i–5

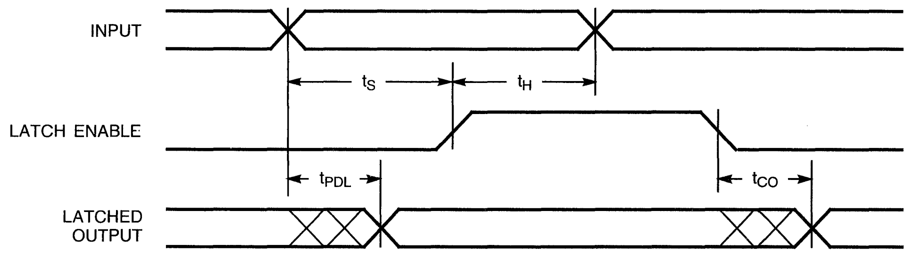
### Registered Output



7c372i–6

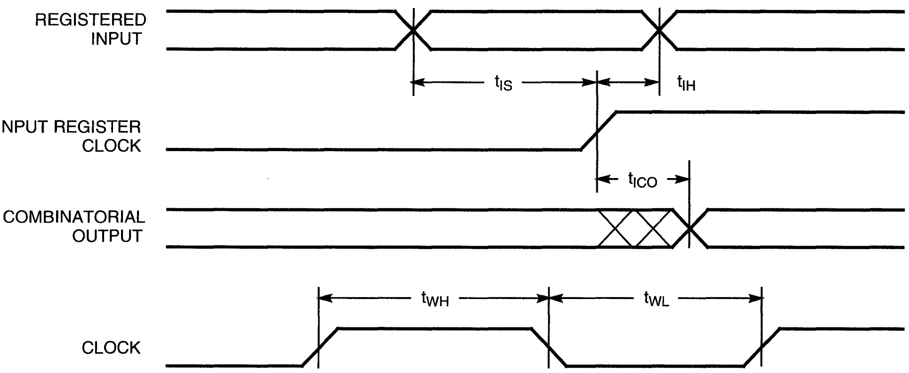
## Switching Waveforms (continued)

### Latched Output



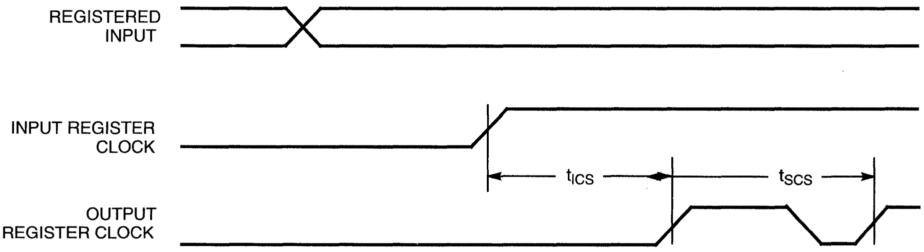
7c372i–7

### Registered Input



7c372i–8

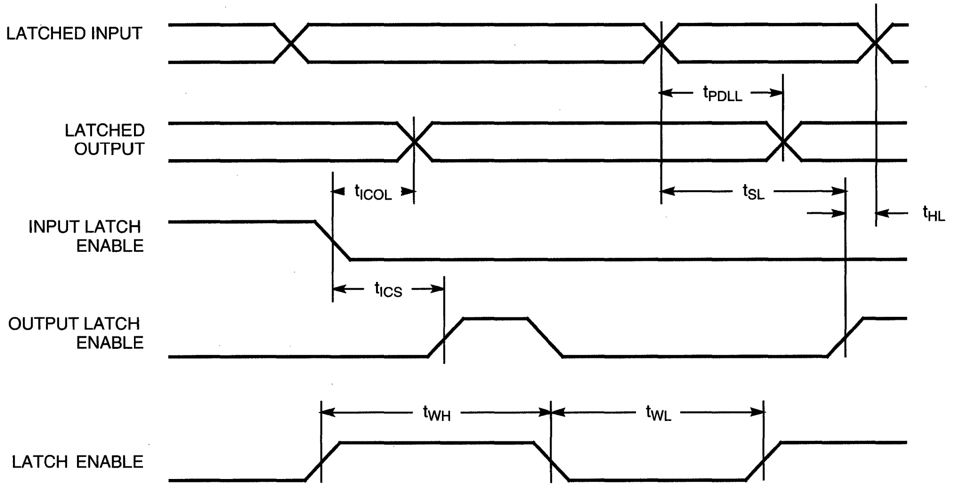
### Clock to Clock



7c372i–9

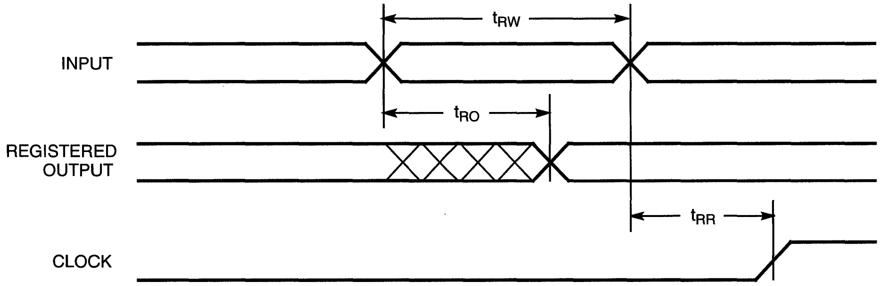
## Switching Waveforms (continued)

### Latched Input and Output



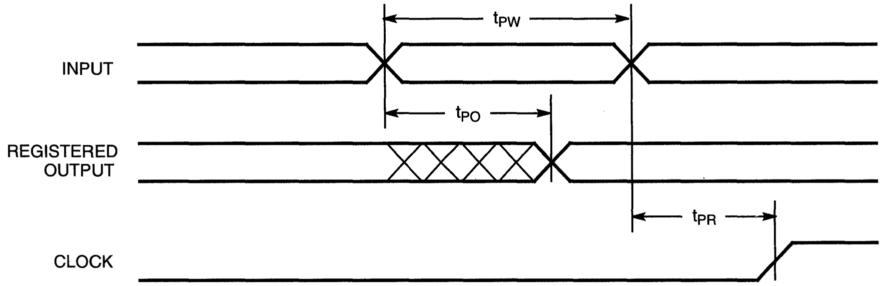
7c372i–11

### Asynchronous Reset



7c372i–12

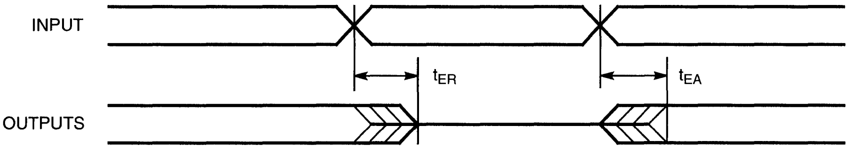
### Asynchronous Preset



7c372i–13

## Switching Waveforms (continued)

### Output Enable/Disable



7c372i–15

## Ordering Information

| Speed (M55IHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 125 | CY7C372i-125JC | J67 | 44-Lead Plastic Leaded Chip Carrier | Commercial |
| 100 | CY7C372i-100JC | J67 | 44-Lead Plastic Leaded Chip Carrier | Commercial |
| 83 | CY7C372i-83JC | J67 | 44-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY7C372i-83JI | J67 | 44-Lead Plastic Leaded Chip Carrier | Industrial |
| | CY7C372i-83YMB | Y67 | 44-Lead Ceramic Leaded Chip Carrier | Military |
| 66 | CY7C372i-66JC | J67 | 44-Lead Plastic Leaded Chip Carrier | Commercial |
| | CY7C372i-66YMB | Y67 | 44-Lead Ceramic Leaded Chip Carrier | Military |
| | CY7C372i-66JI | J67 | 44-Lead Plastic Leaded Chip Carrier | Industrial |
| 66 | CY7C372iL-66JC | J67 | 44-Lead Plastic Leaded Chip Carrier | Commercial |

## MILITARY SPECIFICATIONS Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

Document #: 38-00498-B

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{ICO}$ | 9, 10, 11 |
| $t_S$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |
| $t_{IS}$ | 9, 10, 11 |
| $t_{IH}$ | 9, 10, 11 |
| $t_{ICS}$ | 9, 10, 11 |

ISR, UltraLogic, FLASH370, FLASH370i, and *Warp2*Sim are trademarks of Cypress Semiconductor Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor.
ABEL is a trademark of Data I/O Corporation.
LOG/iC is a trademark of Isdata Corporation.
CUPL is a trademark of Logical Devices Incorporated.

CYPRESS

# CY7C373i

# UltraLogic™ 64-Macrocell Flash CPLD

## Features

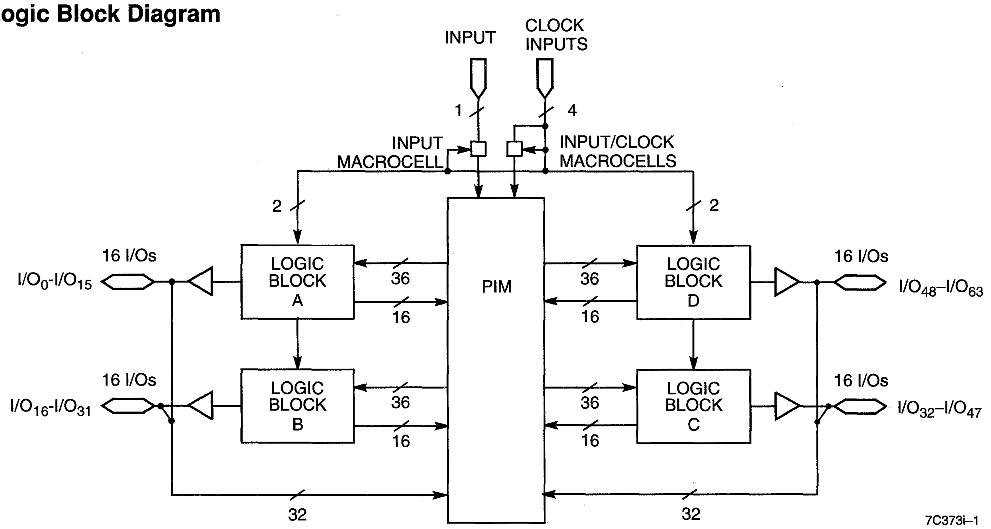
- 64 macrocells in four logic blocks
- 64 I/O pins
- 5 dedicated inputs including 4 clock pins
- In-System Reprogrammable (ISR™) Flash technology
  - —JTAG interface
- Bus Hold capabilities on all I/Os and dedicated inputs
- No hidden delays
- High speed
  - —$f_{MAX}$ = 125 MHz
  - —$t_{PD}$ = 10 ns
  - —$t_S$ = 5.5 ns
  - —$t_{CO}$ = 6.5 ns
- Fully PCI compliant
- Available in 84-pin PLCC and 100-pin TQFP packages
- Pin compatible with the CY7C374i

## Functional Description

The CY7C373i is an In-System Reprogrammable Complex Programmable Logic Device (CPLD) and is part of the FLASH370i™ family of high-density, high-speed CPLDs. Like all members of the FLASH370i family, the CY7C373i is designed to bring the ease of use and high performance of the 22V10, as well as PCI Local Bus Specification support, to high-density CPLDs.

Like all of the UltraLogic FLASH370i devices, the CY7C373i is electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows, thereby reducing costs. The Cypress ISR function is implementedthrough a JTAG serial interface. Data is shifted in and out through the SDI and SDO pins.The ISR interface is enabled using the programming voltage pin ($ISR_{EN}$). Additionally, because of the superior routability of the FLASH370i devices, ISR often allows users to change existing logic designs while simultaneously fixing pinout assignments.

## Logic Block Diagram



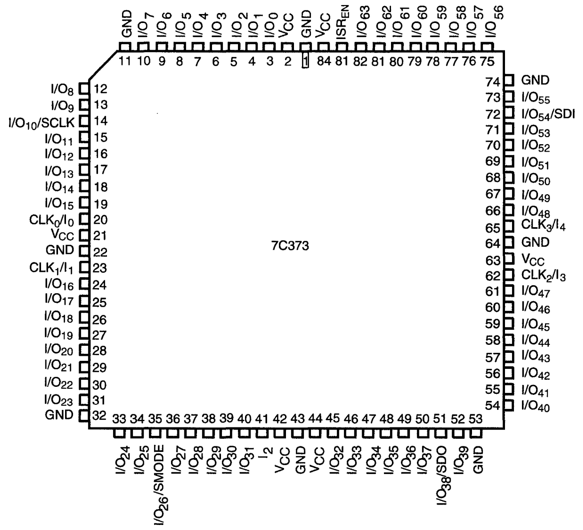
7C373i–1

## Selection Guide

| | 7C373i–125 | 7C373i–100 | 7C373i–83 | 7C373i–66 | 7C373iL–66 |
|---|---|---|---|---|---|
| Maximum Propagation Delay (ns) | 10 | 12 | 15 | 20 | 20 |
| Minimum Set–up, $t_S$ (ns) | 5.5 | 6.0 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 6.5 | 6.5 | 8 | 10 | 10 |
| Typical Supply Current, $I_{CC}$ (mA) | 75 | 75 | 75 | 75 | 45 |

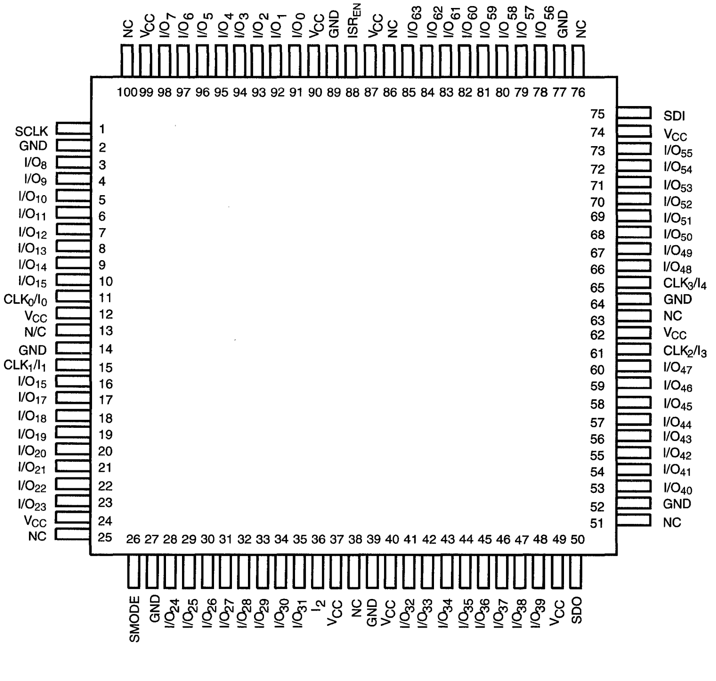
## Pin Configurations

**PLCC**
**Top View**



7C373i–3

**TQFP**
**Top View**



7C373i–2

## Functional Description (continued)

The 64 macrocells in the CY7C373i are divided between four logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370i architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

Like all members of the FLASH370i family, the CY7C373i is rich in I/O resources. Every macrocell in the device features an associated I/O pin, resulting in 64 I/O pins on the CY7C373i. In addition, there is one dedicated input and four input/clock pins.

Finally, the CY7C373i features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used or the type of application, the timing parameters on the CY7C373i remain the same.

### Logic Block

The number of logic blocks distinguishes the members of the FLASH370i family. The CY7C373i includes four logic blocks. Each logic block is constructed of a product term array, a product term allocator, and 16 macrocells.

*Product Term Array*

The product term array in the FLASH370i logic block includes 36 inputs from the PIM and outputs 86 product terms to the product term allocator. The 36 inputs from the PIM are available in both positive and negative polarity, making the overall array size 72 x 86. This large array in each logic block allows for very complex functions to be implemented in single passes through the device.

*Product Term Allocator*

The product term allocator is a dynamic, configurable resource that shifts product term resources to macrocells that require them. Any number of product terms between 0 and 16 inclusive can be assigned to any of the logic block macrocells (this is called product term steering). Furthermore, product terms can be shared among multiple macrocells. This means that product terms that are common to more than one output can be implemented in a single product term. Product term steering and product term sharing help to increase the effective density of the FLASH370i CPLDs. Note that the product term allocator is handled by software and is invisible to the user.

*I/O Macrocell*

Each of the macrocells on the CY7C373i has a separate I/O pin associated with it. In other words, each I/O pin is shared by two macrocells. The input to the macrocell is the sum of between 0 and 16 product terms from the product term allocator. The macrocell includes a register that can be optionally bypassed, polarity control over the input sum-term, and two global clocks to trigger the register. The macrocell also features a separate feedback path to the PIM so that the register can be buried if the I/O pin is used as an input.

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) connects the four logic blocks on the CY7C373i to the inputs and to each other. All inputs (including feedbacks) travel through the PIM. There is no speed penalty incurred by signals traversing the PIM.

### Programming

For an overview of ISR programming, refer to the FLASH370i Family data sheet and for ISR cable and software specifications, refer to ISR data sheets. For a detailed description of ISR capabilities, refer to the Cypress application note, "An Introduction to In System Reprogramming with FLASH370i."

### PCI Compliance

The FLASH370i family of CMOS CPLDs are fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The simple and predictable timing model of FLASH370i ensures compliance with the PCI AC specifications independent of the design. On the other hand, in CPLD and FPGA architectures without simple and predictable timing, PCI compliance is dependent upon routing and product term distribution.

### Bus Hold Capabilities on all I/Os and Dedicated Inputs

In addition to ISR capability, a new feature called bus-hold has been added to all FLASH370i I/Os and dedicated input pins. Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device's performance. As a latch, bus-hold recalls the last state of a pin when it is three-stated, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to $V_{CC}$ or GND.

### Design Tools

Development software for the CY7C373i is available from Cypress's *Warp2*®, *Warp2*Sim™, and *Warp3*® software packages. Both of these products are based on the IEEE standard VHDL language. Cypress also supports third-party vendors such as ABEL™, CUPL™, and LOG/iC™. Please refer to third-party tool support data sheets for further information.

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | −65°C to +150°C |
| Ambient Temperature with Power Applied | −55°C to +125°C |
| Supply Voltage to Ground Potential | −0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | −0.5V to +7.0V |
| DC Input Voltage | −0.5V to +7.0V |
| DC Program Voltage | 12.5V |
| Output Current into Outputs | 16 mA |
| Static Discharge Voltage<br>(per MIL–STD–883, Method 3015) | >2001V |
| Latch–Up Current | >200 mA |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | −40°C to +85°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range

| Parameter | Description | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = −3.2 mA (Com'l/Ind)[1] | 2.4 | | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA (Com'l/Ind)[1] | | | 0.5 | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH Voltage for all Inputs[2] | | 2.0 | | 7.0 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW Voltage for all Inputs[2] | | −0.5 | | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I$ = $V_{CC}$ | | −10 | | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | GND ≤ $V_O$ ≤ $V_{CC}$, Output Disabled | | −50 | | +50 | µA |
| $I_{OS}$ | Output Short Circuit Current[3, 4] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | −30 | | −160 | mA |
| $I_{CC}$ | Power Supply Current[5] | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 mHz, $V_{IN}$ = GND, $V_{CC}$ | Com'l/Ind. | | 75 | 125 | mA |
| | | | Com'l “L”, −66 | | 45 | 75 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | | µA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | −75 | | | µA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | | +500 | µA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | | −500 | µA |

## Capacitance[4]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{IN}$[6] | Input Capacitance | $V_{IN}$ = 5.0V at f=1 MHz | | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 5 | 12 | pF |

**Notes:**

[1] $I_{OH}$ = −2 mA, $I_{OL}$ = 2 mA for SDO.

[2] These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.

[3] Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[4] Tested initially and after any design or process changes that may affect these parameters.

[5] Measured with 16-bit counter programmed into each logic block.

[6] $C_{I/O}$ for dedicated Inputs, and I/Os with JTAG functionality is 12 pF Max., and for $ISR_{EN}$ is 15 pF Max.

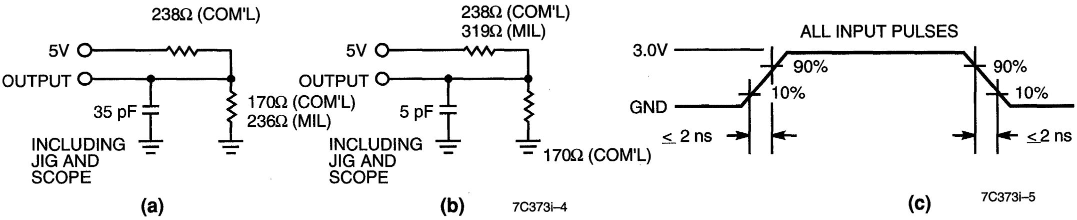
## Inductance[4]

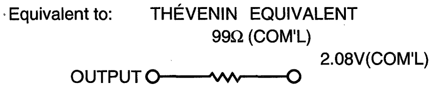
| Parameter | Description | Test Conditions | 100–Pin TQFP | 84–Lead PLCC | Unit |
|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | 8 | nH |

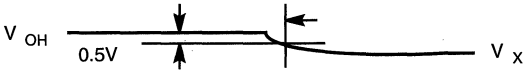
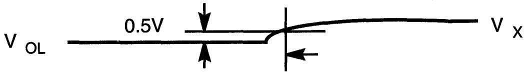
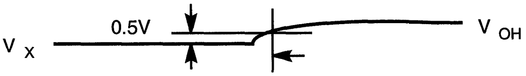
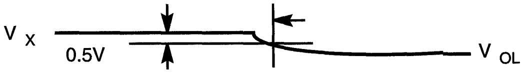
## Endurance Characteristics[4]

| Parameter | Description | Test Conditions | Min. | Unit |
|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | Cycles |

## AC Test Loads and Waveforms





| Parameter[7] | $V_X$ | Output Waveform–Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | 1.5V |  |
| $t_{EA(-)}$ | $V_{the}$ |  |

(d) Test Waveforms

**Note:**

7. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[8]

| | | 7C373i–125 | | 7C373i–100 | | 7C373i–83 | | 7C373i–66<br>7C373iL–66 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| **Combinatorial Mode Parameters** | | | | | | | | | | |
| $t_{PD}$ | Input to Combinatorial Output | | 10 | | 12 | | 15 | | 20 | ns |
| $t_{PDL}$ | Input to Output Through Transparent Input or Output Latch | | 13 | | 15 | | 18 | | 22 | ns |
| $t_{PDLL}$ | Input to Output Through Transparent Input and Output Latches | | 15 | | 16 | | 19 | | 24 | ns |
| $t_{EA}$ | Input to Output Enable | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{ER}$ | Input to Output Disable | | 14 | | 16 | | 19 | | 24 | ns |
| **Input Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[4] | 3 | | 3 | | 4 | | 5 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[4] | 3 | | 3 | | 4 | | 5 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{ICO}$ | Input Register Clock or Latch Enable to Combinatorial Output | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{ICOL}$ | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 16 | | 18 | | 21 | | 26 | ns |
| **Output Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{CO}$ | Clock or Latch Enable to Output | | 6.5 | | 6.5 | | 8 | | 10 | ns |
| $t_S$ | Set-Up Time from Input to Clock or Latch Enable | 5.5 | | 6 | | 8 | | 10 | | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{CO2}$ | Output Clock or Latch Enable to Output Delay (Through Memory Array) | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{SCS}$ | Output Clock or Latch Enable to Output Clock or Latch Enable (Through Memory Array) | 8 | | 10 | | 12 | | 15 | | ns |
| $t_{SL}$ | Set-Up Time from Input Through Transparent Latch to Output Register Clock or Latch Enable | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Clock or Latch Enable | 0 | | 0 | | 0 | | 0 | | ns |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[4] | 125 | | 100 | | 83 | | 66 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$)[4] | 153.8 | | 153.8 | | 125 | | 100 | | MHz |
| $f_{MAX3}$ | Maximum Frequency of (2) CY7C373is with External Feedback (Lesser of $1/(t_{CO} + t_S)$ and $1/(t_{WL} + t_{WH})$)[4] | 83.3 | | 80 | | 62.5 | | 50 | | MHz |
| $t_{OH}$–$t_{IH}$ 37x | Output Data Stable from Output clock Minus Input Register Hold Time for 7C37x[4, 9] | 0 | | 0 | | 0 | | 0 | | ns |

**Notes:**

[8] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

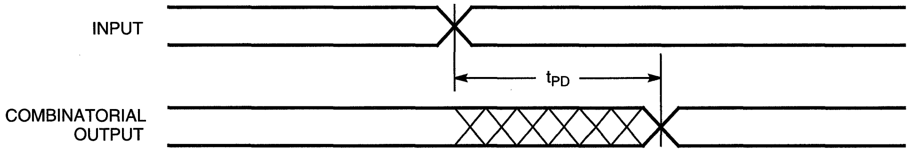
[9] This specification is intended to guarantee interface compatibility of the other members of the CY7C370i family with the CY7C373i. This specification is met for the devices operating at the same ambient temperature and at the same power supply voltage.

## Switching Characteristics Over the Operating Range[8] (continued)

| Parameter | Description | 7C373i–125 | | 7C373i–100 | | 7C373i–83 | | 7C373i–66<br>7C373iL–66 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Pipelined Mode Parameters** | | | | | | | | | | |
| $t_{ICS}$ | Input Register Clock to Output Register Clock | 8 | | 10 | | 12 | | 15 | | ns |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$)[4] | 125 | | 83.3 | | 66.6 | | 50.0 | | MHz |
| **Reset/Preset Parameters** | | | | | | | | | | |
| $t_{RW}$ | Asynchronous Reset Width[4] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[4] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{RO}$ | Asynchronous Reset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| $t_{PW}$ | Asynchronous Preset Width[4] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{PO}$ | Asynchronous Preset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| $t_{POR}$ | Power–On Reset[4] | | 1 | | 1 | | 1 | | 1 | µs |
| **Tap Controller Parameter** | | | | | | | | | | |
| $f_{TAP}$ | Tap Controller Frequency | 500 | | 500 | | 500 | | 500 | | kHz |

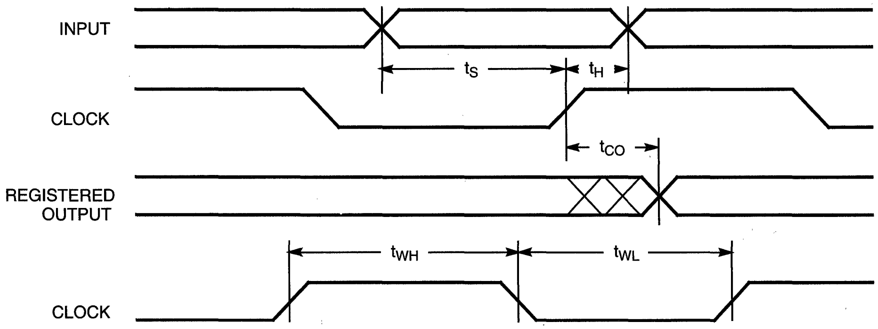
## Switching Waveforms

### Combinatorial Output



7C373i–6

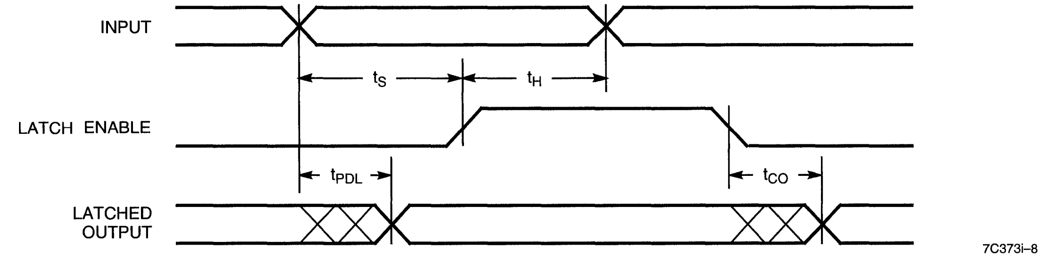
### Registered Output



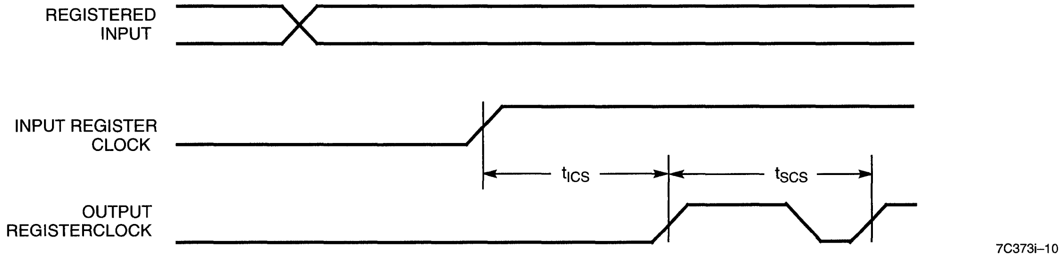
7C373i–7

## Switching Waveforms (continued)

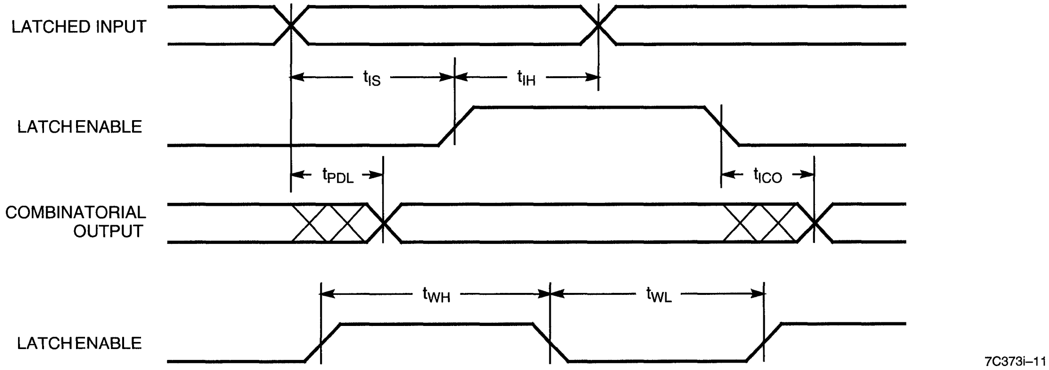
### Latched Output



### Clock to Clock

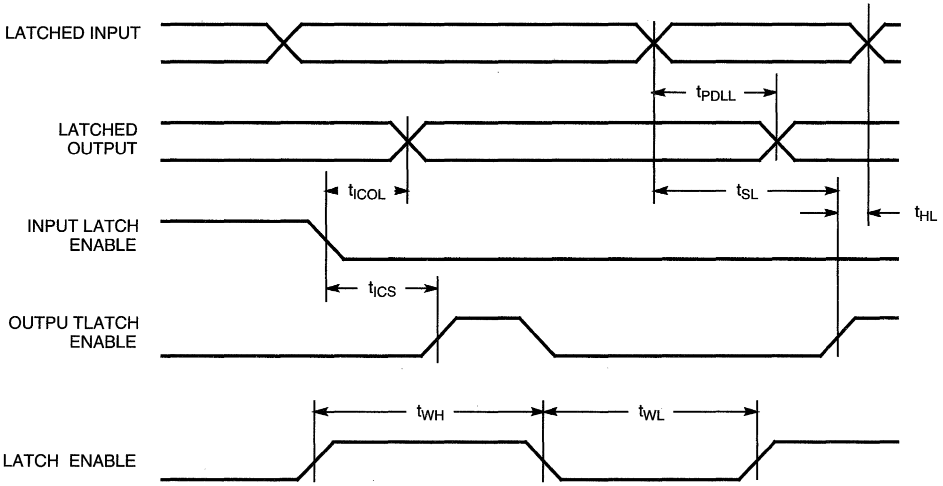


### Latched Input



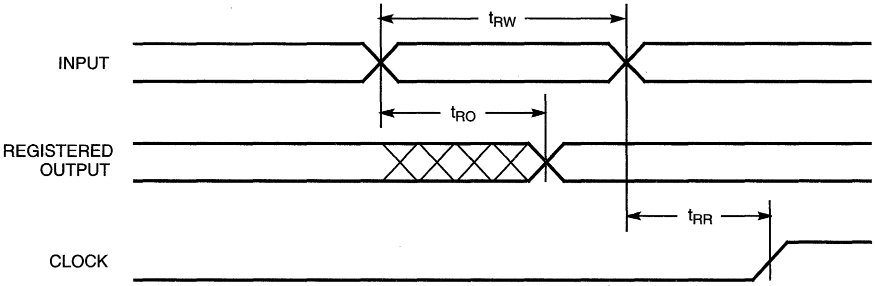
## Switching Waveforms (continued)

### Latched Input and Output



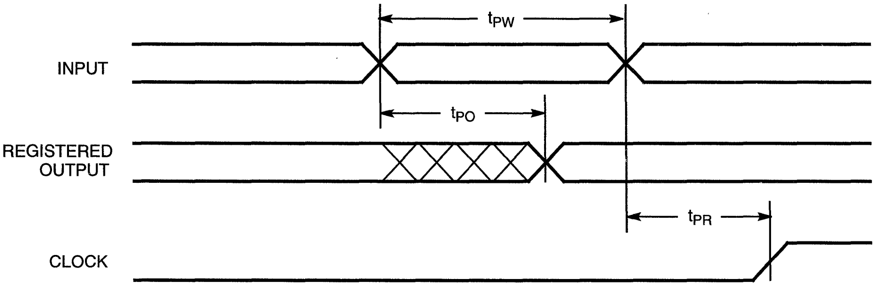
7C373i–12

### Asynchronous Reset



7C373i–13

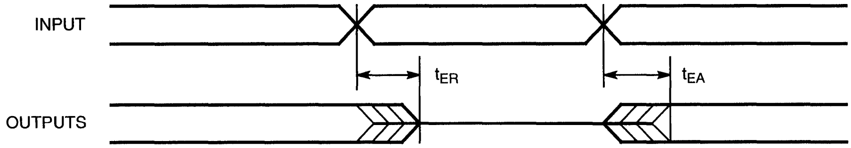
### Asynchronous Preset



7C373i–14

## Switching Waveforms (continued)

### Output Enable/Disable



7C373i–16

## Ordering Information

| Speed (MHz) | Ordering Code | Package Type | Package Type | Operating Range |
|---|---|---|---|---|
| 125 | CY7C373i–125AC | A100 | 100-Pin Thin Quad Flatpack | Commercial |
| | CY7C373i–125JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| 100 | CY7C373i–100AC | A100 | 100-Pin Thin Quad Flatpack | Commercial |
| | CY7C373i–100JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| 83 | CY7C373i–83AC | A100 | 100-Pin Thin Quad Flatpack | Commercial |
| | CY7C373i–83JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C373i–83AI | A100 | 100-Pin Thin Quad Flatpack | Industrial |
| | CY7C373i–83JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| 66 | CY7C373i–66AC | A100 | 100-Pin Thin Quad Flatpack | Commercial |
| | CY7C373i–66JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C373i–66AI | A100 | 100-Pin Thin Quad Flatpack | Industrial |
| | CY7C373i–66JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| 66 | CY7C373iL–66JC | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial |

Document #: 38–00495–B

FLASH370, FLASH370i, ISR, UltraLogic, and *Warp2*Sim are trademarks of Cypress Semiconductor Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor.
ABEL is a trademark of Data I/O Corporation.
LOG/iC is a trademark of Isdata Corporation.
CUPL is a trademark of Logical Devices Incorporated.

CYPRESS

# CY7C374i

# UltraLogic™ 128-Macrocell Flash CPLD

## Features

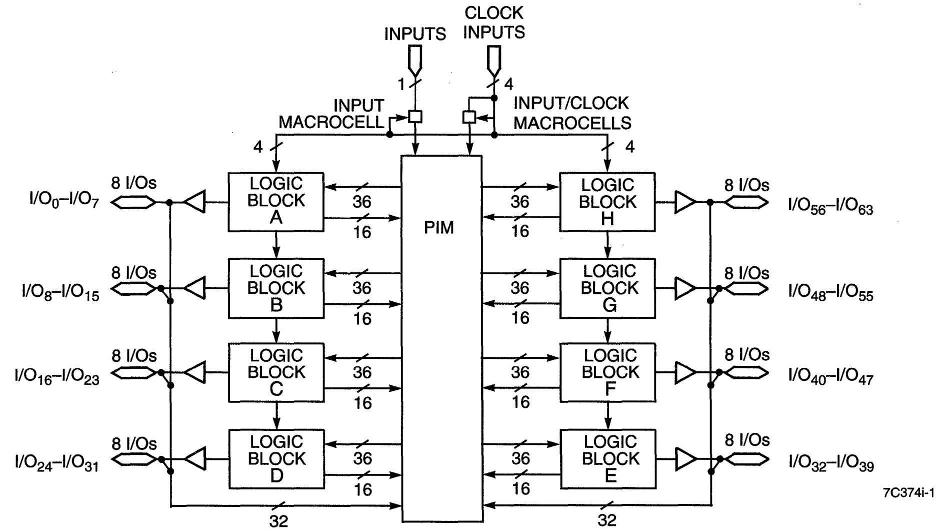
- **128 macrocells in eight logic blocks**
- **64 I/O pins**
- **5 dedicated inputs including 4 clock pins**
- **In-System Reprogrammable (ISR™) Flash technology**
  - **—JTAG interface**
- **Bus Hold capabilities on all I/Os and dedicated inputs**
- **No hidden delays**
- **High speed**
  - **—$f_{MAX}$ = 125 MHz**
  - **—$t_{PD}$ = 10 ns**
  - **—$t_S$ = 5.5 ns**
  - **—$t_{CO}$ = 6.5 ns**
- **Fully PCI compliant**
- **Available in 84-pin PLCC, 84-pin CLCC, and 100-pin TQFP packages**
- **Pin compatible with the CY7C373i**

## Functional Description

The CY7C374i is an In-System Reprogrammable Complex Programmable Logic Device (CPLD) and is part of the FLASH370i™ family of high-density, high-speed CPLDs. Like all members of the FLASH370i family, the CY7C374i is designed to bring the ease of use as well as PCI Local Bus Specification support and high performance of the 22V10 to high-density CPLDs.

Like all of the UltraLogic FLASH370i devices, the CY7C374i is electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows thereby reducing costs. The Cypress ISR function is implemented through a JTAG serial interface. Data is shifted in and out through the SDI and SDO pin. The ISR interface in enabled using the programming voltage pin ($ISR_{EN}$). Additionally, because of the superior routability of the FLASH370i devices, ISR often allows users to change existing logic designs while simultaneously fixing pinout assignments.

## Logic Block Diagram



7C374i-1

## Selection Guide

| | 7C374i-125 | 7C374i–100 | 7C374i–83 | 7C374i–66 | 7C374iL–66 |
|---|---|---|---|---|---|
| Maximum Propagation Delay $t_{PD}$ (ns) | 10 | 12 | 15 | 20 | 20 |
| Minimum Set-Up, $t_S$ (ns) | 5.5 | 6 | 8 | 10 | 10 |
| Maximum Clock to Output, tco (ncs) | 6.5 | 7 | 8 | 10 | 10 |
| Typical Supply Current, $I_{CC}$ (mA) | 125 | 125 | 125 | 125 | 75 |

Shaded ares contain advanced information.

## Functional Description (continued)

The 128 macrocells in the CY7C374i are divided between eight logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370i architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

Like all members of the FLASH370i family, the CY7C374i is rich in I/O resources. Every two macrocells in the device feature an associated I/O pin, resulting in 64 I/O pins on the CY7C374i. In addition, there is one dedicated input and four input/clock pins.

Finally, the CY7C374i features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used or the type of application, the timing parameters on the CY7C374i remain the same.

## Logic Block

The number of logic blocks distinguishes the members of the FLASH370i family. The CY7C374i includes eight logic blocks. Each logic block is constructed of a product term array, a product term allocator, and 16 macrocells.

*Product Term Array*

The product term array in the FLASH370i logic block includes 36 inputs from the PIM and outputs 86 product terms to the product term allocator. The 36 inputs from the PIM are available in both positive and negative polarity, making the overall array size 72 x 86. This large array in each logic block allows for very complex functions to be implemented in single passes through the device.

*Product Term Allocator*

The product term allocator is a dynamic, configurable resource that shifts product terms to macrocells that require them. Any number of product terms between 0 and 16 inclusive can be assigned to any of the logic block macrocells (this is called product term steering). Furthermore, product terms can be shared among multiple macrocells. This means that product terms that are common to more than one output can be implemented in a single product term. Product term steering and product term sharing help to increase the effective density of the FLASH370i CPLDs. Note that product term allocation is handled by software and is invisible to the user.

*I/O Macrocell*

Half of the macrocells on the CY7C374i have I/O pins associated with them. The input to the macrocell is the sum of between 0 and 16 product terms from the product term allocator. The I/O macrocell includes a register that can be optionally bypassed, polarity control over the input sum-term, and two global clocks to trigger the register. The macrocell also features a separate feedback path to the PIM so that the register can be buried if the I/O pin is used as an input.

*Buried Macrocell*

The buried macrocell is very similar to the I/O macrocell. Again, it includes a register that can be configured as combinatorial, as a D flip-flop, a T flip-flop, or a latch. The clock for this register has the same options as described for the I/O macrocell. One difference on the buried macrocell is the addition of input register capability. The user can program the buried macrocell to act as an input register (D-type or latch) whose input comes from the I/O pin associated with the neighboring macrocell. The output of all buried macrocells is sent directly to the PIM regardless of its configuration.

## Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) connects the eight logic blocks on the CY7C374i to the inputs and to each other. All inputs (including feedbacks) travel through the PIM. There is no speed penalty incurred by signals traversing the PIM.

## Programming

For an overview of ISR programming, refer to the FLASH370i Family data sheet and for ISR cable and software specifications, refer to ISR data sheets. For a detailed description of ISR capabilities, refer to the Cypress application note, "An Introduction to In System Reprogramming with FLASH370i."

## PCI Compliance

The FLASH370i family of CMOS CPLDs are fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The simple and predictable timing model of FLASH370i ensures compliance with the PCI AC specifications independent of the design. On the other hand, in CPLD and FPGA architectures without simple and predictable timing, PCI compliance is dependent upon routing and product term distribution.

## Bus Hold Capabilities on all I/Os and Dedicated Inputs

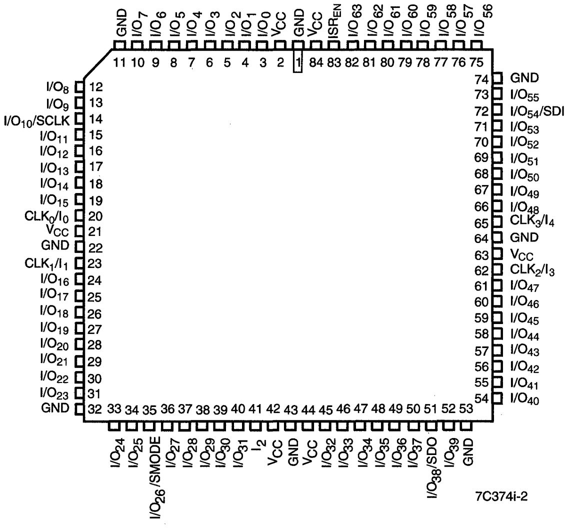
In addition to ISR capability, a new feature called bus-hold has been added to all FLASH370i I/Os and dedicated input pins. Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device's performance. As a latch, bus-hold recalls the last state of a pin when it is three-stated, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to $V_{CC}$ or GND.

## Design Tools

Development software for the CY7C374i is available from Cypress's *Warp2*®, *Warp2*Sim™, and *Warp3*® software packages. Both of these products are based on the IEEE standard VHDL language. Cypress also supports third-party vendors such as ABEL™, CUPL™, and LOG/iC™. Please refer to third-party tool support data sheets for further information.

## Pin Configurations

**PLCC/CLCC**
**Top View**



7C374i-2

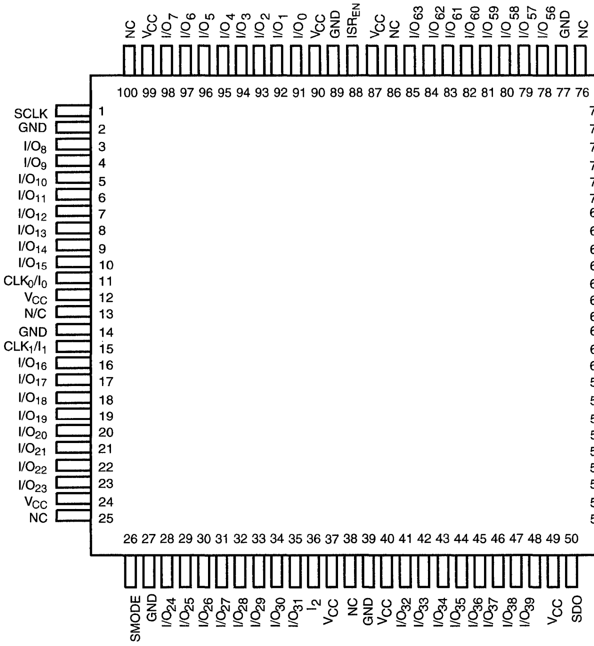
**PGA**
**Bottom View**

|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| L | I/O$_{23}$ | I/O$_{25}$ | I/O$_{26}$ | I/O$_{28}$ | I/O$_{31}$ | I/O$_{33}$ | V$_{CC}$ | I/O$_{34}$ | I/O$_{36}$ | I/O$_{37}$ | I/O$_{39}$ |
| K | I/O$_{21}$ | GND | I/O$_{24}$ | I/O$_{27}$ | I/O$_{30}$ | I$_{2}$ | I/O$_{32}$ | I/O$_{35}$ | I/O$_{38}$ | GND | I/O$_{41}$ |
| J | I/O$_{20}$ | I/O$_{22}$ |  |  | I/O$_{29}$ | V$_{CC}$ | GND |  |  | I/O$_{40}$ | I/O$_{42}$ |
| H | I/O$_{18}$ | I/O$_{19}$ |  |  |  |  |  |  |  | I/O$_{43}$ | I/O$_{44}$ |
| G | CLK1<br>/I$_{1}$ | I/O$_{16}$ | GND |  |  |  |  |  | CLK2<br>/I$_{3}$ | I/O$_{46}$ | I/O$_{47}$ |
| F | I/O$_{17}$ | CLK0<br>/I$_{0}$ | V$_{CC}$ |  |  |  |  |  | V$_{CC}$ | I/O$_{45}$ | GND |
| E | I/O$_{15}$ | I/O$_{14}$ | I/O$_{13}$ |  |  |  |  |  | I/O$_{49}$ | I/O$_{48}$ | CLK3<br>/I$_{4}$ |
| D | I/O$_{12}$ | I/O$_{11}$ |  |  |  |  |  |  |  | I/O$_{51}$ | I/O$_{50}$ |
| C | I/O$_{10}$ | I/O$_{8}$ | ■ |  | I/O$_{1}$ | V$_{CC}$ | ISR$_{EN}$ |  |  | I/O$_{54}$ | I/O$_{52}$ |
| B | I/O$_{9}$ | GND | I/O$_{6}$ | I/O$_{3}$ | I/O$_{0}$ | I/O$_{61}$ | I/O$_{62}$ | I/O$_{59}$ | I/O$_{56}$ | GND | I/O$_{53}$ |
| A | I/O$_{7}$ | I/O$_{5}$ | I/O$_{4}$ | I/O$_{2}$ | V$_{CC}$ | GND | I/O$_{63}$ | I/O$_{60}$ | I/O$_{58}$ | I/O$_{57}$ | I/O$_{55}$ |

7C374i–3

## Pin Configurations

TQFP
Top View



7C374i-4

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................−65°C to +150°C

Ambient Temperature with
Power Applied ..........................................−55°C to +125°C

Supply Voltage to Ground Potential ...............−0.5V to +7.0V

DC Voltage Applied to Outputs
in High Z State .............................................−0.5V to +7.0V

DC Input Voltage ..........................................−0.5V to +7.0V

DC Program Voltage ....................................................12.5V

Output Current into Outputs ........................................16 mA

Static Discharge Voltage............................................>2001V
(per MIL-STD-883, Method 3015)

Latch-Up Current ....................................................>200 mA

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | −40°C to +85°C | 5V ± 5% |
| Military[1] | −55°C to +125°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = −3.2 mA (Com'l/Ind)[3] | 2.4 | | | V |
| | | | $I_{OH}$ = −2.0 mA (Mil) | | | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA (Com'l/Ind)[3] | | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA (Mil) | | | | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH voltage for all inputs[4] | | 2.0 | | 7.0 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW voltage for all inputs[4] | | −0.5 | | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I = V_{CC}$ | | −10 | | +10 | μA |
| $I_{OZ}$ | Output Leakage Current | GND ≤ $V_O$ ≤ $V_{CC}$, Output Disabled | | −50 | | +50 | μA |
| $I_{OS}$ | Output Short Circuit Current[5, 6] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | −30 | | −160 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 MHz, $V_{IN}$ = GND, $V_{CC}$[7] | Com'l/Ind. | | 125 | 200 | mA |
| | | | Com'l "L" −66 | | 75 | 125 | mA |
| | | | Military | | 125 | 250 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | | μA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | −75 | | | μA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | | +500 | μA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | | −500 | μA |

## Capacitance[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{I/O}$[8, 9] | Input Capacitance | $V_{IN}$ = 5.0V at f=1 MHz | | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 5 | 12 | pF |

**Notes:**

1. $T_A$ is the "instant on" case temperature.
2. See the last page of this specification for Group A subgroup testing information.
3. $I_{OH}$ = −2 mA, $I_{OL}$ = 2 mA for SDO.
4. These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.
5. Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
6. Tested initially and after any design or process changes that may affect these parameters.
7. Measured with 16-bit counter programmed into each logic block.
8. $C_{I/O}$ for the CLCC package are 12 pF Max
9. $C_{I/O}$ for dedicated Inputs, and for I/O pins with JTAG functionality is 12 pF Max., and for $ISR_{EN}$ is 15 pF Max.

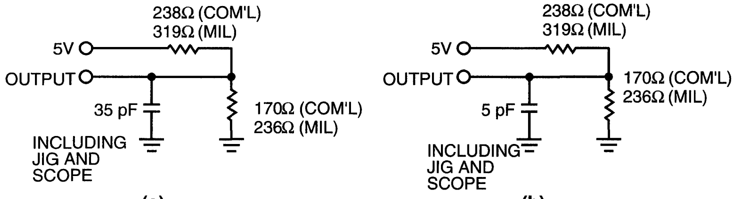
## Inductance[6]

| Parameter | Description | Test Conditions | 100-PinTQFP | 84-Lead PLCC | 84-Lead CLCC | Unit |
|---|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at f = 1 MHz | 8 | 8 | 5 | nH |

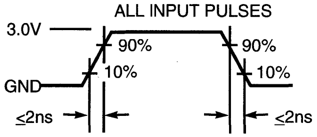
## Endurance Characteristics[6]

| Parameter | Description | Test Conditions | Min. | Unit |
|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | Cycles |

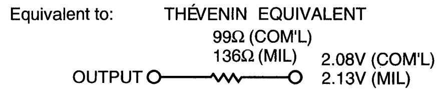
## AC Test Loads and Waveforms

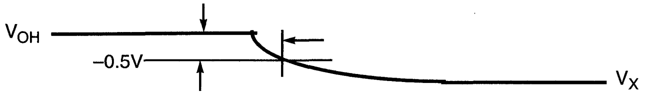
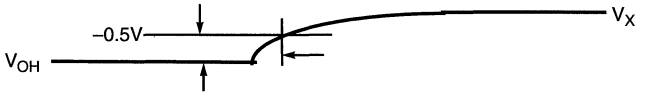
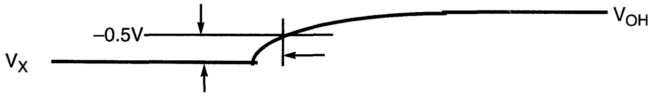
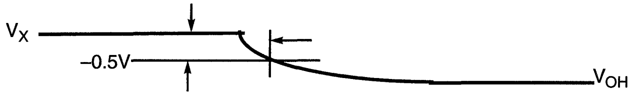


(a) (b) 7C374i-5



(c) 7C374i-6



| Parameter[10] | $V_X$ | Output Waveform Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  |
| $t_{ER(+)}$ | 2.6V |  |
| $t_{EA(+)}$ | 1.5V |  |
| $t_{EA(-)}$ | $V_{thc}$ |  |

**Note:**

10. $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[11]

| Parameter | Description | 7C374i-125 |  | 7C374i–100 |  | 7C374i–83 |  | 7C374i–66<br>7C374iL–66 |  | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. |  |
| **Combinatorial Mode Parameters** |  |  |  |  |  |  |  |  |  |  |
| $t_{PD}$ | Input to Combinatorial Output |  | 10 |  | 12 |  | 15 |  | 20 | ns |
| $t_{PDL}$ | Input to Output Through Transparent Input or Output Latch |  | 13 |  | 15 |  | 18 |  | 22 | ns |
| $t_{PDLL}$ | Input to Output Through Transparent Input and Output Latches |  | 15 |  | 16 |  | 19 |  | 24 | ns |
| $t_{EA}$ | Input to Output Enable |  | 14 |  | 16 |  | 19 |  | 24 | ns |
| $t_{ER}$ | Input to Output Disable |  | 14 |  | 16 |  | 19 |  | 24 | ns |
| **Input Registered/Latched Mode Parameters** |  |  |  |  |  |  |  |  |  |  |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[6] | 3 |  | 3 |  | 4 |  | 5 |  | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[6] | 3 |  | 3 |  | 4 |  | 5 |  | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 |  | 2 |  | 3 |  | 4 |  | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 |  | 2 |  | 3 |  | 4 |  | ns |
| $t_{ICO}$ | Input Register Clock or Latch Enable to Combinatorial Output |  | 14 |  | 16 |  | 19 |  | 24 | ns |
| $t_{ICOL}$ | Input Register Clock or Latch Enable to Output Through Transparent Output Latch |  | 16 |  | 18 |  | 21 |  | 26 | ns |
| **Output Registered/Latched Mode Parameters** |  |  |  |  |  |  |  |  |  |  |
| $t_{CO}$ | Clock or Latch Enable to Output |  | 6.5 |  | 7 |  | 8 |  | 10 | ns |
| $t_S$ | Set-Up Time from Input to Clock or Latch Enable | 5.5 |  | 6 |  | 8 |  | 10 |  | ns |
| $t_H$ | Register or Latch Data Hold Time | 0 |  | 0 |  | 0 |  | 0 |  | ns |
| $t_{CO2}$ | Output Clock or Latch Enable to Output Delay (Through Memory Array) |  | 14 |  | 16 |  | 19 |  | 24 | ns |
| $t_{SCS}$ | Output Clock or Latch Enable to Output Clock or Latch Enable (Through Memory Array) | 8 |  | 10 |  | 12 |  | 15 |  | ns |
| $t_{SL}$ | Set-Up Time from Input Through Transparent Latch to Output Register Clock or Latch Enable | 10 |  | 12 |  | 15 |  | 20 |  | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Clock or Latch Enable | 0 |  | 0 |  | 0 |  | 0 |  | ns |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO}$)[6] | 125 |  | 100 |  | 83 |  | 66 |  | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO}$) | 158.3 |  | 143 |  | 125 |  | 100 |  | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ and $1/(t_{WL} + t_{WH})$) | 83.3 |  | 76.9 |  | 67.5 |  | 50 |  | MHz |
| $t_{OH}–t_{IH}$<br>37x | Output Data Stable from Output Clock Minus Input Register Hold Time for 7C37x[6, 12] | 0 |  | 0 |  | 0 |  | 0 |  | ns |
| **Pipelined Mode Parameters** |  |  |  |  |  |  |  |  |  |  |
| $t_{ICS}$ | Input Register Clock to Output Register Clock | 8 |  | 10 |  | 12 |  | 15 |  | ns |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 125 |  | 100 |  | 83.3 |  | 66.6 |  | MHz |

Shaded ares contain advanced information.

## Switching Characteristics Over the Operating Range[11] (continued)

| Parameter | Description | 7C374i-125 | | 7C374i–100 | | 7C374i–83 | | 7C374i–66<br>7C374iL–66 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Reset/Preset Parameters** | | | | | | | | | | |
| $t_{RW}$ | Asynchronous Reset Width[6] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[6] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{RO}$ | Asynchronous Reset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| $t_{PW}$ | Asynchronous Preset Width[6] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[6] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{PO}$ | Asynchronous Preset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| **Tap Controller Parameter** | | | | | | | | | | |
| $f_{TAP}$ | Tap Controller Frequency | 500 | | 500 | | 500 | | 500 | | kHz |

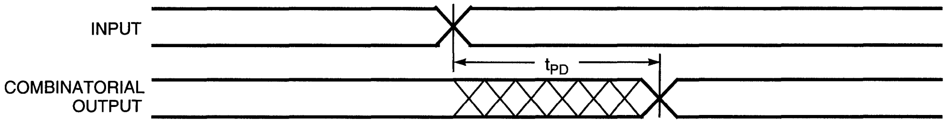
Shaded ares contain advanced information.

**Notes:**

11. All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.
12. This specification is intended to guarantee interface compatibility of the other members of the CY7C370i family with the CY7C374i. This specification is met for the devices operating at the same ambient temperature and at the same power supply voltage.

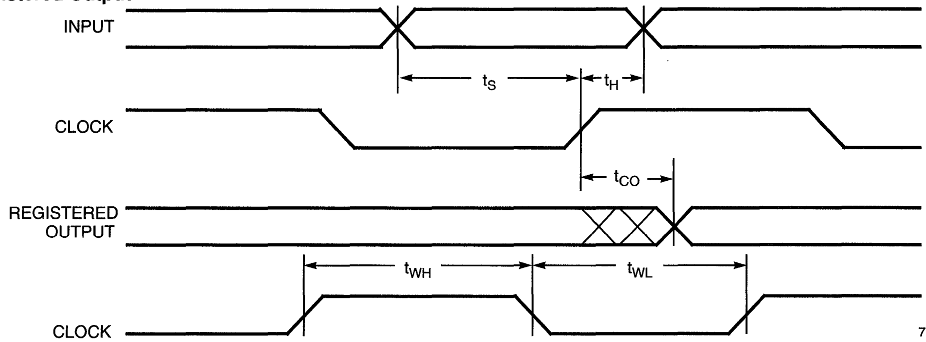
## Switching Waveforms

### Combinatorial Output



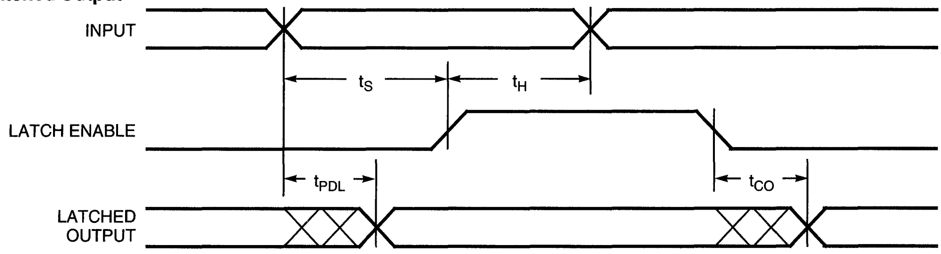
7C374i-7

### Registered Output



7C374i-8

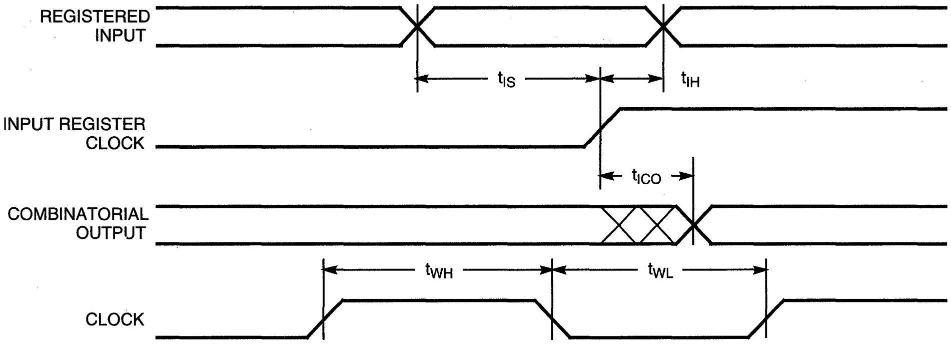
### Latched Output



7C374i-9

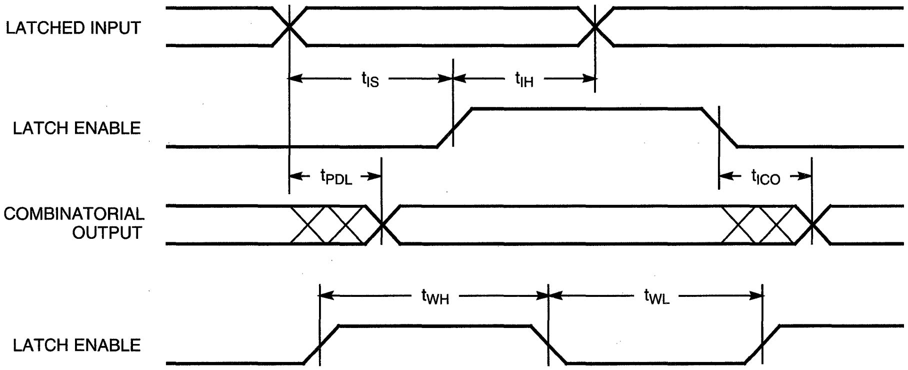
## Switching Waveforms (continued)

### Registered Input



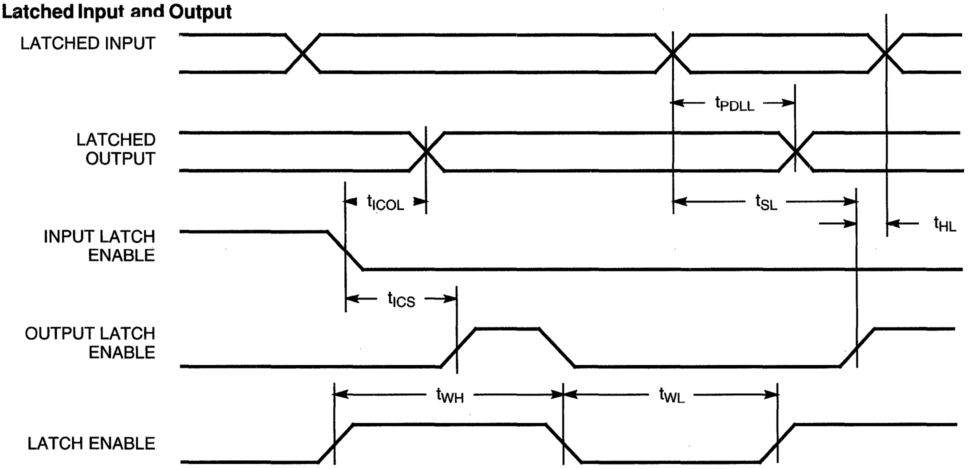
7C374i-10

### Latched Input



7C374i-11

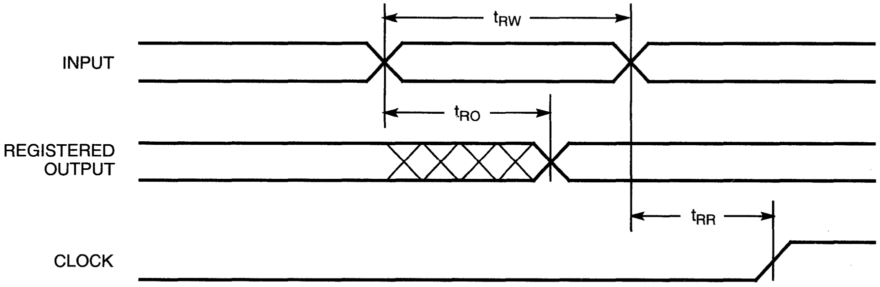
### Latched Input and Output



7C374i-12

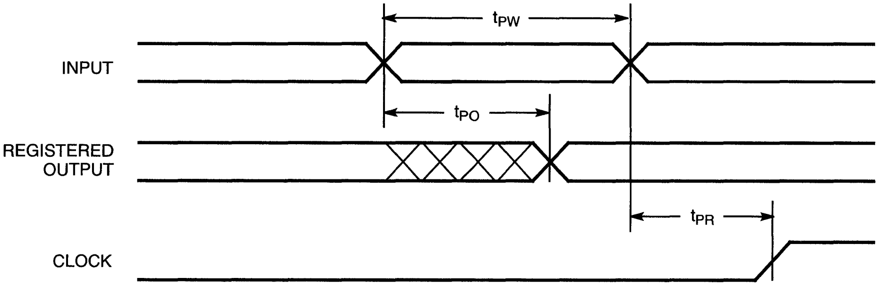
## Switching Waveforms (continued)

### Asynchronous Reset



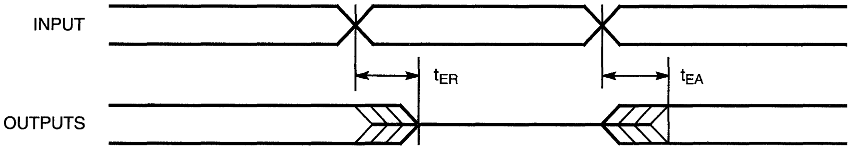
7C374i-13

### Asynchronous Preset



7C374i-14

### Output Enable/Disable



7C374i-16

## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 125 | CY7C374i–125AC | A100 | 100-Pin Thin Quad Flat Pack | Commercial |
| | CY7C374i–125JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| 100 | CY7C374i–100AC | A100 | 100-Pin Thin Quad Flat Pack | Commercial |
| | CY7C374i–100JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| 83 | CY7C374i–83AC | A100 | 100-Pin Thin Quad Flat Pack | Commercial |
| | CY7C374i–83JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C374i–83AI | A100 | 100-Pin Thin Quad Flat Pack | Industrial |
| | CY7C374i–83JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C374i-83GMB | G84 | 84-Pin Ceramic Pin Grid Array | Military |
| | CY7C374i–83YMB | Y84 | 84-Pin Ceramic Leaded Chip Carrier | |
| 66 | CY7C374i–66AC | A100 | 100-Pin Thin Quad Flat Pack | Commercial |
| | CY7C374i–66JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C374i–66AI | A100 | 100-Pin Thin Quad Flat Pack | Industrial |
| | CY7C374i–66JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C374i-66GMB | G84 | 84-Pin Ceramic Pin Grid Array | Military |
| | CY7C374i–66YMB | Y84 | 84-Pin Ceramic Leaded Chip Carrier | |
| | CY7C374iL–66AC | A100 | 100-Pin Thin Quad Flat Pack | Commercial |
| | CY7C374iL–66JC | J83 | 84-Lead Plastic Leaded Chip Carrier | |

Shaded areas contain preliminary information.

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{PDL}$ | 9, 10, 11 |
| $t_{PDLL}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{ICO}$ | 9, 10, 11 |
| $t_{ICOL}$ | 9, 10, 11 |
| $t_{S}$ | 9, 10, 11 |
| $t_{SL}$ | 9, 10, 11 |
| $t_{H}$ | 9, 10, 11 |
| $t_{HL}$ | 9, 10, 11 |
| $t_{IS}$ | 9, 10, 11 |
| $t_{IH}$ | 9, 10, 11 |
| $t_{ICS}$ | 9, 10, 11 |
| $t_{EA}$ | 9, 10, 11 |
| $t_{ER}$ | 9, 10, 11 |

Document #: 38-00496-B

CYPRESS

# CY7C375i

# UltraLogic™ 128-Macrocell Flash CPLD

## Features

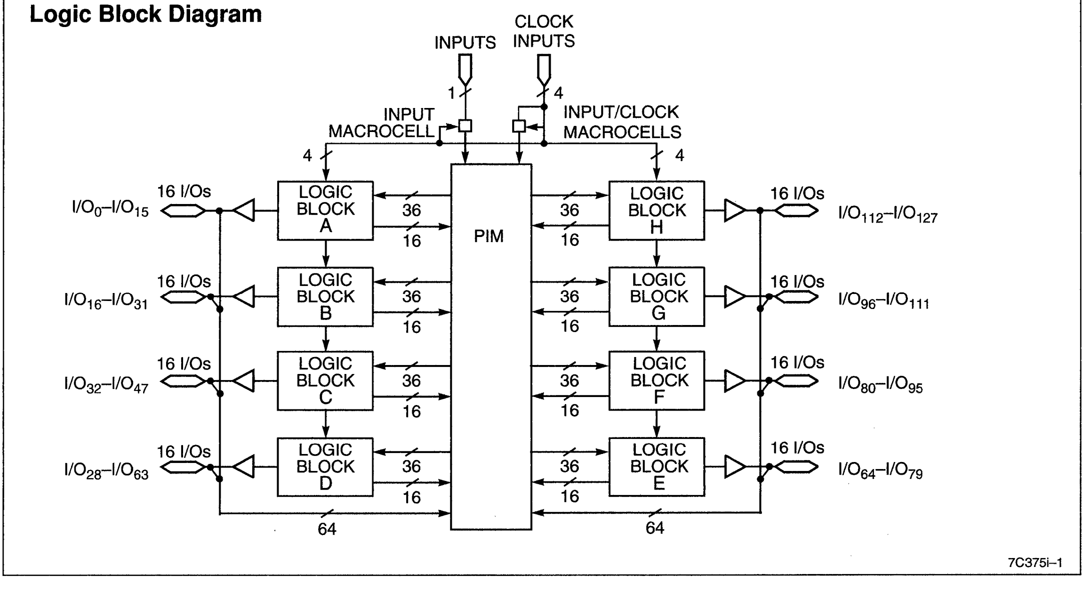
- **128 macrocells in eight logic blocks**
- **128 I/O pins**
- **5 dedicated inputs including 4 clock pins**
- **In-System Reprogrammable (ISR™) Flash technology**
- **JTAG interface**
- **Bus Hold capabilities on all I/Os and dedicated inputs**
- **No hidden delays**
- **High speed**
  - **—$f_{MAX}$ = 125 MHz**
  - **—$t_{PD}$ = 10 ns**
  - **—$t_S$ = 5.5 ns**
  - **—$t_{CO}$ = 6.5 ns**
- **Fully PCI compliant**
- **Available in 160-pin TQFP, CQFP, and PGA packages**

## Functional Description

The CY7C375i is an In-System Reprogrammable Complex Programmable Logic Device (CPLD) and is part of the FLASH370i™ family of high-density, high-speed CPLDs. Like all members of the FLASH370i family, the CY7C375i is designed to bring the ease of use and high performance of the 22V10 to high-density PLDs.

Like all of the UltraLogic FLASH370i devices, the CY7C375i is electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows thereby reducing costs. The Cypress ISR function is implemented through a JTAG serial interface. Data is shifted in and out through the SDI and SDO pins. The ISR interface is enabled using the programming voltage pin ($ISR_{EN}$). Additionally, because of the superior routability of the FLASH370i devices, ISR often allows users to change existing logic designs while simultaneously fixing pinout assignments.

## Logic Block Diagram



7C375i–1

## Selection Guide

| | 7C375i-125 | 7C375i–100 | 7C375i–83 | 7C375i–66 | 7C375iL–66 |
|---|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | 10 | 12 | 15 | 20 | 20 |
| Minimum Set-Up, $t_S$ (ns) | 5.5 | 6 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | 6.5 | 7 | 8 | 10 | 10 |
| Typical Supply Current, $I_{CC}$ (mA) | 125 | 125 | 125 | 125 | 75 |

Shaded areas contain preliminary information.

## Functional Description (continued)

The 128 macrocells in the CY7C375i are divided between eight logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370i architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

Like all members of the FLASH370i family, the CY7C375i is rich in I/O resources. Every macrocell in the device features an associated I/O pin, resulting in 128 I/O pins on the CY7C375i. In addition, there is one dedicated input and four input/clock pins.

Finally, the CY7C375i features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used or the type of application, the timing parameters on the CY7C375i remain the same.

### Logic Block

The number of logic blocks distinguishes the members of the FLASH370i family. The CY7C375i includes eight logic blocks. Each logic block is constructed of a product term array, a product term allocator, and 16 macrocells.

*Product Term Array*

The product term array in the FLASH370i logic block includes 36 inputs from the PIM and outputs 86 product terms to the product term allocator. The 36 inputs from the PIM are available in both positive and negative polarity, making the overall array size 72 x 86. This large array in each logic block allows for very complex functions to be implemented in single passes through the device.

*Product Term Allocator*

The product term allocator is a dynamic, configurable resource that shifts product terms to macrocells that require them. Any number of product terms between 0 and 16 inclusive can be assigned to any of the logic block macrocells (this is called product term steering). Furthermore, product terms can be shared among multiple macrocells. This means that product terms that are common to more than one output can be implemented in a single product term. Product term steering and product term sharing help to increase the effective density of the FLASH370i PLDs. Note that product term allocation is handled by software and is invisible to the user.

*I/O Macrocell*

Each of the macrocells on the CY7C375i has a separate I/O pin associated with it. The input to the macrocell is the sum of between 0 and 16 product terms from the product term allocator. The macrocell includes a register that can be optionally bypassed, polarity control over the input sum-term, and four global clocks to trigger the register. The macrocell also features a separate feedback path to the PIM so that the register can be buried if the I/O pin is used as an input.

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) connects the eight logic blocks on the CY7C375i to the inputs and to each other. All inputs (including feedbacks) travel through the PIM. There is no speed penalty incurred by signals traversing the PIM.

### Programming

For an overview of ISR programming, refer to the FLASH370i Family data sheet and for ISR cable and software specifications, refer to ISR data sheets. For a detailed description of ISR capabilities, refer to the Cypress application note, “An Introduction to In System Reprogramming with FLASH370i.”

### PCI Compliance

The FLASH370i family of CMOS CPLDs are fully compliant with the PCI Local Bus Specification published by the PCI Special Interest Group. The simple and predictable timing model of FLASH370i ensures compliance with the PCI AC specifications independent of the design. On the other hand, in CPLD and FPGA architectures without simple and predictable timing, PCI compliance is dependent upon routing and product term distribution.

### Bus Hold Capabilities on all I/Os and Dedicated Inputs

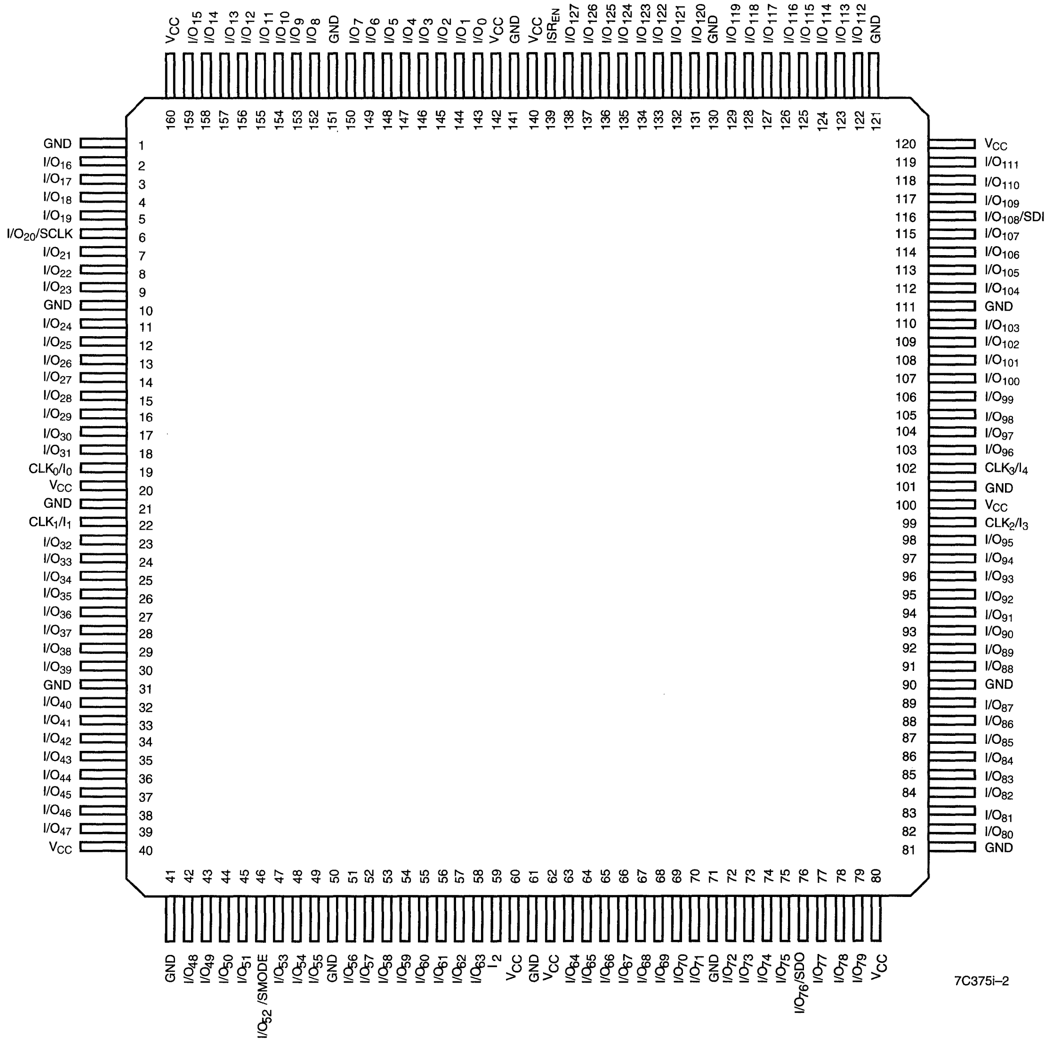
In addition to ISR capability, a new feature called bus-hold has been added to all FLASH370i I/Os and dedicated input pins. Bus-hold, which is an improved version of the popular internal pull-up resistor, is a weak latch connected to the pin that does not degrade the device’s performance. As a latch, bus-hold recalls the last state of a pin when it is three-stated, thus reducing system noise in bus-interface applications. Bus-hold additionally allows unused device pins to remain unconnected on the board, which is particularly useful during prototyping as designers can route new signals to the device without cutting trace connections to VCC or GND.

### Design Tools

Development software for the CY7C375i is available from all of Cypress’s *Warp*™ software packages. Both of these products are based on the IEEE standard VHDL language. Cypress also supports third-party vendors such as ABEL™, CUPL™, and LOG/iC™. Please refer to third-party tool support data sheets for further information.

## Pin Configurations

**Top View**
**TQFP/CQFP**



7C375i–2

## Pin Configurations (continued)

**PGA**
**BottomView**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | I/O$_{109}$ | I/O$_{112}$ | I/O$_{115}$ | I/O$_{118}$ | I/O$_{121}$ | I/O$_{123}$ | I/O$_{126}$ | I/O$_{127}$ | I/O$_{0}$ | I/O$_{3}$ | I/O$_{5}$ | I/O$_{7}$ | I/O$_{10}$ | I/O$_{11}$ | I/O$_{14}$ |
| P | I/O$_{106}$ | I/O$_{110}$ | I/O$_{113}$ | I/O$_{116}$ | I/O$_{119}$ | I/O$_{122}$ | I/O$_{125}$ | GND | I/O$_{1}$ | I/O$_{4}$ | I/O$_{6}$ | I/O$_{9}$ | I/O$_{13}$ | I/O$_{15}$ | I/O$_{16}$ |
| N | I/O$_{105}$ | I/O$_{108}$<br>/SDI | I/O$_{111}$ | I/O$_{114}$ | I/O$_{117}$ | I/O$_{120}$ | I/O$_{124}$ | ISR$_{EN}$ | I/O$_{2}$ | GND | I/O$_{8}$ | I/O$_{12}$ | GND | I/O$_{17}$ | I/O$_{19}$ |
| M | I/O$_{102}$ | I/O$_{104}$ | I/O$_{107}$ | V$_{CC}$ | | | V$_{CC}$ | GND | V$_{CC}$ | | | GND | I/O$_{18}$ | I/O$_{20}$<br>/SCLK | I/O$_{22}$ |
| L | I/O$_{100}$ | I/O$_{101}$ | I/O$_{103}$ | | | | | | | | | | I/O$_{21}$ | I/O$_{23}$ | I/O$_{25}$ |
| K | I/O$_{98}$ | I/O$_{99}$ | GND | | | | | | | | | | I/O$_{24}$ | I/O$_{26}$ | I/O$_{27}$ |
| J | I/O$_{96}$ | I/O$_{97}$ | CLK$_{3}$<br>/I$_{4}$ | V$_{CC}$ | | | | | | | | V$_{CC}$ | CLK$_{28}$ | I/O$_{29}$ | I/O$_{30}$ |
| H | I/O$_{95}$ | GND | CLK$_{2}$<br>/I$_{3}$ | GND | | | | | | | | GND | CLK$_{0}$<br>/I$_{0}$ | GND | I/O$_{31}$ |
| G | I/O$_{94}$ | I/O$_{93}$ | I/O$_{92}$ | V$_{CC}$ | | | | | | | | V$_{CC}$ | CLK1<br>/I1 | I/O$_{33}$ | I/O$_{32}$ |
| F | I/O$_{91}$ | I/O$_{90}$ | I/O$_{88}$ | | | | | | | | | | GND | I/O$_{35}$ | I/O$_{34}$ |
| E | I/O$_{89}$ | I/O$_{87}$ | I/O$_{85}$ | | ■ | | | | | | | | I/O$_{39}$ | I/O$_{37}$ | I/O$_{36}$ |
| D | I/O$_{86}$ | I/O$_{84}$ | I/O$_{82}$ | GND | | | V$_{CC}$ | GND | V$_{CC}$ | | | V$_{CC}$ | I/O$_{43}$ | I/O$_{40}$ | I/O$_{38}$ |
| C | I/O$_{83}$ | I/O$_{81}$ | GND | I/O$_{76}$<br>/SDO | I/O$_{72}$ | GND | I/O$_{66}$ | I$_{2}$ | I/O$_{60}$ | I/O$_{56}$ | I/O$_{53}$ | I/O$_{50}$ | I/O$_{47}$ | I/O$_{44}$ | I/O$_{41}$ |
| B | I/O$_{80}$ | I/O$_{79}$ | I/O$_{77}$ | I/O$_{73}$ | I/O$_{70}$ | I/O$_{68}$ | I/O$_{65}$ | GND | I/O$_{61}$ | I/O$_{58}$ | I/O$_{55}$ | I/O$_{52}$/<br>SMODE | I/O$_{49}$ | I/O$_{46}$ | I/O$_{42}$ |
| A | I/O$_{78}$ | I/O$_{75}$ | I/O$_{74}$ | I/O$_{71}$ | I/O$_{69}$ | I/O$_{67}$ | I/O$_{64}$ | I/O$_{63}$ | I/O$_{62}$ | I/O$_{59}$ | I/O$_{57}$ | I/O$_{71}$ | I/O$_{51}$ | I/O$_{48}$ | I/O$_{45}$ |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

7C375i–3

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | −65°C to +150°C |
| Ambient Temperature with Power Applied | −55°C to +125°C |
| Supply Voltage to Ground Potential | −0.5V to +7.0V |
| DC Voltage Applied to Outputs in High Z State | −0.5V to +7.0V |
| DC Input Voltage | −0.5V to +7.0V |
| DC Program Voltage | 12.5V |
| Output Current into Outputs | 16 mA |
| Static Discharge Voltage (per MIL-STD-883, Method 3015) | >2001V |
| Latch-Up Current | >200 mA |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | −40°C to +85°C | 5V ± 10% |
| Military[1] | −55°C to +125°C | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min. | $I_{OH}$ = −3.2 mA (Com'l/Ind)[3] | 2.4 | | | V |
| | | | $I_{OH}$ = −2.0 mA (Mil) | | | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min. | $I_{OL}$ = 16 mA (Com'l/Ind)[3] | | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA (Mil) | | | | V |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Input Logical HIGH voltage for all inputs[4] | | 2.0 | | 7.0 | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Input Logical LOW voltage for all inputs[4] | | −0.5 | | 0.8 | V |
| $I_{IX}$ | Input Load Current | $V_I$ = Internal GND, $V_I = V_{CC}$ | | −10 | | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | GND ≤ $V_O$ ≤ $V_{CC}$, Output Disabled | | −50 | | +50 | µA |
| $I_{OS}$ | Output Short Circuit Current[5, 6] | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V | | −30 | | −160 | mA |
| $I_{CC}$ | Power Supply Current[7] | $V_{CC}$ = Max., $I_{OUT}$ = 0 mA, f = 1 MHz, $V_{IN}$ = GND, $V_{CC}$ | Com'l/Ind. | | 125 | 200 | mA |
| | | | Com'l "L" −66 | | 75 | 125 | mA |
| | | | Military | | 125 | 250 | mA |
| $I_{BHL}$ | Input Bus Hold LOW Sustaining Current | $V_{CC}$ = Min., $V_{IL}$ = 0.8V | | +75 | | | µA |
| $I_{BHH}$ | Input Bus Hold HIGH Sustaining Current | $V_{CC}$ = Min., $V_{IH}$ = 2.0V | | −75 | | | µA |
| $I_{BHLO}$ | Input Bus Hold LOW Overdrive Current | $V_{CC}$ = Max. | | | | +500 | µA |
| $I_{BHHO}$ | Input Bus Hold HIGH Overdrive Current | $V_{CC}$ = Max. | | | | −500 | µA |

## Capacitance[6]

| Parameter | Description | Test Conditions | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $C_{I/O}$[8] | Input/Output Capacitance | $V_{IN}$ = 5.0V at f=1 MHz | | 8 | pF |
| $C_{CLK}$ | Clock Signal Capacitance | $V_{IN}$ = 5.0V at f = 1 MHz | 5 | 12 | pF |

**Notes:**

1. $T_A$ is the "instant on" case temperature.
2. See the last page of this specification for Group A subgroup testing information.
3. $I_{OH}$ = −2 mA, $I_{OL}$ = 2 mA for SDO.
4. These are absolute values with respect to device ground. All overshoots due to system or tester noise are included.
5. Not more than one output should be tested at a time. Duration of the short circuit should not exceed 1 second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
6. Tested initially and after any design or process changes that may affect these parameters.
7. Measured with 16-bit counter programmed into each logic block.
8. $C_{I/O}$ for dedicated inputs, and for I/O pins with JTAG functionality is 12 pF,and for the $ISR_{EN}$ pin is 15 pF Max.

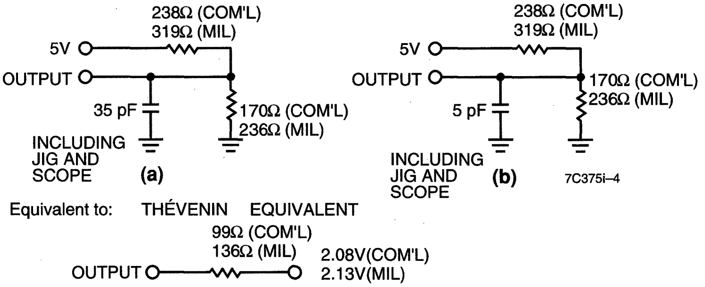
## Inductance[6]

| Parameter | Description | Test Conditions | 160-Lead TQFP | 160-Pin CQFP | 160-Pin CPGA | Unit |
|---|---|---|---|---|---|---|
| L | Maximum Pin Inductance | $V_{IN}$ = 5.0V at 5 = 1 MHz | 9 | 6 | 10 | nH |

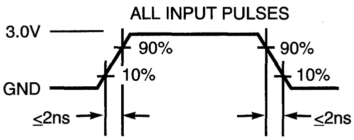
## Endurance Characteristics[6]

| Parameter | Description | Test Conditions | Min. | Unit |
|---|---|---|---|---|
| N | Minimum Reprogramming Cycles | Normal Programming Conditions | 100 | Cycles |

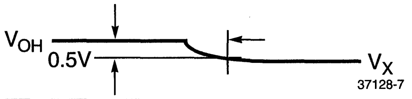
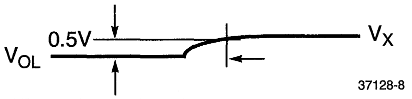
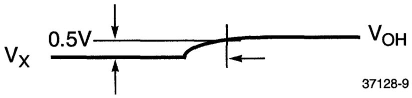
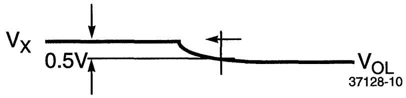
## AC Test Loads and Waveforms



7C375i–4



7C375i–5

| Parameter[9] | $V_X$ | Output Waveforms--Measurement Level |
|---|---|---|
| $t_{ER(-)}$ | 1.5V |  <br>37128-7 |
| $t_{ER(+)}$ | 2.6V |  <br>37128-8 |
| $t_{EA(+)}$ | 1.5V |  <br>37128-9 |
| $t_{EA(-)}$ | $V_{the}$ |  <br>37128-10 |

**(d) Test Waveforms**

**Note:**

[9] $t_{ER}$ measured with 5-pF AC Test Load and $t_{EA}$ measured with 35-pF AC Test Load.

## Switching Characteristics Over the Operating Range[10]

| Parameter | Description | 7C375i-125 | | 7C375i–100 | | 7C375i–83 | | 7C375iL–66 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| **Combinatorial Mode Parameters** | | | | | | | | | | |
| $t_{PD}$ | Input to Combinatorial Output | | 10 | | 12 | | 15 | | 20 | ns |
| $t_{PDL}$ | Input to Output Through Transparent Input or Output Latch | | 13 | | 15 | | 18 | | 22 | ns |
| $t_{PDLL}$ | Input to Output Through Transparent Input and Output Latches | | 15 | | 16 | | 19 | | 24 | ns |
| $t_{EA}$ | Input to Output Enable | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{ER}$ | Input to Output Disable | | 14 | | 16 | | 19 | | 24 | ns |
| **Input Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{WL}$ | Clock or Latch Enable Input LOW Time[6] | 3 | | 3 | | 4 | | 5 | | ns |
| $t_{WH}$ | Clock or Latch Enable Input HIGH Time[6] | 3 | | 3 | | 4 | | 5 | | ns |
| $t_{IS}$ | Input Register or Latch Set-Up Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{IH}$ | Input Register or Latch Hold Time | 2 | | 2 | | 3 | | 4 | | ns |
| $t_{ICO}$ | Input Register Clock or Latch Enable to Combinatorial Output | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{ICOL}$ | Input Register Clock or Latch Enable to Output Through Transparent Output Latch | | 16 | | 18 | | 21 | | 26 | ns |
| **Ouptut Registered/Latched Mode Parameters** | | | | | | | | | | |
| $t_{CO}$ | Clock or Latch Enable to Output | | 6.5 | | 7 | | 8 | | 10 | ns |
| $t_{S}$ | Set-Up Time from Input to Clock or Latch Enable | 5.5 | | 6 | | 8 | | 10 | | ns |
| $t_{H}$ | Register or Latch Data Hold Time | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{CO2}$ | Output Clock or Latch Enable to Output Delay (Through Memory Array) | | 14 | | 16 | | 19 | | 24 | ns |
| $t_{SCS}$ | Output Clock or Latch Enable to Output Clock or Latch Enable (Through Memory Array) | 8 | | 10 | | 12 | | 15 | | ns |
| $t_{SL}$ | Set-Up Time from Input Through Transparent Latch to Output Register Clock or Latch Enable | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{HL}$ | Hold Time for Input Through Transparent Latch from Output Register Clock or Latch Enable | 0 | | 0 | | 0 | | 0 | | ns |
| $f_{MAX1}$ | Maximum Frequency with Internal Feedback (Least of $1/t_{SCS}$, $1/(t_S + t_H)$, or $1/t_{CO})$[6] | 125 | | 100 | | 83 | | 66 | | MHz |
| $f_{MAX2}$ | Maximum Frequency Data Path in Output Registered/Latched Mode (Lesser of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $1/t_{CO})$ | 158.3 | | 143 | | 125 | | 100 | | MHz |
| $f_{MAX3}$ | Maximum Frequency with External Feedback (Lesser of $1/(t_{CO} + t_S)$ and $1/(t_{WL} + t_{WH}$, | 83.3 | | 76.9 | | 62.5 | | 50 | | MHz |

Shaded areas contain preliminary information.

**Notes:**

[10] All AC parameters are measured with 16 outputs switching and 35-pF AC Test Load.

[11] This specification is intended to guarantee interface compatibility of the other members of the CY7C370i family with the CY7C375i. This specification is met for the devices operating at the same ambient temperature and at the same power supply voltage.

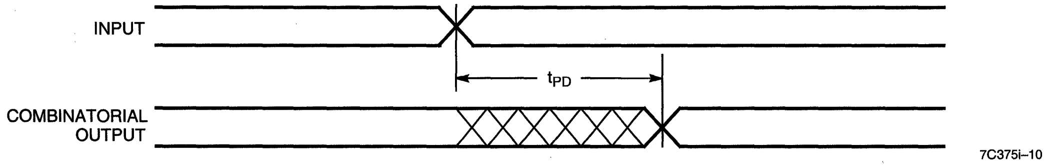
## Switching Characteristics Over the Operating Range[10] (continued)

| Parameter | Description | 7C375i-125 | | 7C375i–100 | | 7C375i–83 | | 7C375iL–66 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{OH}$–$t_{IH}$ 37x | Output Data Stable from Output Clock Minus Input Register Hold Time for 7C37x[6, 11] | 0 | | 0 | | 0 | | 0 | | ns |
| **Pipelined Mode Parameters** | | | | | | | | | | |
| $t_{ICS}$ | Input Register Clock to Output Register Clock | 8 | | 10 | | 12 | | 15 | | ns |
| $f_{MAX4}$ | Maximum Frequency in Pipelined Mode (Least of $1/(t_{CO} + t_{IS})$, $1/t_{ICS}$, $1/(t_{WL} + t_{WH})$, $1/(t_{IS} + t_{IH})$, or $1/t_{SCS}$) | 125 | | 100 | | 83.3 | | 66.6 | | MHz |
| **Reset/Preset Parameters** | | | | | | | | | | |
| $t_{RW}$ | Asynchronous Reset Width[6] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{RR}$ | Asynchronous Reset Recovery Time[6] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{RO}$ | Asynchronous Reset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| $t_{PW}$ | Asynchronous Preset Width[6] | 10 | | 12 | | 15 | | 20 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[6] | 12 | | 14 | | 17 | | 22 | | ns |
| $t_{PO}$ | Asynchronous Preset to Output | | 16 | | 18 | | 21 | | 26 | ns |
| **Tap Controller Parameter** | | | | | | | | | | |
| $f_{TAP}$ | Tap Controller Frequency | 500 | | 500 | | 500 | | 500 | | kHz |

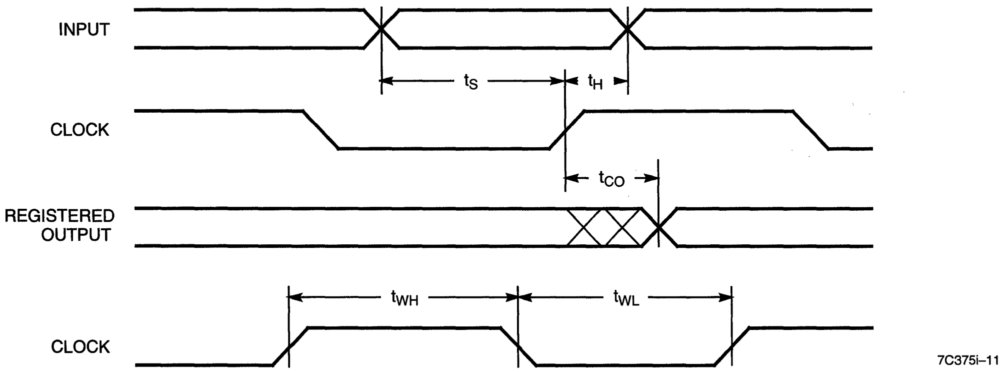
Shaded areas contain preliminary information.

## Switching Waveforms

### Combinatorial Output

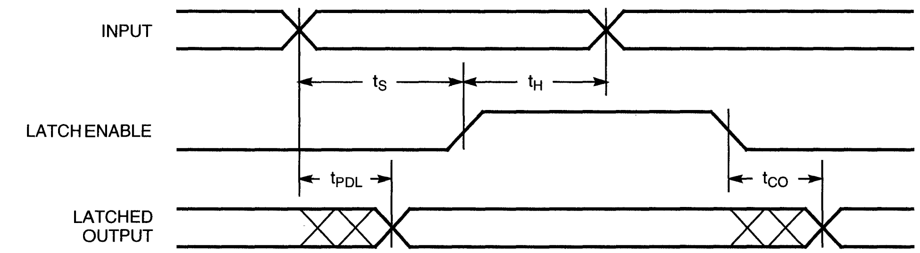


### Registered Output



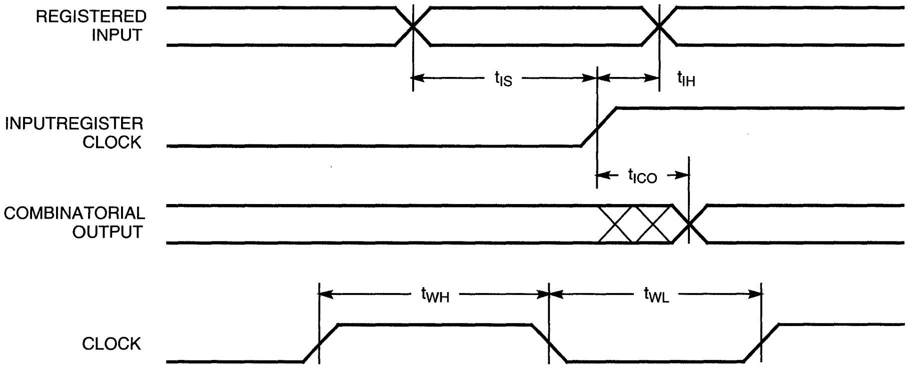
## Switching Waveforms (continued)

### Latched Output



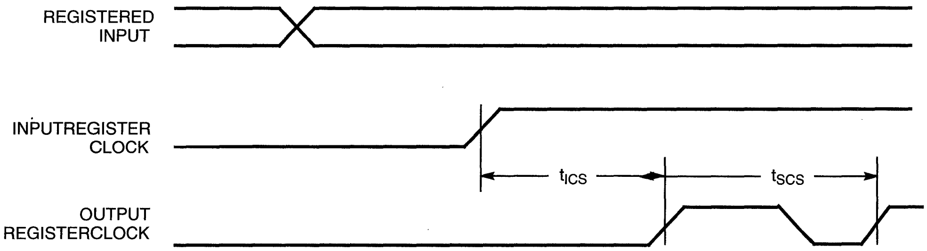
7C375i–12

### Registered Input



7C375i–13

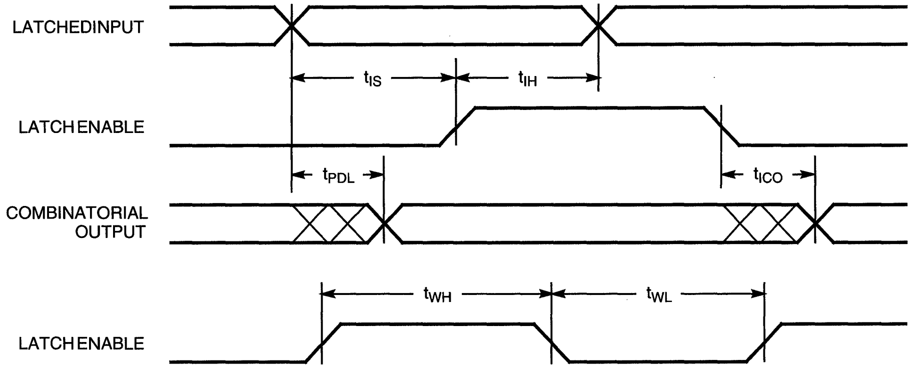
### Clock to Clock



7C375i–14

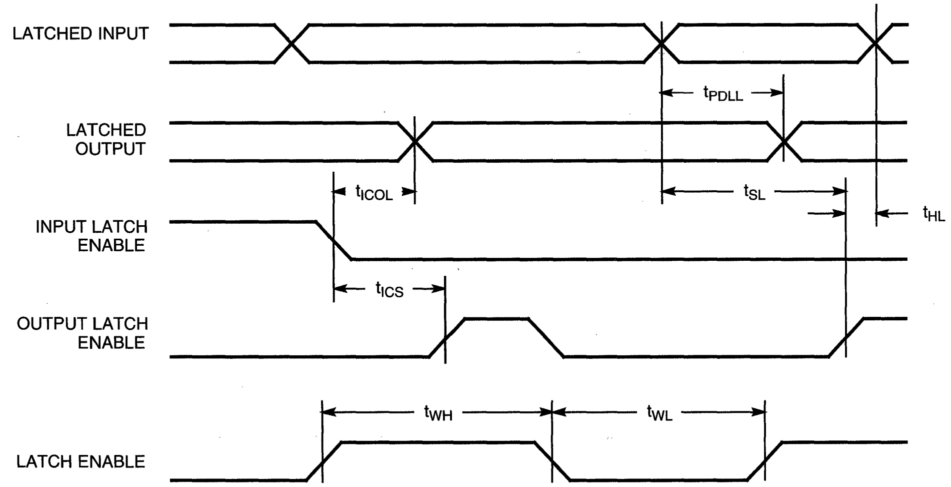
## Switching Waveforms (continued)

### Latched Input



7C375i–15

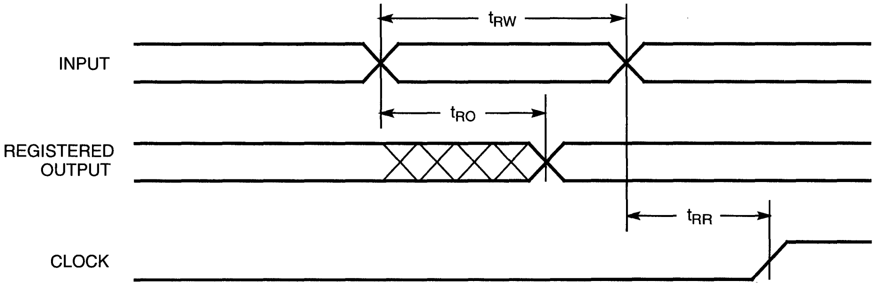
### Latched Input and Output



7C375i–16

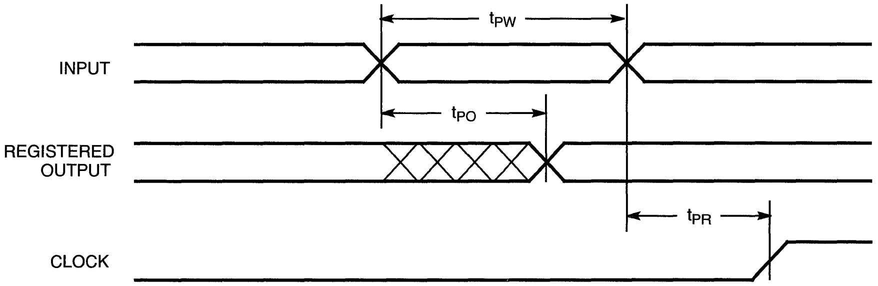
## Switching Waveforms (continued)

### Asynchronous Reset



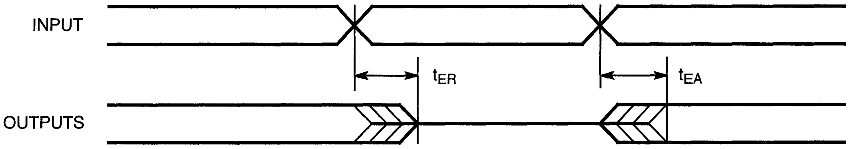
7C375i–17

### Asynchronous Preset



7C375i–18

### Output Enable/Disable



7C375i–20

## Ordering Information

| Speed (MHz) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 125 | CY7C375i-125AC | A160 | 160-Lead Thin Quad Flatpack | Commercial |
| 100 | CY7C375i–100AC | A160 | 160-Lead Thin Quad Flatpack | Commercial |
| 83 | CY7C375i–83AC | A160 | 160-Lead Thin Quad Flatpack | Commercial |
| | CY7C375i–83AI | A160 | 160-Lead Thin Quad Flatpack | Industrial |
| | CY7C375i–83GMB | G160 | 160-Pin Grid Array | Military |
| | CY7C375i–83UMB | U162 | 160-Pin Ceramic Quad Flatpack[12] | |
| 66 | CY7C375i–66AC | A160 | 160-Lead Thin Quad Flatpack | Commercial |
| | CY7C375i–66AI | A160 | 160-Lead Thin Quad Flatpack | Industrial |
| | CY7C375i–66GMB | G160 | 160-Pin Grid Array | Military |
| | CY7C375i–66UMB | U162 | 160-Pin Ceramic Quad Flatpack[12] | |
| 66 | CY7C375iL–66AC | A160 | 160-Lead Thin Quad Flatpack | Commercial |

Shaded areas contain preliminary information.

## MILITARY SPECIFICATIONS

### Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD}$ | 9, 10, 11 |
| $t_{CO}$ | 9, 10, 11 |
| $t_{ICO}$ | 9, 10, 11 |
| $t_S$ | 9, 10, 11 |
| $t_H$ | 9, 10, 11 |
| $t_{IS}$ | 9, 10, 11 |
| $t_{IH}$ | 9, 10, 11 |
| $t_{ICS}$ | 9, 10, 11 |

Document #: 38–00494–C

**Note:**

[12] Standard product ships trim and formed in a carrier. This product is also available in a molded carrier ring. Contact local Cypress office for package information.

ISR, UltraLogic, FLASH370, FLASH370i, and *Warp2*Sim are trademarks of Cypress Semiconductor Corporation.
ABEL is a trademark of Data I/O Corporation.
LOG/iC is a trademark of Isdata Corporation.
CUPL is a trademark of Logical Devices Incorporated.

fax id: 6125

CYPRESS

# CPLD Family
## FLASH370™
# UltraLogic™ High-Density Flash CPLDs

## Features

- **Flash erasable CMOS CPLDs**
- **High density**
  - **—32–128 macrocells**
  - **—32–128 I/O pins**
  - **—Multiple clock pins**
- **High speed**
  - **—$t_{PD}$ = 8.5–12 ns**
  - **—$t_S$ = 5–7 ns**
  - **—$t_{CO}$ = 6–7 ns**
- **Fast Programmable Interconnect Matrix (PIM)**
  - **—Uniform predictable delay, independent of routing**
- **Intelligent product term allocator**
  - **—0–16 product terms to any macrocell**
  - **—Provides product term steering on an individual basis**
  - **—Provides product term sharing among local macrocells**
  - **—Doesn't strand macrocells**
- **Simple timing model**
  - **—No fanout delays**
  - **—No expander delays**
  - **—No dedicated vs. I/O pin delays**
  - **—No additional delay through PIM**
  - **—No penalty for using full 16 product terms**
  - **—No delay for steering or sharing product terms**
- **Flexible clocking**
  - **—2-4 clock pins per device**
  - **—Clock polarity control**
- **Security bit and user ID supported**
- **Packages**
  - **—44–160 pins**
  - **—PLCC, CLCC, PGA, and TQFP packages**
- ***Warp2*®**
  - **—Low-cost IEEE 1164-compliant VHDL development system**
  - **—Available on PC, Sun, and HP platforms**
  - **—Supports all PLDs and CPLDs**
- ***Warp3*® CAE development system**
  - **—VHDL input**
  - **—ViewLogic graphical user interface**
  - **—Schematic capture (ViewDraw™)**
  - **—VHDL simulation (ViewSim™)**
  - **—Available on PC, HP, and Sun platforms**

## General Description

The FLASH370™ family of CMOS CPLDs provides a range of high-density programmable logic solutions with unparalleled performance. Each member of the family is designed with Cypress's state-of-the-art Flash technology. All of the devices are electrically erasable and reprogrammable, simplifying product inventory and reducing costs.

The FLASH370 family is designed to bring the flexibility, ease of use and performance of the 22V10 to high-density CPLDs. The architecture is based on a number of logic blocks that are connected by a Programmable Interconnect Matrix (PIM). Each logic block features its own product term array, product term allocator array, and 16 macrocells. The PIM distributes signals from one logic block to another as well as all inputs from pins.

The family features a wide variety of densities and pin counts to choose from. At each density there are two packaging options to choose from—one that is I/O intensive and another that is register intensive. For example, the CY7C374 and CY7C375 both feature 128 macrocells. On the CY7C374, available in an 84-pin package, half of the macrocells are buried and half are available on I/O pins. On the CY7C375 all of the macrocells are fed to I/O pins and the device is available in the 160-pin package. *Figure 1* shows a block diagram of the CY7C374/5.

## Functional Description

### Programmable Interconnect Matrix

The Programmable Interconnect Matrix (PIM) consists of a completely global routing matrix for signals from I/O pins and feedbacks from the logic blocks. The PIM is an extremely robust interconnect that avoids fitting and density limitations. Routing is automatically accomplished by software and the propagation delay through the PIM is transparent to the user. Signals from any pin or any logic block can be routed to any or all logic blocks.

3

## FLASH370 Selection Guide

| Device | Pins | Macrocells | Dedicated Inputs | I/O Pins | Flip-Flops | Speed ($t_{PD}$) | Speed ($f_{MAX}$) |
|---|---|---|---|---|---|---|---|
| 371 | 44 | 32 | 6 | 32 | 44 | 8.5 | 143 |
| 372 | 44 | 64 | 6 | 32 | 76 | 10 | 125 |
| 373 | 84 | 64 | 6 | 64 | 76 | 10 | 125 |
| 374 | 84 | 128 | 6 | 64 | 140 | 12 | 100 |
| 375 | 160 | 128 | 6 | 128 | 140 | 12 | 100 |

## Logic Block Diagram

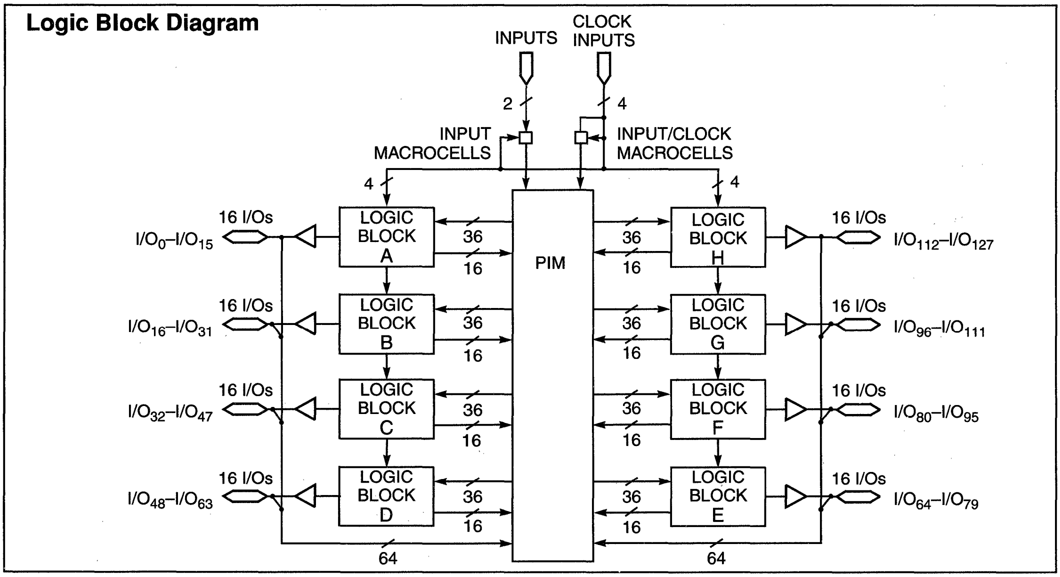


**Figure 1. CY7C375 Block Diagram**

## Functional Description (continued)

The inputs to the PIM consist of all I/O and dedicated input pins and all macrocell feedbacks from within the logic blocks. The number of PIM inputs increases with pincount and the number of logic blocks. The outputs from the PIM are signals routed to the appropriate logic block(s). Each logic block receives 36 inputs from the PIM and their complements, allowing for 32-bit operations to be implemented in a single pass through the device. The wide number of inputs to the logic block also improves the routing capacity of the FLASH370 family.

An important feature of the PIM involves timing. The propagation delay through the PIM is accounted for in the timing specifications for each device. There is no additional delay for traveling through the PIM. In fact, all inputs travel through the PIM. Likewise, there are no route-dependent timing parameters on the FLASH370 devices. The worst-case PIM delays are incorporated in all appropriate FLASH370 specifications.

Routing signals through the PIM is completely invisible to the user. All routing is accomplished by software—no hand routing is necessary. *Warp* and third-party development packages automatically route designs for the FLASH370 family in a matter of minutes. Finally, the rich routing resources of the FLASH370 family accommodate last minute logic changes while maintaining fixed pin assignments.

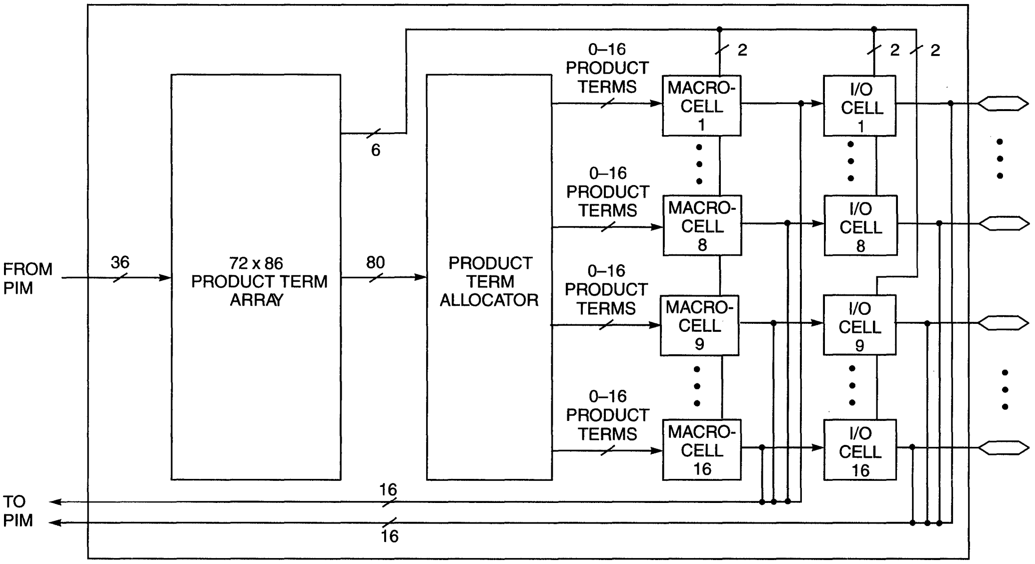
### Logic Block

The logic block is the basic building block of the FLASH370 architecture. It consists of a product term array, an intelligent product-term allocator, 16 macrocells, and a number of I/O cells. The number of I/O cells varies depending on the device used.

There are two types of logic blocks in the FLASH370 family. The first type features an equal number (16) of I/O cells and macrocells and is shown in *Figure 2*. This architecture is best for I/O-intensive applications. The second type of logic block features a buried macrocell along with each I/O macrocell. In other words, in each logic block, there are eight macrocells that are connected to I/O cells and eight macrocells that are internally fed back to the PIM only. This organization is designed for register-intensive applications and is displayed in *Figure 3*. Note that at each FLASH370 density (except the smallest), an I/O intensive and a register-intensive device is available.

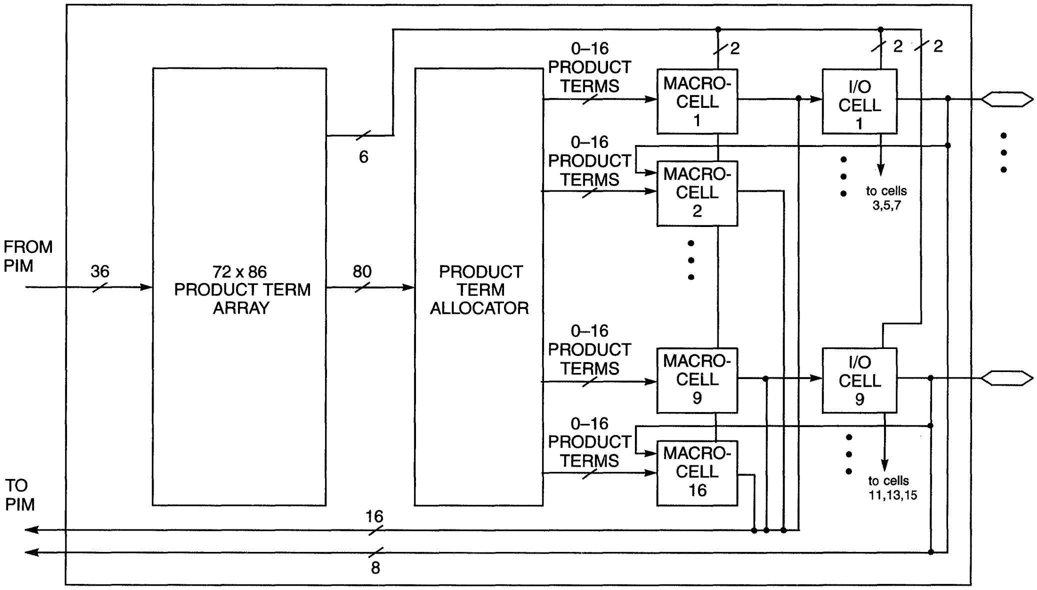
### Product Term Array

Each logic block features a 72 x 86 programmable product term array. This array is fed with 36 inputs from the PIM, which originate from macrocell feedbacks and device pins. Active LOW and active HIGH versions of each of these inputs are generated to create the full 72-input field. The 86 product terms in the array can be created from any of the 72 inputs.



flash370–2

**Figure 2. Logic Block for CY7C371, CY7C373, and CY7C375 (I/O Intensive)**



flash370–3

**Figure 3. Logic Block for CY7C372 and CY7C374 (Register Intensive)**

Of the 86 product terms, 80 are for general-purpose use for the 16 macrocells in the logic block. Four of the remaining six product terms in the logic block are output enable (OE) product terms. Each of the OE product terms controls up to eight of the 16 macrocells and is selectable on an individual macrocell basis. In other words, each I/O cell can select between one of two OE product terms to control the output buffer. The first two of these four OE product terms are available to the upper half of the I/O macrocells in a logic block. The other two OE product terms are available to the lower half of the I/O macrocells in a logic block. The final two product terms in each logic block are dedicated asynchronous set and asynchronous reset product terms.

### Product Term Allocator

Through the product term allocator, software automatically distributes product terms among the 16 macrocells in the logic block as needed. A total of 80 product terms are available from the local product term array. The product term allocator provides two important capabilities without affecting performance: product term steering and product term sharing.

*Product Term Steering*

Product term steering is the process of assigning product terms to macrocells as needed. For example, if one macrocell requires ten product terms while another needs just three, the product term allocator will "steer" ten product terms to one macrocell and three to the other. On FLASH370 devices, product terms are steered on an individual basis. Any number between 0 and 16 product terms can be steered to any macrocell. Note that 0 product terms is useful in cases where a particular macrocell is unused or used as an input register.

*Product Term Sharing*

Product term sharing is the process of using the same product term among multiple macrocells. For example, if more than one output has one or more product terms in its equation that are common to other outputs, those product terms are only programmed once. The FLASH370 product term allocator allows sharing across groups of four output macrocells in a variable fashion. The software automatically takes advantage of this capability—the user does not have to intervene. Note that greater usable density can often be achieved if the user "floats" the pin assignment. This allows the compiler to group macrocells that have common product terms adjacently.

Note that neither product term sharing nor product term steering have any effect on the speed of the product. All worst-case steering and sharing configurations have been incorporated in the timing specifications for the FLASH370 devices.

### FLASH370 Macrocell

*I/O Macrocell*

Within each logic block there are 8 or 16 I/O macrocells depending on the device used. *Figure 4* illustrates the architecture of the I/O macrocell. The macrocell features a register that can be configured as combinatorial, a D flip-flop, a T flip-flop, or a level-triggered latch.

The register can be asynchronously set or asynchronously reset at the logic block level with the separate set and reset product terms. Each of these product terms features programmable polarity. This allows the registers to be set or reset based on an AND expression or an OR expression.

Clocking of the register is very flexible. Depending on the device, either two or four global synchronous clocks are available to clock the register. Furthermore, each clock features programmable polarity so that registers can be triggered on falling as well as rising edges (see the Dedicated/Clock Inputs section). Clock polarity is chosen at the logic block level.

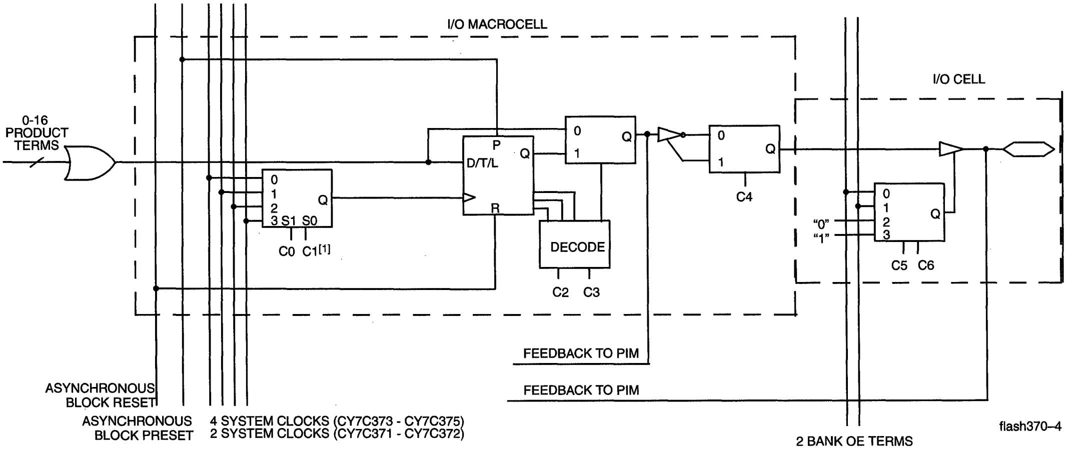


**Figure 4. I/O Macrocell**

**Note:**

[1] C1 is not used on the CY7C371 and CY7C372.

At the output of the macrocell, a polarity control mux is available to select active LOW or active HIGH signals. This has the added advantage of allowing significant logic reduction to occur in many applications.

The FLASH370 macrocell features a feedback path to the PIM separate from the I/O pin input path. This means that if the macrocell is buried (fed back internally only), the associated I/O pin can still be used as an input.

*Buried Macrocell*

Some of the devices in the FLASH370 family feature additional macrocells that do not feed individual I/O pins. *Figure 5* displays the architecture of the I/O and buried macrocells for these devices. The I/O macrocell is identical to the one on devices without buried macrocells.

The buried macrocell is very similar to the I/O macrocell. Again, it includes a register that can be configured as combinatorial, a D flip-flop, a T flip-flop, or a latch. The clock for this register has the same options as described for the I/O macrocell. The primary difference between the I/O macrocell and the buried macrocell is that the buried macrocell does not have the ability to output data directly to an I/O pin.

One additional difference on the buried macrocell is the addition of input register capability. The buried macrocell can be configured to act as an input register (D-type or latch) whose input comes from the I/O pin associated with the neighboring macrocell. The output of all buried macrocells is sent directly to the PIM regardless of its configuration.

### FLASH370 I/O Cell

The I/O cell on the FLASH370 devices is illustrated along with the I/O macrocell in *Figures 4* and *5*. The user can program the I/O cell to change the way the three-state output buffer is enabled and/or disabled. Each output can be set permanently on (output only), permanently off (input only), or dynamically controlled by one of two OE product terms.

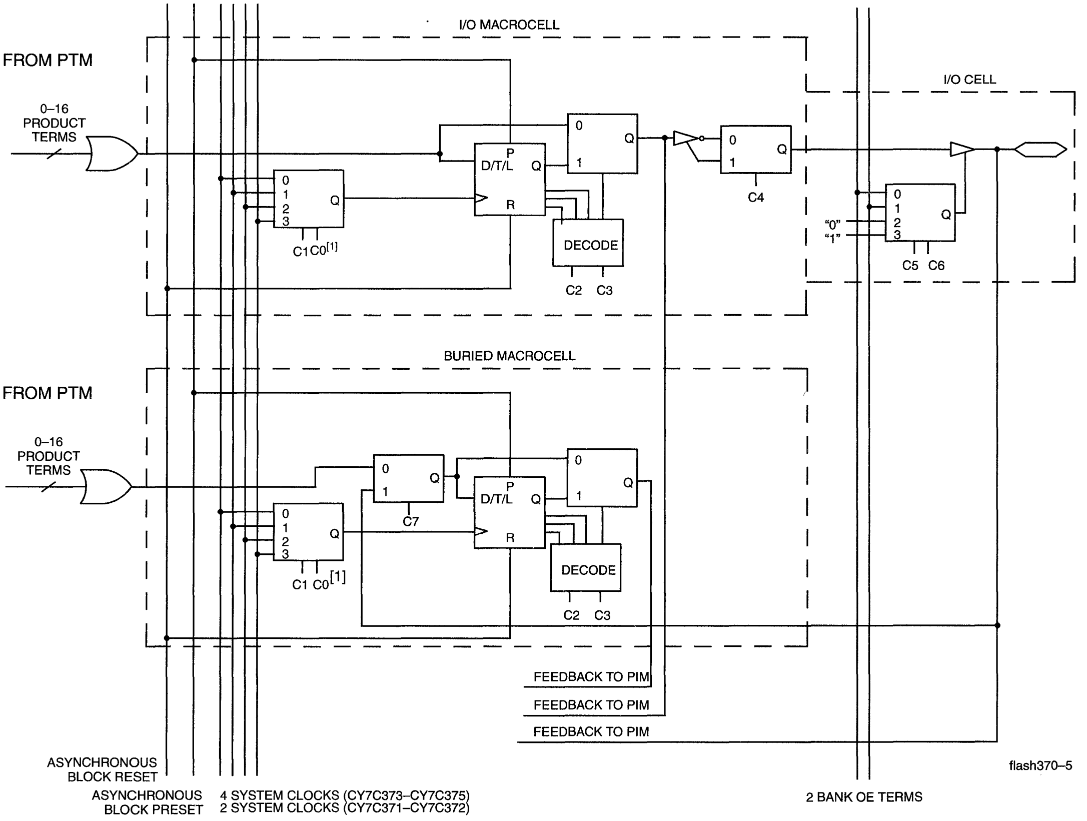


**Figure 5. I/O and Buried Macrocells**

### Dedicated/Clock Inputs

Six pins on each member of the FLASH370 family are designated as input-only. There are two types of dedicated inputs on FLASH370 devices: input pins and input/clock pins. *Figure 6* illustrates the architecture for input pins. Four input options are available for the user: combinatorial, registered, double-registered, or latched. If a registered or latched option is selected, any one of the input clocks can be selected for control.

*Figure 7* illustrates the architecture of input/clock pins. There are either two or four input/clock pins available, depending on the device selected. (The CY7C371 and CY7C372 have two input/clock pins while the other devices have four input/clock pins.) Like the input pins, input/clock pins can be combinatorial, registered, double registered, or latched. In addition, these pins feed the clocking structures throughout the device. The clock path at the input is user-configurable in polarity. The polarity of the clock signal can also be controlled by the user. Note that this polarity is separately controlled for input registers and output registers.

### Timing Model

One of the most important features of the FLASH370 family is the simplicity of its timing. All delays are worst case and system performance is unaffected by the features used or not used on the parts. *Figure 8* illustrates the true timing model for the 8.5-ns devices. For combinatorial paths, any input to any output incurs an 8.5-ns worst-case delay regardless of the amount of logic used. For synchronous systems, the input set-up time to the output macrocells for any input is 5.0 ns and the clock to output time is also 6.0 ns. Again, these measurements are for any output and clock, regardless of the logic used.

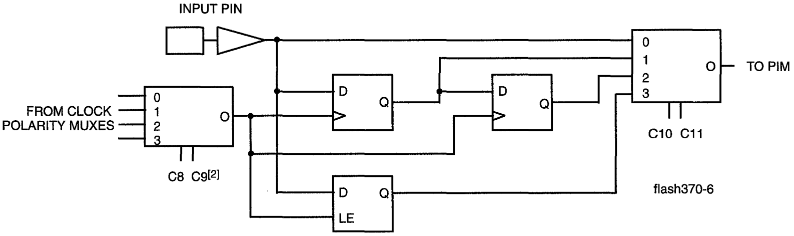


**Figure 6. input Pins**

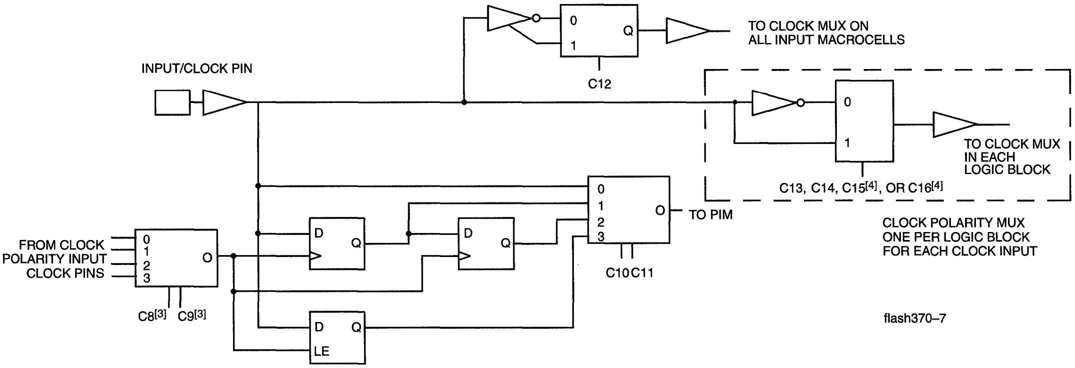


**Figure 7. Input/Clock Pins**

**Notes:**

[2] C9 is not used on the CY7C371 and CY7C372.

[3] C8 and C9 are not included on the CY7C371 and CY7C372 since each input/clock pin has the other input/clock pin as its clock.

[4] C15 and C16 are not used on the CY7C371 and CY7C372 since there are two clocks.

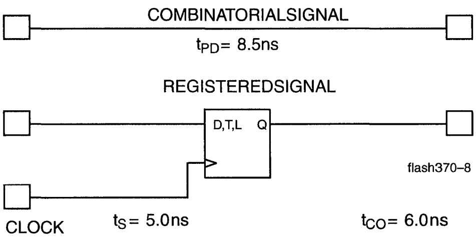


**Figure 8. Timing Model for CY7C371**

Stated another way, the FLASH370 features:

- no fanout delays
- no expander delays
- no dedicated vs. I/O pin delays
- no additional delay through PIM
- no penalty for using 0-16 product terms
- no added delay for steering product terms
- no added delay for sharing product terms
- no routing delays
- no output bypass delays

The simple timing model of the FLASH370 family eliminates unexpected performance penalties.

## Development Software Support

### *Warp2*

*Warp2* is a state-of-the-art VHDL compilers for designing with Cypress PLDs and CPLDs. *Warp2* utilizes a proper subset of IEEE 1164 VHDL as the Hardware Description Language (HDL) for design entry. VHDL provides a number of significant benefits for the design entry. *Warp2* accepts VHDL input, synthesizes and optimizes the entered design, and outputs a JEDEC map for the desired device. For simulation, *Warp2* provides the graphical waveform simulator called Nova.

VHDL (VHSIC Hardware Description Language) is an open, powerful, non-proprietary language that is a standard for behavioral design entry and simulation. It is already mandated for use by the Department of Defense and supported by every major vendor of CAE tools. VHDL allows designers to learn a single language that is useful for all facets of the design process. See separate data sheet for further information.

### *Warp3*

*Warp3* is a sophisticated design tool that is based on the latest version of ViewLogic's CAE design environment. *Warp3* features schematic capture (ViewDraw™), VHDL waveform simulation (ViewSim™), a VHDL debugger, and VHDL synthesis, all integrated in a graphical design environment. *Warp3* is available on PCs using Windows 3.1 or subsequent versions, and on HP and Sun workstations. See separate data sheet for further information.

### Third-Party Software

Cypress maintains a very strong commitment to third-party design software vendors. All major third-party software vendors (including ABEL™, LOG/iC™, CUPL™, and Minc) will provide support for the FLASH370 family of devices. To expedite this support, Cypress supplies vendors with all pertinent architectural information as well as design fitters for our products.

## Programming

The *Impulse3*™ device programmer from Cypress programs all Cypress PLDs, CPLDs, and PROMs. This unit is a programmer that connects to any IBM-compatible PC via the printer port.

### Third-Party Programmers

As with development software, Cypress strongly supports third-party programmers. All major third-party programmers (including Data I/O, Logical Devices, Minato, SMS, and Stag) will support the FLASH370 family.

Document #: 38-00215-E

**For new designs, see CY7C371i.**

CYPRESS

# CY7C371

# UltraLogic™ 32-Macrocell Flash CPLD

## Features

- 32 macrocells in two logic blocks
- 32 I/O pins
- 6 dedicated inputs including 2 clock pins
- No hidden delays
- High speed
  - —$f_{MAX}$ = 143 MHz
  - —$t_{PD}$= 8.5 ns
  - —$t_S$ = 5 ns
  - —$t_{CO}$ = 6 ns
- Electrically alterable FLASH technology
- Available in 44-pin PLCC, CLCC, and TQFP packages
- Pin compatible with the CY7C372

## Functional Description

The CY7C371 is a Flash erasable Complex Programmable Logic Device (CPLD) and is part of the FLASH370 family of high-density, high-speed CPLDs. Like all members of the FLASH370 family, the CY7C371 is designed to bring the ease of use and high performance of the 22V10 to high-density CPLDs.

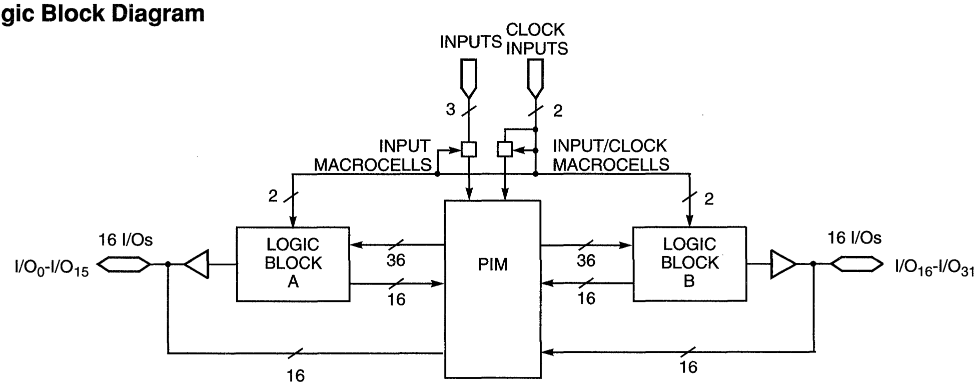
The 32 macrocells in the CY7C371 are divided between two logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370 architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

Like all members of the FLASH370 family, the CY7C371 is rich in I/O resources. Each macrocell in the device features an associated I/O pin, resulting in 32 I/O pins on the CY7C371. In addition, there are four dedicated inputs and two input/clock pins.

Finally, the CY7C371 features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used or the type of application, the timing parameters on the CY7C371 remain the same.

## Logic Block Diagram



## Selection Guide

| | | 7C371-143 | 7C371-110 | 7C371-83 | 7C371L-83 | 7C371-66 | 7C371L-66 |
|---|---|---|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | | 8.5 | 10 | 12 | 12 | 15 | 15 |
| Minimum Set-Up, $t_S$ (ns) | | 5 | 6 | 10 | 10 | 12 | 12 |
| Maximum Clock to Output, $t_{CO}$ (ns) | | 6 | 6.5 | 10 | 10 | 12 | 12 |
| Maximum Supply Current, $I_{CC}$ (mA) | Commercial | 220 | 175 | 175 | 90 | 175 | 90 |
| | Military/Ind. | | | 220 | 110 | 220 | 110 |

Shaded area contains preliminary information.

Ultralogic and FLASH370 are trademarks of Cypress Semiconductor Corporation.

**For new designs, see CY7C372i.**

CYPRESS

# CY7C372

# UltraLogic™ 64-Macrocell Flash CPLD

## Features

- **64 macrocells in four logic blocks**
- **32 I/O pins**
- **6 dedicated inputs including 2 clock pins**
- **No hidden delays**
- **High speed**
  - **—$f_{MAX}$ = 125 MHz**
  - **—$t_{PD}$ = 10 ns**
  - **—$t_S$ = 5.5 ns**
  - **—$t_{CO}$ = 6.5 ns**
- **Electrically alterable Flash technology**
- **Available in 44-pin PLCC and CLCC packages**
- **Pin compatible with the CY7C371**

## Functional Description

The CY7C372 is a Flash erasable Complex Programmable Logic Device (CPLD) and is part of the FLASH370™ family of high-density, high-speed CPLDs. Like all members of the FLASH370 family, the CY7C372 is designed to bring the ease of use and high performance of the 22V10 to high-density CPLDs.

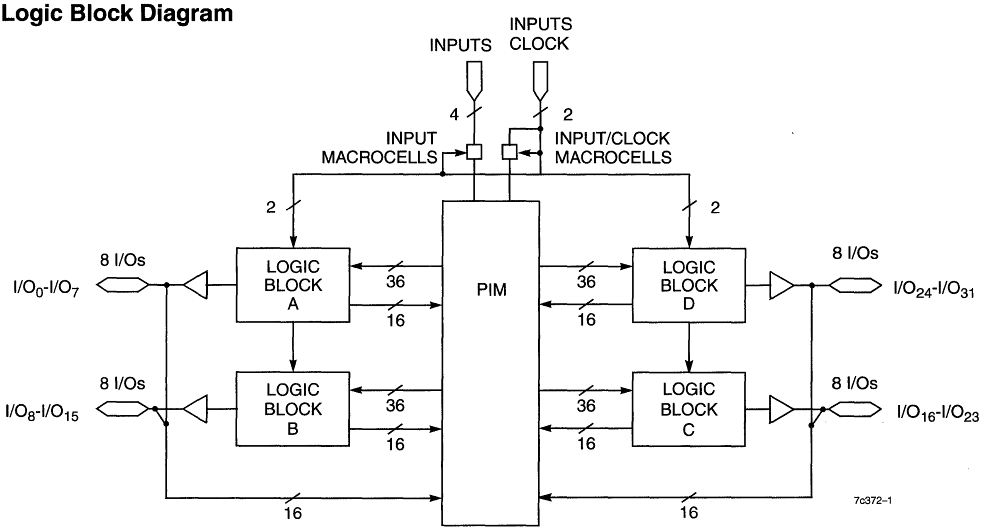
The 64 macrocells in the CY7C372 are divided between four logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370 architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

Like all members of the FLASH370 family, the CY7C372 is rich in I/O resources. Every two macrocells in the device feature an associated I/O pin, resulting in 32 I/O pins on the CY7C372. In addition, there are four dedicated inputs and two input/clock pins.

Finally, the CY7C372 features a very simple timing model. Unlike other high-density CPLD architectures, there are no hidden speed delays such as fanout effects, interconnect delays, or expander delays. Regardless of the number of resources used or the type of application, the timing parameters on the CY7C372 remain the same.

## Logic Block Diagram



## Selection Guide

| | | 7C372-125 | 7C372-100 | 7C372-83 | 7C372-66 | 7C372L-66 |
|---|---|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | | 10 | 12 | 15 | 20 | 20 |
| Minimum Set-up, $t_S$ (ns) | | 5.5 | 6 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | | 6.5 | 6.5 | 8 | 10 | 10 |
| Maximum Supply Current, $I_{CC}$ (mA) | Commercial | 280 | 250 | 250 | 250 | 125 |
| | Military/Industrial | | | 300 | 300 | |

Shaded areas contain preliminary information.

UltraLogic and FLASH370 are trademarks of Cypress Semiconductor Corporation.

**For new designs, see CY7C373i.**

CYPRESS

# CY7C373

# UltraLogic™ 64-Macrocell Flash CPLD

## Features

- **64 macrocells in four logic blocks**
- **64 I/O pins**
- **6 dedicated inputs including 4 clock pins**
- **No hidden delays**
- **High speed**
  - —$f_{MAX}$ = 125 MHz
  - —$t_{PD}$ = 10 ns
  - —$t_S$ = 5.5 ns
  - —$t_{CO}$ = 6.5 ns
- **Electrically alterable Flash technology**
- **Available in 84-pin PLCC and 100-pin TQFP packages**
- **Pin compatible with the CY7C374**

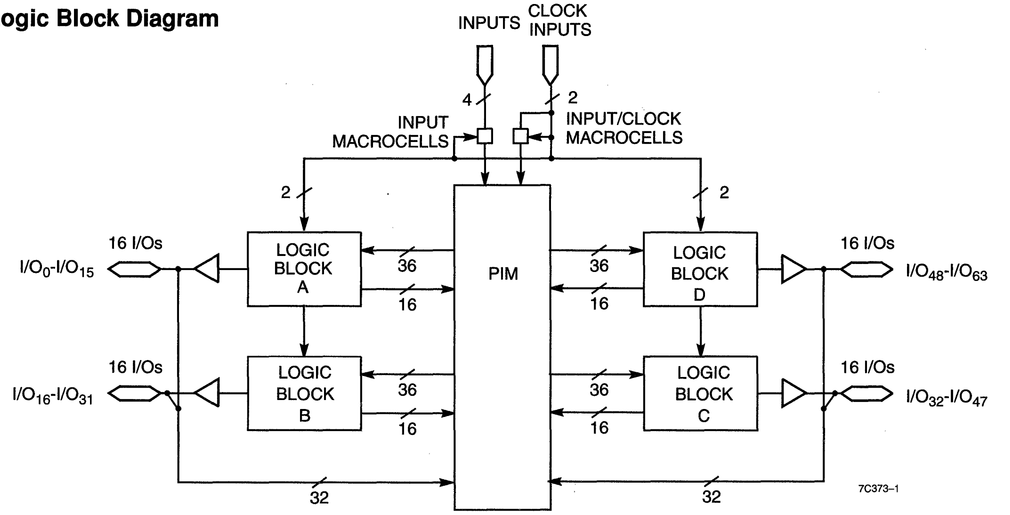
## Functional Description

The CY7C373 is a Flash erasable Complex Programmable Logic Device (CPLD) and is part of the FLASH370™ family of high-density, high-speed CPLDs. Like all members of the FLASH370 family, the CY7C373 is designed to bring the ease of use and high performance of the 22V10 to high-density CPLDs.

The 64 macrocells in the CY7C373 are divided between four logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370 architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

## Logic Block Diagram



## Selection Guide

| | | 7C373-125 | 7C373-100 | 7C373-83 | 7C373-66 | 7C373L-66 |
|---|---|---|---|---|---|---|
| Maximum Propagation Delay (ns) | | 10 | 12 | 15 | 20 | 20 |
| Minimum Set-up, $t_S$ (ns) | | 5.5 | 6 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | | 6.5 | 6.5 | 8 | 10 | 10 |
| Maximum Supply Current, $I_{CC}$ (mA) | Commercial | 280 | 250 | 250 | 250 | 125 |
| | Industrial | | | 300 | 300 | |

Shaded area contains preliminary information.

Ultralogic and FLASH370 are trademarks of Cypress Semiconductor Corporation.

**For new designs, see CY7C374i.**

CYPRESS

# CY7C374

# UltraLogic™ 128-Macrocell Flash CPLD

## Features

- **128 macrocells in eight logic blocks**
- **64 I/O pins**
- **6 dedicated inputs including 4 clock pins**
- **No hidden delays**
- **High speed**
  - **—$f_{MAX}$ = 100 MHz**
  - **—$t_{PD}$ = 12 ns**
  - **—$t_S$ = 6 ns**
  - **—$t_{CO}$ = 7 ns**
- **Electrically Alterable Flash technology**
- **Available in 84-pin PLCC, 84-pin CLCC, 100-pin TQFP, and 84-pin PGA packages**
- **Pin compatible with the CY7C373**

## Functional Description

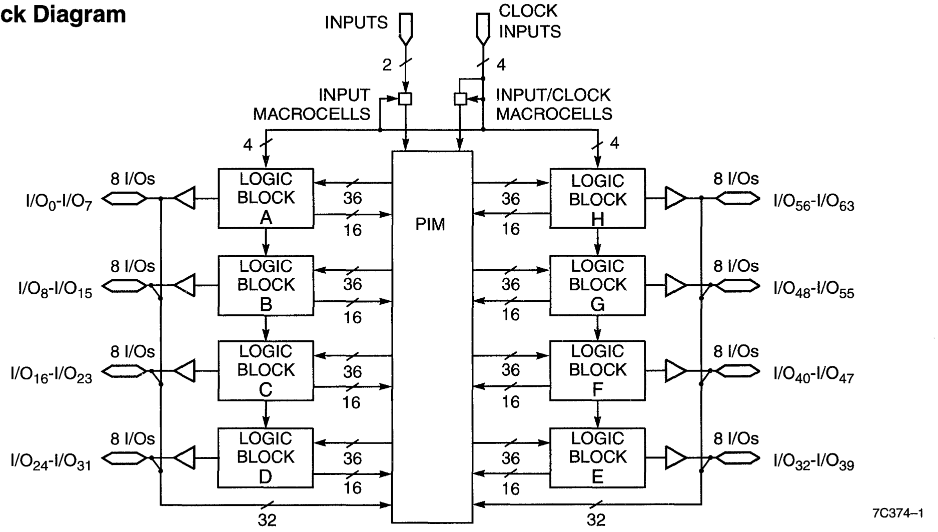
The CY7C374 is a Flash erasable Complex Programmable Logic Device (CPLD) and is part of the FLASH370™ family of high-density, high-speed CPLDs. Like all members of the FLASH370 family, the CY7C374 is designed to bring the ease of use and high performance of the 22V10 to high-density CPLDs.

The 128 macrocells in the CY7C374 are divided between eight logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370 architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

The CY7C374 is a register intesive 128-Macrocell CPLD. Every two macrocells in the device feature an associated I/O pin, resulting in 64 I/O pins on the CY7C374. In addition, there are two dedicated inputs and four input/clock pins.

## Logic Block Diagram



## Selection Guide

| | | 7C374-100 | 7C374-83 | 7C374-66 | 7C374L-66 |
|---|---|---|---|---|---|
| Maximum Propagation Delay $t_{PD}$ (ns) | | 12 | 15 | 20 | 20 |
| Minimum Set-Up, $t_S$ (ns) | | 6 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ncs) | | 7 | 8 | 10 | 10 |
| Maximum Supply Current, $I_{CC}$ (mA) | Commercial | 300 | 300 | 300 | 150 |
| | Military/Industrial | | 370 | 370 | |

UltraLogic and FLASH370 are trademarks of Cypress Semiconductor Corporation.

**For new designs, see CY7C375i.**

CYPRESS

# CY7C375

# UltraLogic™ 128-Macrocell Flash CPLD

## Features

- **28 macrocells in eight logic blocks**
- **28 I/O pins**
- **6 dedicated inputs including 4 clock pins**
- **No hidden delays**
- **High speed**
  - —$f_{MAX}$ = 100 MHz
  - —$t_{PD}$ = 12 ns
  - —$t_S$ = 6 ns
  - —$t_{CO}$ = 7 ns
- **Electrically alterable FLASH technology**
- **Available in 160-pin TQFP, CQFP, and PGA packages**

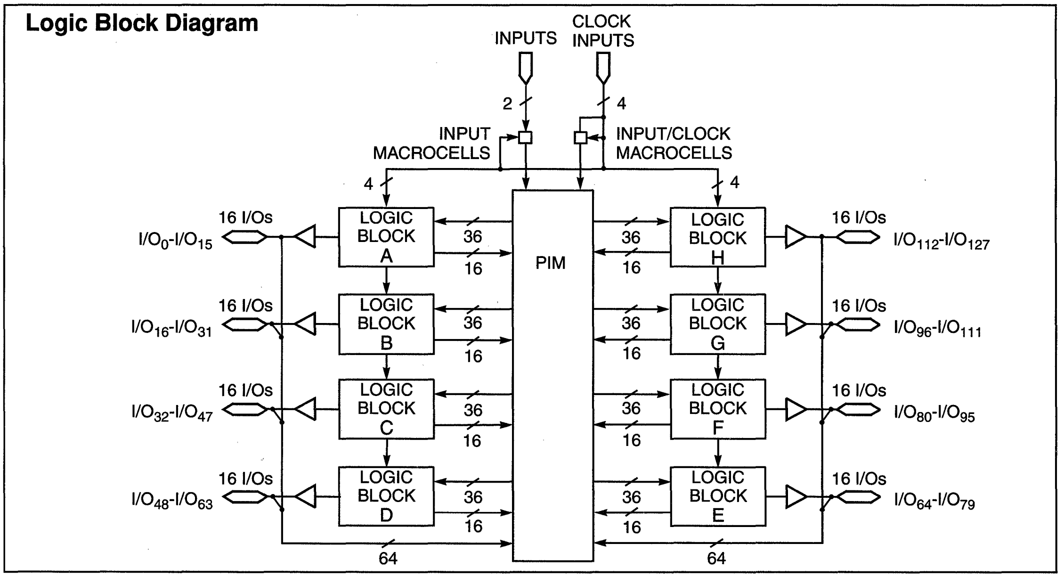
## Functional Description

The CY7C375 is a Flash erasable Complex Programmable Logic Device (CPLD) and is part of the FLASH370™ family of high-density, high-speed CPLDs. Like all members of the FLASH370 family, the CY7C375 is designed to bring the ease of use and high performance of the 22V10 to high-density PLDs.

The 128 macrocells in the CY7C375 are divided between eight logic blocks. Each logic block includes 16 macrocells, a 72 x 86 product term array, and an intelligent product term allocator.

The logic blocks in the FLASH370 architecture are connected with an extremely fast and predictable routing resource—the Programmable Interconnect Matrix (PIM). The PIM brings flexibility, routability, speed, and a uniform delay to the interconnect.

## Logic Block Diagram



## Selection Guide

| | | 7C375-100 | 7C375-83 | 7C375-66 | 7C375L-66 |
|---|---|---|---|---|---|
| Maximum Propagation Delay, $t_{PD}$ (ns) | | 12 | 15 | 20 | 20 |
| Minimum Set-Up, $t_S$ (ns) | | 6 | 8 | 10 | 10 |
| Maximum Clock to Output, $t_{CO}$ (ns) | | 7 | 8 | 10 | 10 |
| Maximum Supply Current, $I_{CC}$ (mA) | Commercial | 330 | 300 | 300 | 150 |
| | Military/Industrial | | 370 | 370 | |

Ultralogic and FLASH370 are trademarks of Cypress Semiconductor Corporation.

CYPRESS

# CY7C340 EPLD Family

# Multiple Array Matrix High-Density EPLDs

## Features

- **Erasable, user-configurable CMOS EPLDs capable of implementing high-density custom logic functions**
- **0.8-micron double-metal CMOS EPROM technology (CY7C34X)**
- **Advanced 0.65-micron CMOS technology to increase performance (CY7C34XB)**
- **Multiple Array MatriX architecture optimized for speed, density, and straightforward design implementation**
  - **—Programmable Interconnect Array (PIA) simplifies routing**
  - **—Flexible macrocells increase utilization**
  - **—Programmable clock control**
  - **—Expander product terms implement complex logic functions**
- ***Warp2*®**
  - **—Low-cost VHDL compiler for CPLDs and PLDs**
  - **—IEEE 1164-compliant VHDL**
  - **—Available on PC and Sun platforms**
- ***Warp3*®**
  - **—VHDL synthesis**
  - **—ViewLogic graphical user interface**
  - **—Schematic capture (ViewDraw™)**
  - **—VHDL simulation (ViewSim™)**
  - **—Available on PC and Sun platforms**

## General Description

The Cypress Multiple Array Matrix (MAX®) family of EPLDs provides a user-configurable, high-density solution to general-purpose logic integration requirements. With the combination of innovative architecture and state-of-the-art process, the MAX EPLDs offer LSI density without sacrificing speed.

The MAX architecture makes it ideal for replacing large amounts of TTL SSI and MSI logic. For example, a 74161 counter utilizes only 3% of the 128 macrocells available in the CY7C342B. Similarly, a 74151 8-to-1 multiplexer consumes less than 1% of the over 1,000 product terms in the CY7C342B. This allows the designer to replace 50 or more TTL packages with just one MAX EPLD. The family comes in a range of densities, shown below. By standardizing on a few MAX building blocks, the designer can replace hundreds of different 7400 series part numbers currently used in most digital systems.

The family is based on an architecture of flexible macrocells grouped together into Logic Array Blocks (LABs). Within the LAB is a group of additional product terms called expander product terms. These expanders are used and shared by the macrocells, allowing complex functions of up to 35 product terms to be easily implemented in a single macrocell. A Programmable Interconnect Array (PIA) globally routes all signals within devices containing more than one LAB. This architecture is fabricated on the Cypress 0.8-micron, double-layer-metal CMOS EPROM process, yielding devices with significantly higher integration, density and system clock speed than the largest of previous generation EPLDs. The CY7C34XB devices are 0.65-micron shrinks of the original 0.8-micron family. The CY7C34XBs offer faster speed bins for each device in the Cypress MAX family.

The density and performance of the CY7C340 family is accessed using Cypress's *Warp2* and *Warp3* design software. *Warp2* provides state-of-the-art VHDL synthesis for MAX and FLASH370™ at a very low cost. *Warp3* is a sophisticated CAE tool that includes schematic capture (ViewDraw) and timing simulation (ViewSim) in addition to VHDL synthesis. Consult the *Warp2* and *Warp3* datasheets for more information about the development tools.

3

## Max Family Members

| Feature | CY7C344(B) | CY7C343(B) | CY7C342B | CY7C346(B) | CY7C341B |
|---|---|---|---|---|---|
| Macrocells | 32 | 64 | 128 | 128 | 192 |
| MAX Flip-Flops | 32 | 64 | 128 | 128 | 192 |
| MAX Latches[1] | 64 | 128 | 256 | 256 | 384 |
| MAX Inputs[2] | 23 | 35 | 59 | 84 | 71 |
| MAX Outputs | 16 | 28 | 52 | 64 | 64 |
| Packages | 28H,J,W,P | 44H,J | 68H,J,R | 84H,J 100R,N | 84H,J,R |

Key: P—Plastic DIP; H—Windowed Ceramic Leaded Chip Carrier; J—Plastic J-Lead Chip Carrier; R—Windowed Pin Grid Array; W—Windowed Ceramic DIP; N—Plastic Quad Flat Pack

**Notes:**

[1] When all expander product terms are used to implement latches.

[2] With one output.

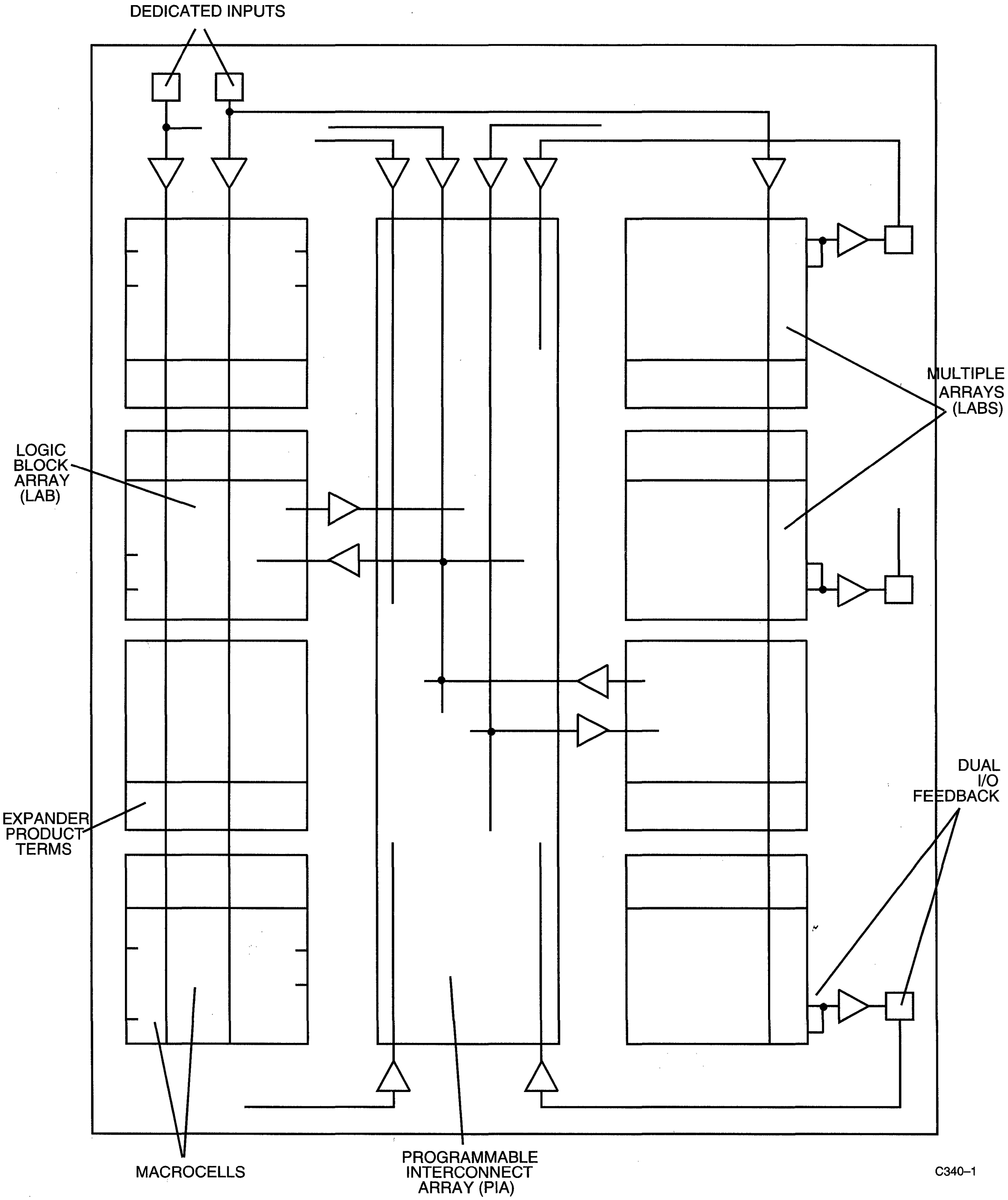


**Figure 1. Key MAX Features**

## Functional Description

### The Logic Array Block

The logic array block, shown in *Figure 2*, is the heart of the MAX architecture. It consists of a macrocell array, expander product term array, and an I/O block. The number of macrocells, expanders, and I/O vary, depending upon the device used. Global feedback of all signals is provided within a LAB, giving each functional block complete access to the LAB resources. The LAB itself is fed by the programmable interconnect array and dedicated input bus. The feedbacks of the macrocells and I/O pins feed the PIA, providing access to them through other LABs in the device. The members of the CY7C340 family of EPLDs that have a single LAB use a global bus, so a PIA is not needed (see *Figure 3*).

### The MAX Macrocell

Traditionally, PLDs have been divided into either PLA (programmable AND, programmable OR), or PAL® (programmable AND, fixed OR) architectures. PLDs of the latter type provide faster input-to-output delays, but can be inefficient due to fixed allocation of product terms. Statistical analysis of PLD logic designs has shown that 70% of all logic functions (per macrocell) require three product terms or less.

The macrocell structure of MAX has been optimized to handle variable product term requirements. As shown in *Figure 4*, each macrocell consists of a product term array and a configurable register. In the macrocell, combinatorial logic is implemented with three product terms ORed together, which then feeds an XOR gate. The second input to the XOR gate is also controlled by a product term, providing the ability to control active HIGH or active LOW logic and to implement T- and JK-type flip-flops.

If more product terms are required to implement a given function, they may be added to the macrocell from the expander product term array. These additional product terms may be added to any macrocell, allowing the designer to build gate-intensive logic, such as address decoders, adders, comparators, and complex state machines, without using extra macrocells.

The register within the macrocell may be programmed for either D, T, JK, or RS operation. It may alternately be configured as a flow-through latch for minimum input-to-output delays, or bypassed entirely for purely combinatorial logic. In addition, each register supports both asynchronous preset and clear, allowing asynchronous loading of counters of shift registers, as found in many standard TTL functions. These registers may be clocked with a synchronous system clock, or clocked independently from the logic array.

### Expander Product Terms

The expander product terms, as shown in *Figure 5*, are fed by the dedicated input bus, the programmable interconnect array, the macrocell feedback, the expanders themselves, and the I/O pin feedbacks. The outputs of the expanders then go to each and every product term in the macrocell array. This allows expanders to be “shared” by the product terms in the logic array block. One expander may feed all macrocells in the LAB, or even multiple product terms in the same macrocell. Since these expanders feed the secondary product terms (preset, clear, clock, and output enable) of each macrocell, complex logic functions may be implemented without utilizing another macrocell. Likewise, expanders may feed and be shared by other expanders, to implement complex multilevel logic and input latches.

3

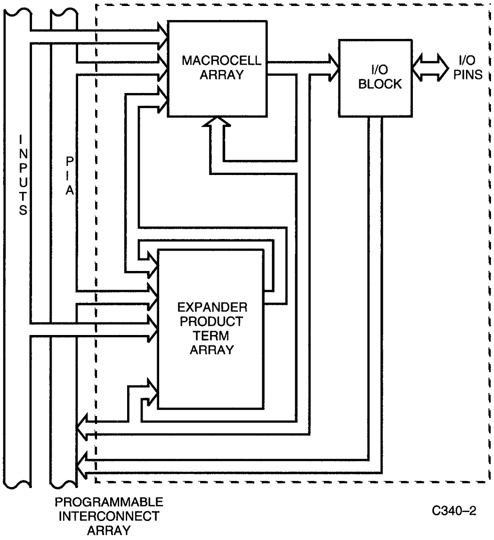


**Figure 2. Typical LAB Block Diagram**

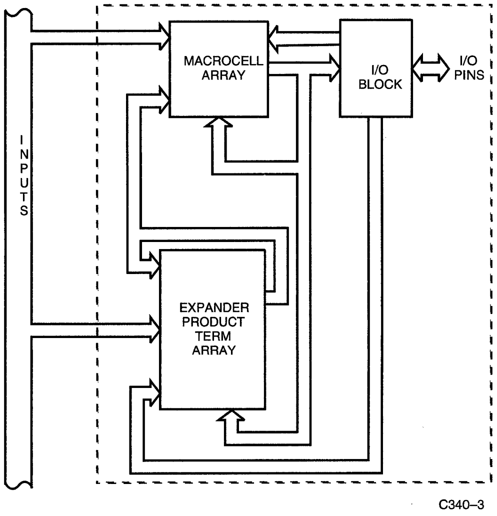
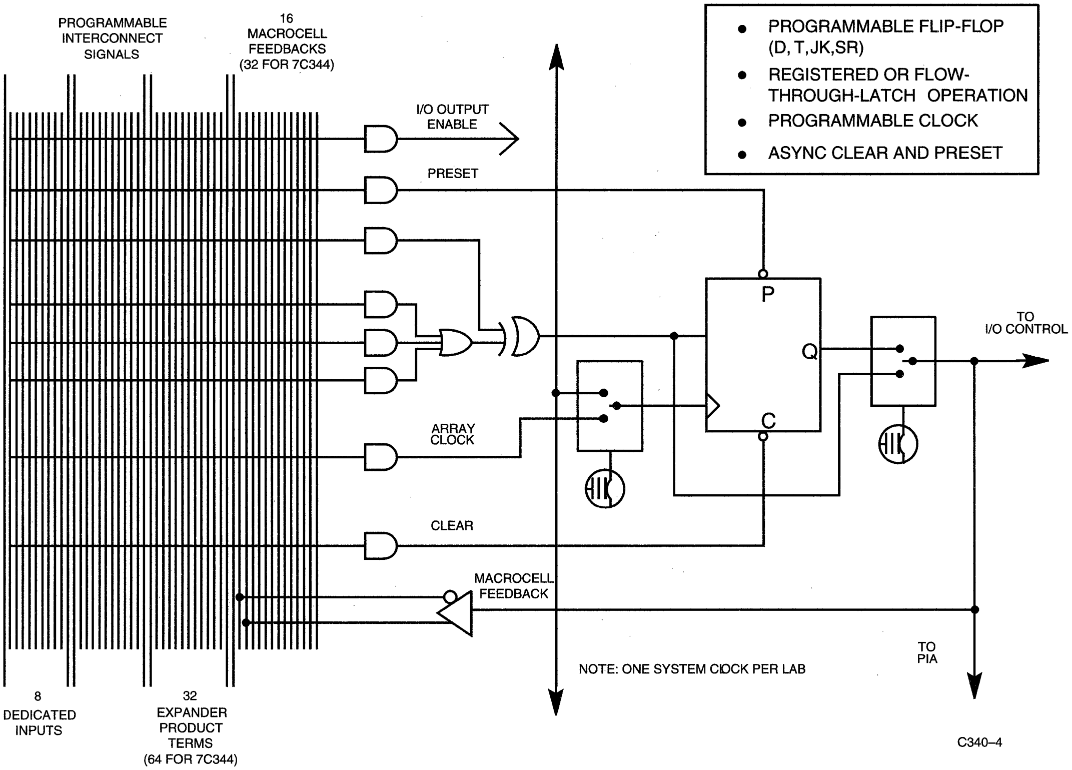


**Figure 3. 7C344 LAB Block Diagram**


PROGRAMMABLE INTERCONNECT SIGNALS
16 MACROCELL FEEDBACKS (32 FOR 7C344)
I/O OUTPUT ENABLE
PRESET
ARRAY CLOCK
CLEAR
MACROCELL FEEDBACK
P
Q
C
TO I/O CONTROL
TO PIA
NOTE: ONE SYSTEM CLOCK PER LAB
8 DEDICATED INPUTS
32 EXPANDER PRODUCT TERMS (64 FOR 7C344)
C340–4

- PROGRAMMABLE FLIP-FLOP (D, T, JK, SR)
- REGISTERED OR FLOW-THROUGH-LATCH OPERATION
- PROGRAMMABLE CLOCK
- ASYNC CLEAR AND PRESET

**Figure 4. Macrocell Block Diagram**

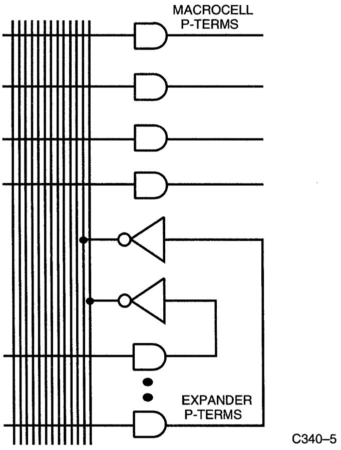


**Figure 5. Expander Product Terms**

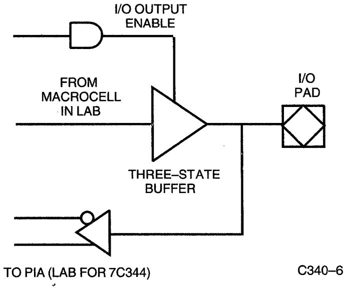


**Figure 6. I/O Block Diagram**

### I/O Block

Separate from the macrocell array is the I/O control block of the LAB. *Figure 6* shows the I/O block diagram. The three-state buffer is controlled by a macrocell product term and the drives the I/O pad. The input of this buffer comes from a macrocell within the associated LAB. The feedback path from the I/O pin may feed other blocks within the LAB, as well as the PIA. By decoupling the I/O pins from the flip-flops, all the registers in the LAB are "buried," allowing the I/O pins to be used as dedicated outputs, bidirectional outputs, or as additional dedicated inputs. Therefore, applications requiring many buried flip-flops, such as counters, shift registers, and state machines, no longer consume both the macrocell register and the associated I/O pin, as in earlier devices.

### The Programmable Interconnect Array

PLD density and speed has traditionally been limited by signal routing; i.e., getting signals from one macrocell to another. For smaller devices, a single array is used and all signals are available to all macrocells. But as the devices increase in density, the number of signals being routed becomes very large, increasing the amount of silicon used for interconnections. Also, because the signal must be global, the added loading on the internal connection path reduces

the overall speed performance of the device. The MAX architecture solves these problems. It is based on the concept of small, flexible logic array blocks that, in the larger devices, are interconnected by a PIA.

The PIA solves interconnect limitations by routing only the signals needed by each LAB. The architecture is designed so that every signal on the chip is within the PIA. The PIA is then programmed to give each LAB access to the signals that it requires. Consequently, each LAB receives only the signals needed. This effectively solves any routing problems that may arise in a design without degrading the performance of the device. Unlike masked or programmable gate arrays, which induce variable delays dependent on routing, the PIA has a fixed delay from point to point. This eliminates undesired skews among logic signals, which may cause glitches in internal or external logic.

## Development Software Support

### *Warp2*

*Warp2* is a state-of-the-art VHDL compiler for designing with Cypress PLDs and CPLDs. *Warp2* utilizes a proper subset of IEEE 1164 VHDL as its Hardware Description Language (HDL) for design entry. VHDL provides a number of significant benefits for the design entry process. *Warp2* accepts VHDL input, synthesizes and optimizes the entered design, and outputs a JEDEC map for the desired device. For functional simulation, *Warp2* provides a graphical waveform simulator (NOVA).

VHDL (VHSIC Hardware Description Language) is an open, powerful, non-proprietary language that is a standard for behavioral design entry and simulation. It is already mandated for use by the Department of Defense, and supported by every major vendor of CAE tools. VHDL allows designers to learn a single language that is useful for all facets of the design process.

### *Warp3*

*Warp3* is a sophisticated design tool that is based on the latest version of ViewLogic's CAE design environment. *Warp3* features schematic capture (ViewDraw), VHDL waveform simulation (ViewSim), a VHDL debugger, and VHDL synthesis, all integrated in a graphical design environment. *Warp3* is available on PCs using Windows 3.1 or subsequent versions, and on Sun and HP workstations.

For further information on Warp software, see the *Warp2* and *Warp3* datasheets contained in this data book.

### Third-Party Software

Cypress maintains a very strong commitment to third-party design software vendors. All major third-party software vendors provide support for the MAX family of devices. To expedite this support, Cypress supplies vendors with all pertinent architectural information as well as design fitters for our products.

## Programming

The *Impulse3*™ device programmers from Cypress will program all Cypress PLDs, CPLDs, and PROMs. The unit is a standalone programmer that connects to any IBM-compatible PC via the printer port.

### Third-Party Programmers

As with development software, Cypress strongly supports third-party programmers. All major third-party programmers support the MAX family.

## Cross Reference

| ALTERA | CYPRESS |
|---|---|
| PREFIX EPM | PREFIX: CY |
| PREFIX: EP | PREFIX: PALC |
| 22V10–10C | PALC22V10D–7C |
| 22V10–10C | PALC22V10D–10C |
| 22V10–10C | PAL22V10C–7C+ |
| 22V10–10C | PAL22V10C–10C+ |
| 22V10–15C | PALC22V10B–15C |
| 22V10–15C | PALC22V10D–15C |
| 5032DC | 7C344–25WC |
| 5032DC–2 | 7C344–20WC |
| 5032DC–15 | 7C344–15WC |
| 5032DC–17 | Call Factory |
| 5032DC–20 | 7C344–20WC |
| 5032DC–25 | 7C344–25WC |
| 5032DM | 7C344–25WMB |
| 5032DM–25 | 7C344–25WMB |
| 5032JC | 7C344–25HC |
| 5032JC–2 | 7C344–20HC |
| 5032JC–15 | 7C344–15HC |
| 5032JC–17 | Call Factory |
| 5032JC–20 | 7C344–20HC |
| 5032JC–25 | 7C344–25HC |

**Cross Reference** (continued)

| ALTERA | CYPRESS |
|---|---|
| 5032JM | 7C344–25HMB |
| 5032JM–25 | 7C344–25HMB |
| 5032LC | 7C344–25JC |
| 5032LC–2 | 7C344–20JC |
| 5032LC–15 | 7C344–15JC |
| 5032LC–17 | Call Factory |
| 5032LC–20 | 7C344–20JC |
| 5032LC–25 | 7C344–25JC |
| 5032PC | 7C344–25PC |
| 5032PC–2 | 7C344–20PC |
| 5032PC–15 | 7C344–15PC |
| 5032PC–17 | Call Factory |
| 5032PC–20 | 7C344–20PC |
| 5032PC–25 | 7C344–25PC |
| 5064JC | 7C343–35HC |
| 5064JC–1 | 7C343–25HC |
| 5064JC–2 | 7C343–30HC |
| 5064JI | 7C343–35HI |
| 5064JM | 7C343–35HMB |
| 5064LC | 7C343–35JC |
| 5064LC–1 | 7C343–25JC |
| 5064LC–2 | 7C343–30JC |
| 5128AGC–12 | 7C342B–12RC |
| 5128AGC–15 | 7C342B–15RC |
| 5128AGC–20 | 7C342B–20RC |
| 5128AJC–12 | 7C342B–12HC |
| 5128AJC–15 | 7C342B–15HC |
| 5128AJC–20 | 7C342B–20HC |
| 5128ALC–12 | 7C342B–12JC |
| 5128ALC–15 | 7C342B–15JC |
| 5128ALC–20 | 7C342B–20JC |
| 5128GC | 7C342–35RC |
| 5128GC–1 | 7C342–25RC |
| 5128GC–2 | 7C342–30RC |
| 5128GM | 7C342–35RMB |
| 5128JC | 7C342–35HC |
| 5128JC–1 | 7C342–25HC |
| 5128JC–2 | 7C342–30HC |
| 5128J! | 7C342–35HI |
| 5128JI–2 | 7C342–30HI |
| 5128JM | 7C342–35HMB |
| 5128LC | 7C342–35JC |
| 5128LC–1 | 7C342–25JC |
| 5128LC–2 | 7C342–30JC |

**Cross Reference** (continued)

| ALTERA | CYPRESS |
|---|---|
| 5128LI | 7C342–35JI |
| 5128LI–2 | 7C342–30HI |
| 5130GC | 7C346–35RC |
| 5130GC–1 | 7C346–25RC |
| 5130GC–2 | 7C346–30RC |
| 5130GM | 7C346–35RM |
| 5130JC | 7C346–35HC |
| 5130JC–1 | 7C346–25HC |
| 5130JC–2 | 7C346–30HC |
| 5130JM | 7C346–35HM |
| 5130LC | 7C346–35JC |
| 5130LC–1 | 7C346–25JC |
| 5130LC–2 | 7C346–30JC |
| 5130LI | 7C346–35JI |
| 5130LI–2 | 7C346–30JI |
| 5130QC | 7C346–35NC |
| 5130QC–1 | 7C346–25NC |
| 5130QC–2 | 7C346–30NC |
| 5130QI | 7C346–35NI |
| 5192AGC–15 | 7C341B–15RC |
| 5192AGC–20 | 7C341B–20RC |
| 5192AJC–15 | 7C341B–15HC |
| 5192AJC–20 | 7C341B–20HC |
| 5192ALC–1 | 7C341B–15JC |
| 5192ALC–2 | 7C341B–20JC |
| 5192GC | 7C341–35RC |
| 5192GC–1 | 7C341–25RC |
| 5192GC–2 | 7C341–30RC |
| 5192JM | 7C341–35HM |
| 5192JC | 7C341–35HC |
| 5192JC–1 | 7C341–25HC |
| 5192JC–2 | 7C341–30HC |
| 5192GM | 7C341–35RM |
| 5192JI | 7C341–35HI |
| 5192LC | 7C341–35JC |
| 5192LC–1 | 7C341–25JC |
| 5192LC–2 | 7C341–30JC |

Document #: 38-00087-D

# CY7C341B

# 192-Macrocell MAX® EPLD

## Features

- **192 macrocells in 12 LABs**
- **8 dedicated inputs, 64 bidirectional I/O pin**
- **Advanced 0.65-micron CMOS technology to increase performance**
- **Programmable interconnect array**
- **384 expander product terms**
- **Available in 84-pin HLCC, PLCC, and PGA packages**

## Functional Description

The CY7C341B is an Erasable Programmable Logic Device (EPLD) in which CMOS EPROM cells are used to configure logic functions within the device. The MAX architecture is 100% user configurable allowing the devices to accommodate a variety of independent logic functions.

The 192 macrocells in the CY7C341B is divided into 12 Logic Array Blocks (LABs), 16 per LAB. There are 384 expander product terms, 32 per LAB, to be used and shared by the macrocells within each LAB. Each LAB is interconnected with a programmable interconnect array, allowing all signals to be routed throughout the chip.

The speed and density of the CY7C341B allows them to be used in a wide range of applications, from replacement of large amounts of 7400 series TTL logic, to complex controllers and multifunction chips. With greater than 37 times the functionality of 20-pin PLDs, the CY7C341B allows the replacement of over 75 TTL devices. By replacing large amounts of logic, the CY7C341B reduces board space, part count, and increases system reliability.

Each LAB contains 16 macrocells. In LABs A, F, G, and L, 8 macrocells are connected to I/O pins and 8 are buried, while for LABs B, C, D, E, H, I, J, and K, 4 macrocells are connected to I/O pins and 12 are buried. Moreover, in addition to the I/O and buried macrocells, there are 32 single product term logic expanders in each LAB. Their use greatly enhances the capability of the macrocells without increasing the number of product terms in each macrocell.

## Logic Array Blocks

There are 12 logic array blocks in the CY7C341B. Each LAB consists of a macrocell array containing 16 macrocells, an expander product term array containing 32 expanders, and an I/O block. The LAB is fed by the programmable interconnect array and the dedicated input bus. All macrocell feedbacks go to the macrocell array, the expander array, and the programmable interconnect array. Expanders feed themselves and the macrocell array. All I/O feedbacks go to the programmable interconnect array so that they may be accessed by macrocells in other LABs as well as the macrocells in the LAB in which they are situated.

Externally, the CY7C341B provide 8 dedicated inputs, one of which may be used as a system clock. There are 64 I/O pins that may be individually configured for input, output, or bidirectional data flow.

## Programmable Interconnect Array

The Programmable Interconnect Array (PIA) solves interconnect limitations by routing only the signals needed by each logic array block. The inputs to the PIA are the outputs of every macrocell within the device and the I/O pin feedback of every pin on the device.

Unlike masked or programmable gate arrays, which induce variable delay dependent on routing, the PIA has a fixed delay. This eliminates undesired skews among logic signals, which may cause glitches in internal or external logic. The fixed delay, regardless of programmable interconnect array configuration, simplifies design by assuring that internal signal skews or races are avoided. The result is ease of design implementation, often in a single pass, without the multiple internal logic placement and routing iterations required for a programmable gate array to achieve design timing objectives.

## Timing Delays

Timing delays within the CY7C341B may be easily determined using *Warp2*®, or *Warp3*® software or by the model shown in *Figure 1*. The CY7C341B has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information, the *Warp3* software provides a timing simulator.

## Design Recommendations

For proper operation, input and output pins must be constrained to the range GND ≤ (VIN or VOUT) ≤ VCC. Unused inputs must always be tied to an appropriate logic level (either VCC or GND). Each set of VCC and GND pins must be connected together directly at the device. Power supply decoupling capacitors of at least 0.2 µF must be connected between VCC and GND. For the most effective decoupling, each VCC pin should be separately decoupled to GND, directly at the device. Decoupling capacitors should have good frequency response, such as monolithic ceramic types.

## Design Security

The CY7C341B contains a programmable design security feature that controls the access to the data programmed into the device. If this programmable feature is used, a proprietary design implemented in the device cannot be copied or retrieved. This enables a high level of design control to be obtained since programmed data within EPROM cells is invisible. The bit that controls this function, along with all other program data, may be reset simply by erasing the device.

## Selection Guide

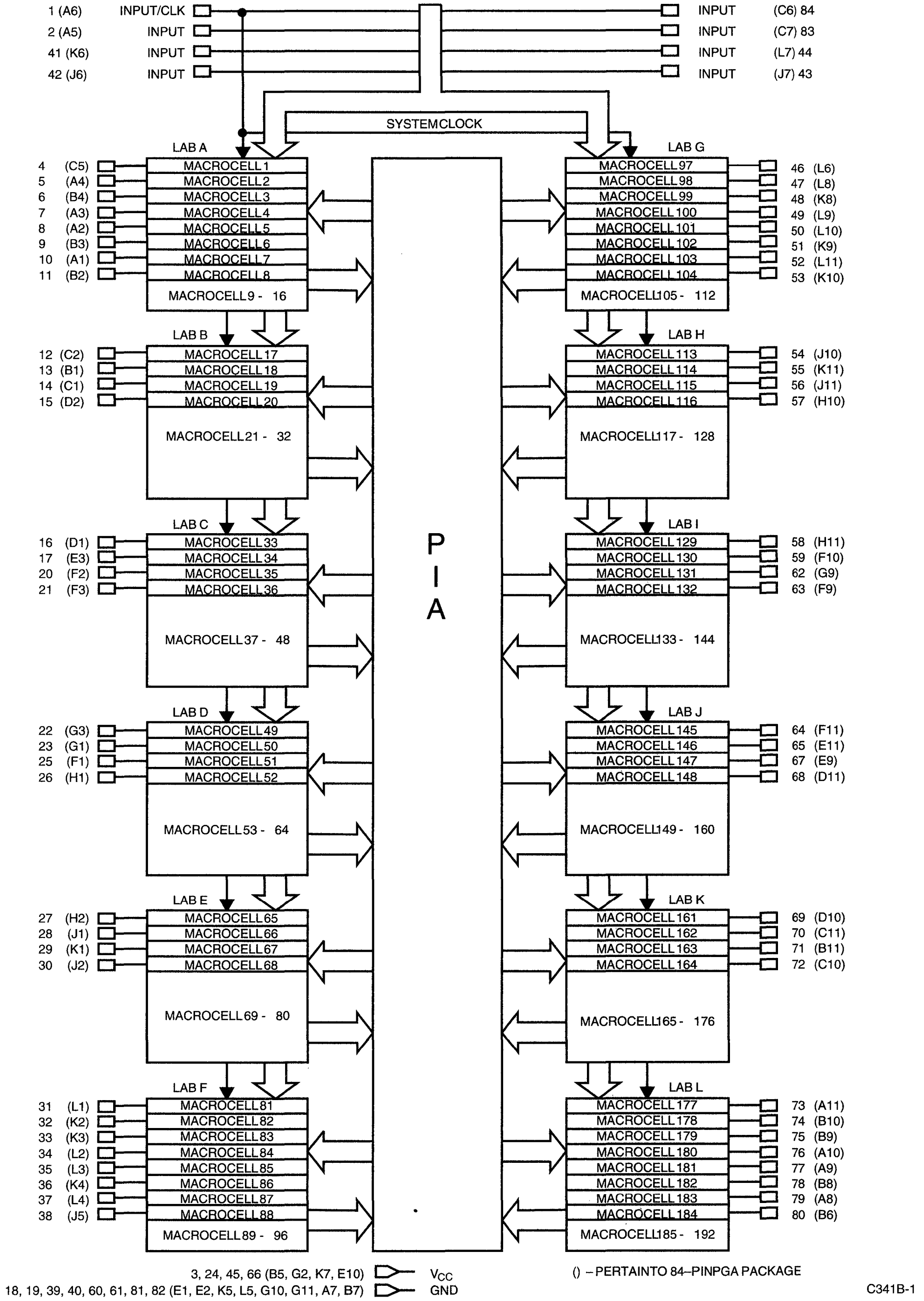
| | | 7C341B-15 | 7C341B-20 | 7C341B-25 | 7C341B-30 | 7C341B-35 |
|---|---|---|---|---|---|---|
| Maximum Access Time (ns) | | 15 | 20 | 25 | 30 | 35 |
| Maximum Operating Current (mA) | Commercial | 380 | 380 | 380 | 380 | 380 |
| | Industrial | 480 | 480 | 480 | 480 | 480 |
| | Military | | 480 | 480 | 480 | 480 |
| Maximum Standby Current (mA) | Commercial | 360 | 360 | 360 | 360 | 360 |
| | Industrial | 435 | 435 | 435 | 435 | 435 |
| | Military | | 435 | 435 | 435 | 435 |

Shaded area contains preliminary information.

MAX is a registered trademark of Altera Corporation.
*Warp* is a trademark of Cypress Semiconductor Corporation.

*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

## Logic Block Diagram

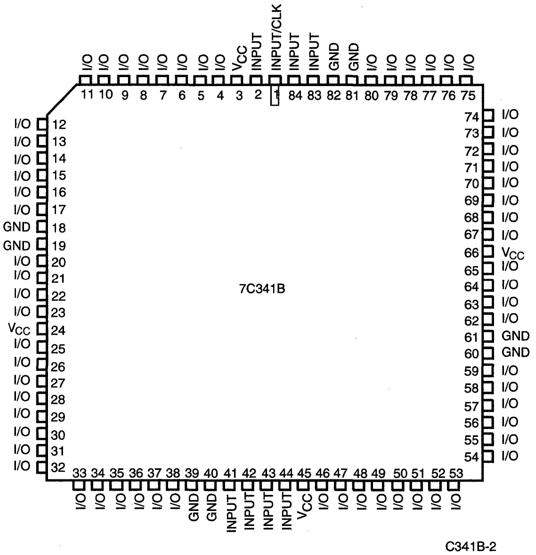


() – PERTAIN TO 84–PIN PGA PACKAGE

C341B-1

## Pin Configurations

PLCC/HLCC
Top View



C341B-2

PGA
BottomView

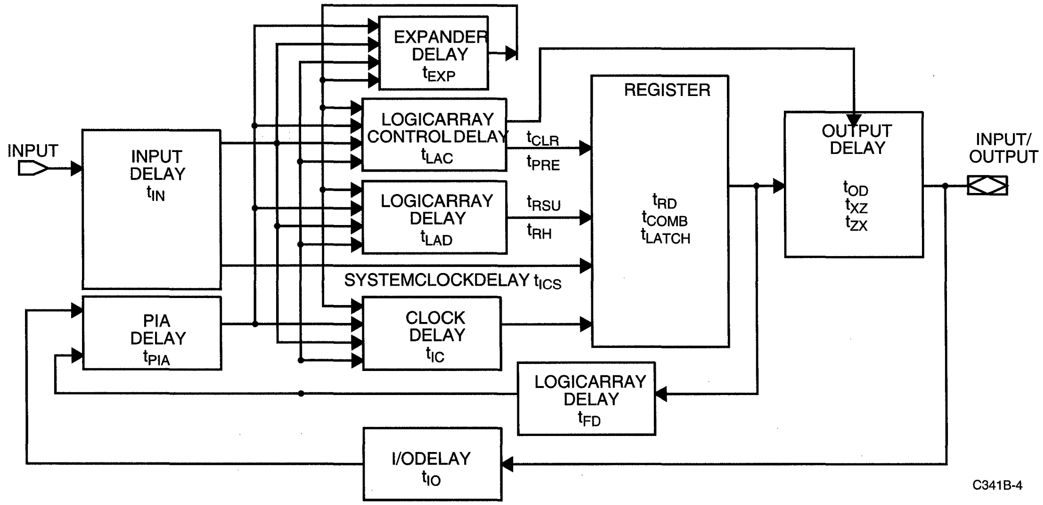
| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| L | I/O | I/O | I/O | I/O | GND | I/O | INPUT | I/O | I/O | I/O | I/O |
| K | I/O | I/O | I/O | I/O | GND | INPUT | $V_{CC}$ | I/O | I/O | I/O | I/O |
| J | I/O | I/O | | | I/O | INPUT | INPUT | | | I/O | I/O |
| H | I/O | I/O | | | | | | | | I/O | I/O |
| G | I/O | $V_{CC}$ | I/O | | | | | | I/O | GND | GND |
| F | I/O | I/O | I/O | | | 7C341B | | | I/O | I/O | I/O |
| E | GND | GND | I/O | | | | | | I/O | $V_{CC}$ | I/O |
| D | I/O | I/O | | | | | | | | I/O | I/O |
| C | I/O | I/O | ■ | | I/O | INPUT | INPUT | | | I/O | I/O |
| B | I/O | I/O | I/O | I/O | $V_{CC}$ | I/O | GND | I/O | I/O | I/O | I/O |
| A | I/O | I/O | I/O | I/O | INPUT | INPUT/ CLK | GND | I/O | I/O | I/O | I/O |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

C341B-3

## Design Security (continued)

The CY7C341B is fully functionally tested and guaranteed through complete testing of each programmable EPROM bit and all internal logic elements thus ensuring 100% programming yield.

The erasable nature of these devices allows test programs to be used and erased during early stages of the production flow. The devices also contain on-board logic test circuitry to allow verification of function and AC specification once encapsulated in non-windowed packages.



C341B-4

**Figure 1. CY7C341B Internal Timing Model**

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ....................................–65°C to +150°C

Ambient Temperature with
Power Applied .................................................. 0°C to +70°C

Maximum Junction Temperature
(Under Bias)............................................................. 150°C

Supply Voltage to Ground Potential.................–2.0V to +7.0V

Maximum Power Dissipation..................................2500 mW

DC $V_{CC}$ or GND Current....................................................500 mA

DC Output Current, per Pin .......................–25 mA to +25 mA

DC Input Voltage[1] ..............................................–3.0V to +7.0V

DC Program Voltage.....................................................13.0V

Static Discharge Voltage........................................... >1100V
(per MIL-STD-883, method 3015)

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | –40°C to +85°C | 5V ± 10% |
| Military | –55°C to +125°C (Case) | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $I_{OH}$ = –4.0 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $I_{OL}$ = 8 mA | | | 0.45 | V |
| $V_{IH}$ | Input HIGH Level | | | 2.2 | $V_{CC}$+0.3 | V |
| $V_{IL}$ | Input LOW Level | | | –0.3 | 0.8 | V |
| $I_{IX}$ | Input Current | GND ≤ $V_{IN}$ ≤ $V_{CC}$ | | –10 | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = $V_{CC}$ or GND | | –40 | +40 | µA |
| $I_{OS}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = GND[3,4] | | –30 | –90 | mA |
| $I_{CC1}$ | Power Supply Current (Standby) | $V_I$ = $V_{CC}$ or GND (No Load) | Com'l | | 360 | mA |
| | | | Mil/Ind | | 435 | mA |
| $I_{CC2}$ | Power Supply Current[5] | $V_I$ = $V_{CC}$ or GND (No Load) f = 1.0 MHz[3, 5] | Com'l | | 380 | mA |
| | | | Mil/Ind | | 480 | mA |
| $t_R$ (Recommended) | Input Rise Time | | | | 100 | ns |
| $t_F$ (Recommended) | Input Fall Time | | | | 100 | ns |

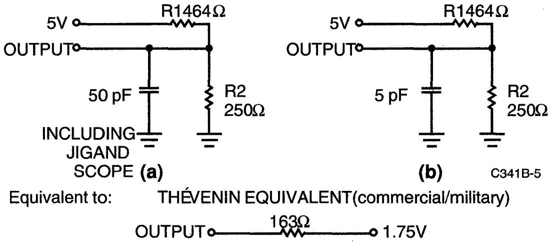
## Capacitance[6]

| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $T_A$ = 25°C, f = 1 MHz, $V_{CC}$ = 5.0V | 10 | pF |
| $C_{OUT}$ | Output Capacitance | | 20 | pF |

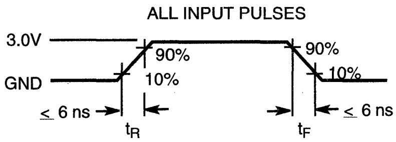
**Notes:**

1. Minimum DC input is -0.3V. During transitions, the inputs may undershoot to -2.0V for periods less than 20 ns.
2. Typical values are for $T_A$ = 25°C and $V_{CC}$ = 5V.
3. Guaranteed but not 100% tested.
4. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
5. This parameter is measured with device programmed as a 16-bit counter in each LAB and is tested periodically by sampling production material.
6. Part (a) in AC Test Load and Waveforms is used for all parameters except $t_{ER}$ and $t_{XZ}$, which is used for part (b) in AC Test Load and Waveforms. All external timing parameters are measured referenced to external pins of the device.

## AC Test Loads and Waveforms



C341B-5



C341B-6

**External Synchronous Switching Characteristics** Over the Operating Range[6]

| Parameter | Description | | 7C341B-15 | | 7C341B-20 | | 7C341B-25 | | 7C341B-30 | | 7C341B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max | Min. | Max | Min. | Max | Min. | Max | Min. | Max | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | Com'l | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| | | Mil | | | | 20 | | 25 | | 30 | | 35 | |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | Com'l | | 25 | | 33 | | 40 | | 45 | | 55 | ns |
| | | Mil | | | | 33 | | 40 | | 45 | | 55 | |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | Com'l | | 23 | | 30 | | 37 | | 44 | | 55 | ns |
| | | Mil | | | | 30 | | 37 | | 44 | | 55 | |
| $t_{PD4}$ | I/O Input to Combinatoria l Output Delay with Expander Delay[3,10] | Com'l | | 33 | | 43 | | 52 | | 59 | | 75 | ns |
| | | Mil | | | | 43 | | 52 | | 59 | | 75 | |
| $t_{EA}$ | Input to Output Enable Delay[3, 7] | Com'l | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| | | Mil | | | | 20 | | 25 | | 30 | | 35 | |
| $t_{ER}$ | Input to Output Disable Delay[6] | Com'l | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| | | Mil | | | | 20 | | 25 | | 30 | | 35 | |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | Com'l | | 7 | | 8 | | 14 | | 16 | | 20 | ns |
| | | Mil | | | | 8 | | 14 | | 16 | | 20 | |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[3,11] | Com'l | | 17 | | 20 | | 30 | | 35 | | 42 | ns |
| | | Mil | | | | 20 | | 30 | | 35 | | 42 | |
| $t_{S1}$ | Dedicated Input or Feedback Set-up Time to Synchronous Clock Output[6,12] | Com'l | 10 | | 13 | | 15 | | 20 | | 25 | | ns |
| | | Mil | | | 13 | | 15 | | 20 | | 25 | | |
| $t_{S2}$ | I/O Input Set-up Time to Synchronous Clock Input[8] | Com'l | 20 | | 24 | | 30 | | 39 | | 45 | | ns |
| | | Mil | | | 24 | | 30 | | 39 | | 45 | | |
| $t_H$ | Input Hold Time from Synchronous Clock Input[6] | Com'l | 0 | | 0 | | 0 | | 0 | | 0 | | ns |
| | | Mil | | | 0 | | 0 | | 0 | | 0 | | |
| $t_{WH}$ | Synchronous Clock Input High Time | Com'l | 5 | | 7 | | 8 | | 10 | | 12.5 | | ns |
| | | Mil | | | 7 | | 8 | | 10 | | 12.5 | | |
| $t_{WL}$ | Synchronous Clock Input Low Time | Com'l | 5 | | 7 | | 8 | | 10 | | 12.5 | | ns |
| | | Mil | | | 7 | | 8 | | 10 | | 12.5 | | |
| $t_{RW}$ | Asynchronous Clear Width[3,6] | Com'l | 16 | | 22 | | 25 | | 30 | | 35 | | ns |
| | | Mil | | | 22 | | 25 | | 30 | | 35 | | |

Shaded area contains preliminary information.

**External Synchronous Switching Characteristics** Over the Operating Range[6] (continued)

| | | | 7C341B-15 | | 7C341B-20 | | 7C341B-25 | | 7C341B-30 | | 7C341B-35 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | | **Min.** | **Max** | **Min.** | **Max** | **Min.** | **Max** | **Min.** | **Max** | **Min.** | **Max** | **Unit** |
| $t_{RR}$ | Asynchronous Clear Recovery[3,7] | Com'l | 16 | | 22 | | 25 | | 30 | | 35 | | ns |
| | | Mil | | | 22 | | 25 | | 30 | | 35 | | |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[5] | Com'l | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| | | Mil | | | | 20 | | 25 | | 30 | | 35 | |
| $t_{PW}$ | Asynchronous Preset Width[3, 6] | Com'l | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| | | Mil | | | 20 | | 25 | | 30 | | 35 | | |
| $t_{PR}$ | Asynchronous Preset Recovery Time[3, 6] | Com'l | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| | | Mil | | | | | 25 | | 30 | | 35 | | |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[6] | Com'l | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| | | Mil | | | | 20 | | 25 | | 30 | | 35 | |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[3,13] | Com'l | | 3 | | 3 | | 3 | | 3 | | 5 | ns |
| | | Mil | | | | 3 | | 3 | | 3 | | 5 | |
| $t_P$ | External Synchronous Clock Period ($1/f_{MAX3}$)[3] | Com'l | 12 | | 14 | | 16 | | 20 | | 25 | | ns |
| | | Mil | | | 14 | | 16 | | 20 | | 25 | | |
| $f_{MAX1}$ | External Feedback Maximum Frequency ($1/(t_{CO1} + t_{S1})$)[3,14] | Com'l | 58.8 | | 50 | | 34.5 | | 27.7 | | 22.2 | | MHz |
| | | Mil | | | 50 | | 34.5 | | 27.7 | | 22.2 | | |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of ($1/(t_{S1} + t_{CF})$) or ($1/t_{CO1}$)[3,15] | Com'l | 76.9 | | 62.5 | | 55.5 | | 43 | | 33 | | MHz |
| | | Mil | | | 62.5 | | 55.5 | | 43 | | 33 | | |
| $f_{MAX3}$ | Data Path Maximum Frequency, least of $1/(t_{WL} + t_{WH})$, $1/(t_{S1} + t_H)$, or ($1/t_{CO1}$)[3,16] | Com'l | 100 | | 71.4 | | 62.5 | | 50 | | 40.0 | | MHz |
| | | Mil | | | 71.4 | | 62.5 | | 50 | | 40.0 | | |
| $f_{MAX4}$ | Maximum Register Toggle Frequency ($1/(t_{WL} + t_{WH})$)[3,17] | Com'l | 100 | | 71.4 | | 62.5 | | 50 | | 40.0 | | MHz |
| | | Mil | | | 71.4 | | 62.5 | | 50 | | 40.0 | | |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[3,18] | Com'l | 3 | | 3 | | 3 | | 3 | | 3 | | ns |
| | | Mil | | | 3 | | 3 | | 3 | | 3 | | |

Shaded area contains preliminary information.

**Notes:**

7. This specification is a measure of the delay from input signal applied to a dedicated input to combinatorial output on any output pin. This delay assumes no expander terms are used to form the logic function.
When this note is applied to any parameter specification it indicates that the signal (data, asynchronous clock, asynchronous clear, and/or asynchronous preset) is applied to a dedicated input only and no signal path (either clock or data) employs expander logic.
If an input signal is applied to an I/O pin an additional delay equal to $t_{PIA}$ should be added to the comparable delay for a dedicated input. If expanders are used, add the maximum expander delay $t_{EXP}$ to the overall delay for the comparable delay without expanders.
8. This specification is a measure of the delay from input signal applied to an I/O macrocell pin to any output. This delay assumes no expander terms are used to form the logic function.
9. This specification is a measure of the delay from an input signal applied to a dedicated input to combinatorial output on any output pin. This delay assumes expander terms are used to form the logic functions and includes the worst-case expander logic delay for one pass through the expander logic.
10. This specification is a measure of the delay from an input signal applied to an I/O macrocell pin to any output. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.
11. This specification is a measure of the delay from synchronous register clock to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used, register is synchronously clocked and all feedback is within the same LAB. This parameter is tested periodically by sampling production material.
12. If data is applied to an I/O input for capture by a macrocell register, the I/O pin set-up time minimums should be observed. These parameters are $t_{S2}$ for synchronous operation and $t_{AS2}$ for asynchronous operation.
13. This specification is a measure of the delay associated with the internal register feedback path. This is the delay from synchronous clock to LAB logic array input. This delay plus the register set-up time, $t_{S1}$, is the minimum internal period for an internal synchronous state machine configuration. This delay is for feedback within the same LAB. This parameter is tested periodically by sampling production material.
14. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which a state machine configuration with external feedback can operate. It is assumed that all data inputs and feedback signals are applied to dedicated inputs. All feedback is assumed to be local originating within the same LAB.

**External Synchronous Switching Characteristics** Over the Operating Range[6] (continued)

| Parameter | Description | | 7C341B-15 | | 7C341B-20 | | 7C341B-25 | | 7C341B-30 | | 7C341B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max | Min. | Max | Min. | Max | Min. | Max | Min. | Max | |
| $t_{ACO1}$ | Dedicated Asynchronous Clock Input to Output Delay[6] | Com'l | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| | | Mil | | | | 20 | | 25 | | 30 | | 35 | |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output [19] | Com'l | | 25 | | 32 | | 40 | | 46 | | 55 | ns |
| | | Mil | | | | 32 | | 40 | | 46 | | 55 | |
| $t_{AS1}$ | Dedicated Input or Feedback Set-up Time to Asynchronous Clock Input[6] | Com'l | 5 | | 5 | | 5 | | 6 | | 8 | | ns |
| | | Mil | | | 5 | | 5 | | 6 | | 8 | | |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[6] | Com'l | 14 | | 18 | | 20 | | 27 | | 30 | | ns |
| | | Mil | | | 18 | | 20 | | 27 | | 30 | | |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[6] | Com'l | 5 | | 6 | | 6 | | 8 | | 10 | | ns |
| | | Mil | | | 6 | | 6 | | 8 | | 10 | | |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[6] | Com'l | 9 | | 10 | | 11 | | 14 | | 16 | | ns |
| | | Mil | | | 10 | | 11 | | 14 | | 16 | | |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[6, 20] | Com'l | 7 | | 8 | | 9 | | 11 | | 14 | | ns |
| | | Mil | | | 8 | | 9 | | 11 | | 14 | | |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[21] | Com'l | | 11 | | 13 | | 15 | | 18 | | 22 | ns |
| | | Mil | | | | 13 | | 15 | | 18 | | 22 | |
| $t_{AP}$ | External Asynchronous Clock Period ($1/f_{MAX4}$) | Com'l | 16 | | 18 | | 20 | | 25 | | 30 | | ns |
| | | Mil | | | 18 | | 20 | | 25 | | 30 | | |
| $f_{MAXA1}$ | External Feedback Maximum Frequency in Asynchronous Mode $1/(t_{ACO1} + t_{AS1})$[22] | Com'l | 50 | | 40 | | 33.3 | | 27 | | 23 | | MHz |
| | | Mil | | | 40 | | 33.3 | | 27 | | 23 | | |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[23] | Com'l | 62.5 | | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| | | Mil | | | 55.5 | | 50 | | 40 | | 33.3 | | |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[24] | Com'l | 62.5 | | 50 | | 40 | | 33.3 | | 28.5 | | MHz |
| | | | | | | | | | | | | | |

Shaded area contains preliminary information.

**Notes:**

15. This specification indicates the guaranteed maximum frequency at which a state machine, with internal-only feedback, can operate. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{CO1}$.
16. This frequency indicates the maximum frequency at which the device may operate in data path mode (dedicated input pin to output pin). This assumes data input signals are applied to dedicated input pins and no expander logic is used. If any of the data inputs are I/O pins, $t_{S2}$ is the appropriate $t_S$ for calculation.
17. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which an individual output or buried register can be cycle by a clock signal applied to the dedicated clock input pin.
18. This parameter indicates the minimum time after a synchronous register clock input that the previous register output data is maintained on the output pin.
19. This specification is a measure of the delay from an asynchronous register clock input to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used in the logic of combinatorial output or the asynchronous clock input. The clock signal is applied to the dedicated clock input pin and all feedback is within a single LAB. This parameter is tested periodically by sampling production material.
20. This parameter is measured with a positive-edge-triggered clock at the register. For negative-edge triggering, the $t_{AWH}$ and $t_{AWL}$ parameters must be swapped. If a given input is used to clock multiple registers with both positive and negative polarity, $t_{AWH}$ should be used for both $t_{AWH}$ and $t_{AWL}$.
21. This specification is a measure of the delay associated with the internal register feedback path for an asynchronous clock to LAB logic array input. This delay plus the asynchronous register set-up time, $t_{AS1}$, is the minimum internal period for an internal asynchronously clocked state machine configuration. This delay is for feedback within the same LAB, and assumes there is no expander logic in the clock path and the clock input signal is applied to a dedicated input pin. This parameter is tested periodically by sampling production material.
22. This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine configuration with external feedback can operate. It is assumed that all data inputs, clock inputs, and feedback signals are applied to dedicated inputs, and that no expander logic is employed in the clock signal path or data path.
23. This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine with internal-only feedback can operate. This parameter is determined by the lesser of $(1/t_{ACF} + t_{AS1}))$ or $(1/(t_{AWH} + t_{AWL}))$. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{ACO1}$.
24. This frequency is the maximum frequency at which the device may operate in the asynchronously clocked data path mode. This specification is determined by the least of $1/(t_{AWH} + t_{AWL})$, $1/(t_{AS1} + t_{AH})$ or $1/t_{ACO1}$. It assumes data and clock input signals are applied to dedicated input pins and no expander loigc is used.
25. This specification indicates the guaranteed maximum frequency at which an individual output or buried register can be cycled in asynchronously clocked mode by a clock signal applied to an external dedicated input pin.

**External Synchronous Switching Characteristics** Over the Operating Range[6] (continued)

| Parameter | Description | | 7C341B-15 | | 7C341B-20 | | 7C341B-25 | | 7C341B-30 | | 7C341B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max | Min. | Max | Min. | Max | Min. | Max | Min. | Max | |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[25] | Com'l | 62.5 | | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| | | Mil | | | 55.5 | | 50 | | 40 | | 33.3 | | |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[26] | Com'l | 15 | | 15 | | 15 | | 15 | | 15 | | ns |
| | | Mil | | | 15 | | 15 | | 15 | | 15 | | |

## Internal Switching Characteristics Over the Operating Range[2]

| Parameter | Description | | 7C341B-15 | | 7C341B-20 | | 7C341B-25 | | 7C341B-30 | | 7C341B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max | Min. | Max | Min. | Max | Min. | Max | Min. | Max | |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | Com'l | | 3 | | 4 | | 5 | | 7 | | 9 | ns |
| | | Mil | | | | 4 | | 5 | | 7 | | 9 | |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | Com'l | | 3 | | 4 | | 6 | | 6 | | 9 | ns |
| | | Mil | | | | 4 | | 6 | | 6 | | 9 | |
| $t_{EXP}$ | Expander Array Delay | Com'l | | 8 | | 10 | | 12 | | 14 | | 20 | ns |
| | | Mil | | | | 10 | | 12 | | 14 | | 20 | |
| $t_{LAD}$ | Logic Array Data Delay | Com'l | | 8 | | 10 | | 12 | | 14 | | 16 | ns |
| | | Mil | | | | 10 | | 12 | | 14 | | 16 | |
| $t_{LAC}$ | Logic Array Control Delay | Com'l | | 5 | | 7 | | 10 | | 12 | | 13 | ns |
| | | Mil | | | | 7 | | 10 | | 12 | | 13 | |
| $t_{OD}$ | Output Buffer and Pad Delay | Com'l | | 3 | | 3 | | 5 | | 5 | | 6 | ns |
| | | Mil | | | | 3 | | 5 | | 5 | | 6 | |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | Com'l | | 5 | | 5 | | 10 | | 11 | | 13 | ns |
| | | Mil | | | | 5 | | 10 | | 11 | | 13 | |
| $t_{XZ}$ | Output Buffer Disable Delay | Com'l | | 5 | | 5 | | 10 | | 11 | | 13 | ns |
| | | Mil | | | | 5 | | 10 | | 11 | | 13 | |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | Com'l | 4 | | 5 | | 6 | | 8 | | 10 | | ns |
| | | Mil | | | 5 | | 6 | | 8 | | 10 | | |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | Com'l | 4 | | 5 | | 6 | | 8 | | 10 | | ns |
| | | Mil | | | 5 | | 6 | | 8 | | 10 | | |
| $t_{LATCH}$ | Flow-Through Latch Delay | Com'l | | 1 | | 2 | | 3 | | 4 | | 4 | ns |
| | | Mil | | | | 2 | | 3 | | 4 | | 4 | |
| $t_{RD}$ | Register Delay | Com'l | | 1 | | 1 | | 1 | | 2 | | 2 | ns |
| | | Mil | | | | 1 | | 1 | | 2 | | 2 | |
| $t_{COMB}$ | Transparent Mode Delay[28] | Com'l | | 1 | | 2 | | 3 | | 4 | | 4 | ns |
| | | Mil | | | | 2 | | 3 | | 4 | | 4 | |
| $t_{CH}$ | Clock High Time | Com'l | 4 | | 6 | | 8 | | 10 | | 12.5 | | ns |
| | | Mil | | | 6 | | 8 | | 10 | | 12.5 | | |
| $t_{CL}$ | Clock Low Time | Com'l | 4 | | 6 | | 8 | | 10 | | 12.5 | | ns |
| | | Mil | | | 6 | | 8 | | 10 | | 12.5 | | |

Shaded area contains preliminary information.

## Internal Switching Characteristics Over the Operating Range[2]

| | | | 7C341B-15 | | 7C341B-20 | | 7C341B-25 | | 7C341B-30 | | 7C341B-35 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | Description | | Min. | Max | Min. | Max | Min. | Max | Min. | Max | Min. | Max | Unit |
| $t_{IC}$ | Asynchronous Clock Logic Delay | Com'l | | 6 | | 8 | | 14 | | 16 | | 18 | ns |
| | | Mil | | | | 8 | | 14 | | 16 | | 18 | |
| $t_{ICS}$ | Synchronous Clock Delay | Com'l | | 0.5 | | 0.5 | | 2 | | 2 | | 3 | ns |
| | | Mil | | | | 0.5 | | 2 | | 2 | | 3 | |
| $t_{FD}$ | Feedback Delay | Com'l | | 1 | | 1 | | 1 | | 1 | | 2 | ns |
| | | Mil | | | | 1 | | 1 | | 1 | | 2 | |
| $t_{PRE}$ | Asynchronous Register Preset Time | Com'l | | 3 | | 3 | | 5 | | 6 | | 7 | ns |
| | | Mil | | | | 3 | | 5 | | 6 | | 7 | |
| $t_{CLR}$ | Asynchronous Register Clear Time | Com'l | | 3 | | 3 | | 5 | | 6 | | 7 | ns |
| | | Mil | | | | 3 | | 5 | | 6 | | 7 | |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | Com'l | 3 | | 4 | | 5 | | 6 | | 7 | | ns |
| | | Mil | | | 4 | | 5 | | 6 | | 7 | | |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | Com'l | 3 | | 4 | | 5 | | 6 | | 7 | | ns |
| | | Mil | | | 4 | | 5 | | 6 | | 7 | | |
| $t_{PIA}$ | Programmable Interconnect Array Delay | Com'l | | 10 | | 12 | | 14 | | 16 | | 20 | ns |
| | | Mil | | | | | | | | 16 | | 20 | |

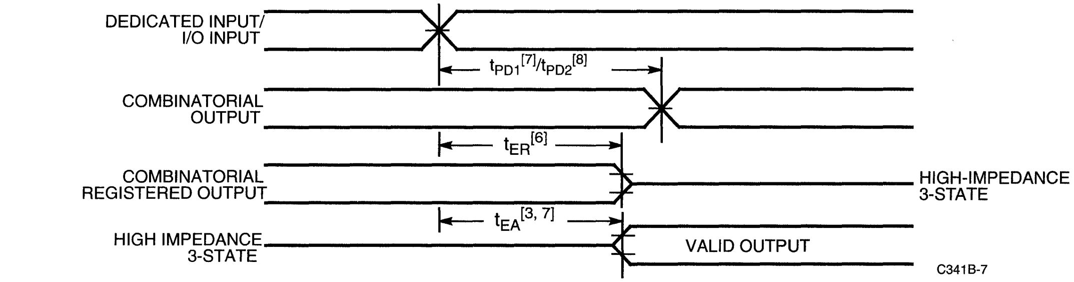
Shaded area contains preliminary information.

**Notes:**

26. This parameter indicates the minimum time that the previous register output data is maintained on the output after an asynchronous register clock input applied to an external dedicated input pin.
27. Sample tested only for an output change of 500 mV.
28. This specification guarantees the maximum combinatorial delay associated with the macrocell register bypass when the macrocell is configured for combinatorial operation.

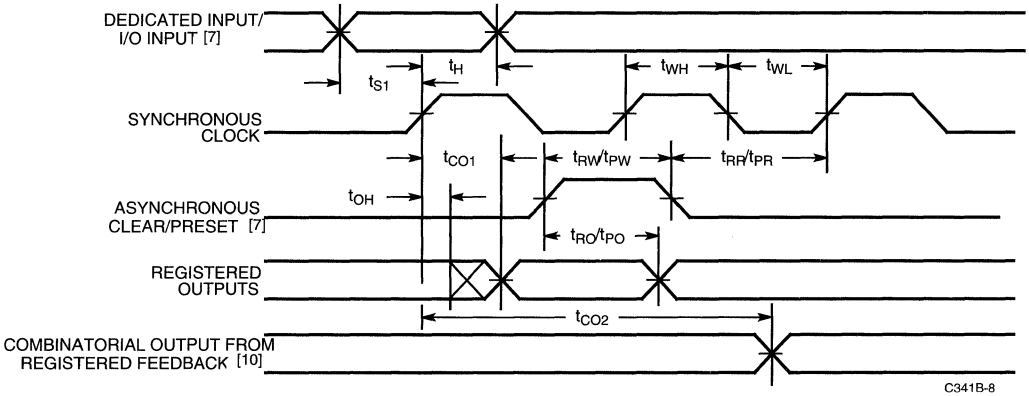
## Switching Waveforms

### External Combinatorial



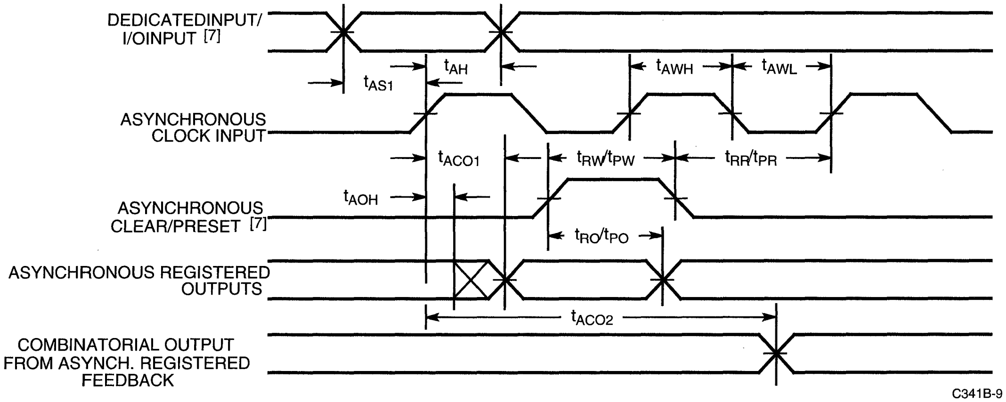
C341B-7

### External Synchronous



C341B-8

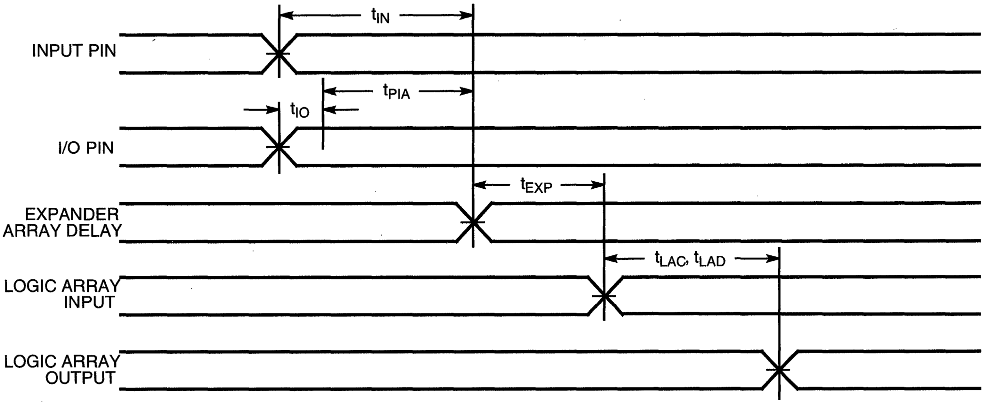
### External Asynchronous



C341B-9

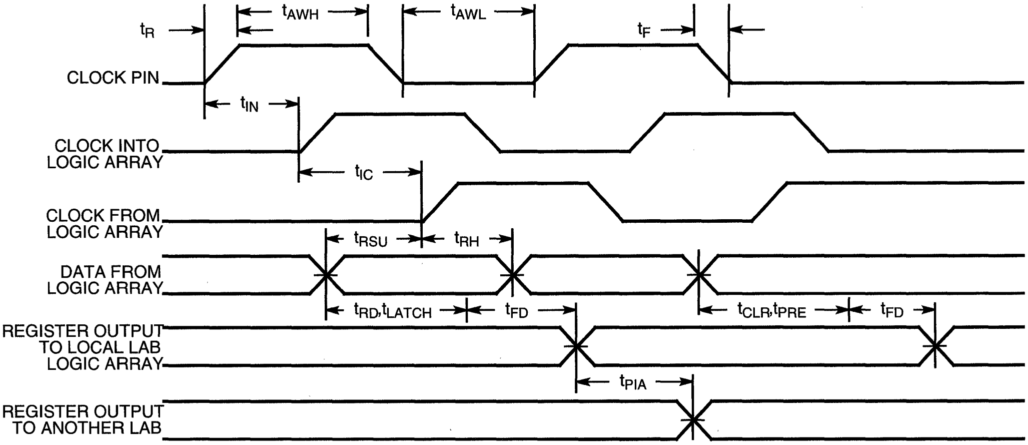
## Switching Waveforms (continued)

### Internal Combinatorial



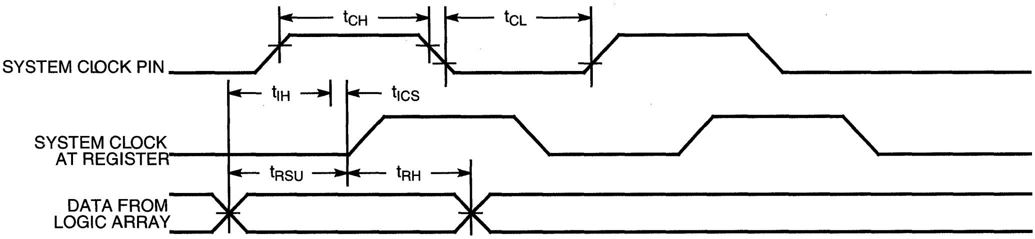
C341B-10

### Internal Asynchronous



C341B-11

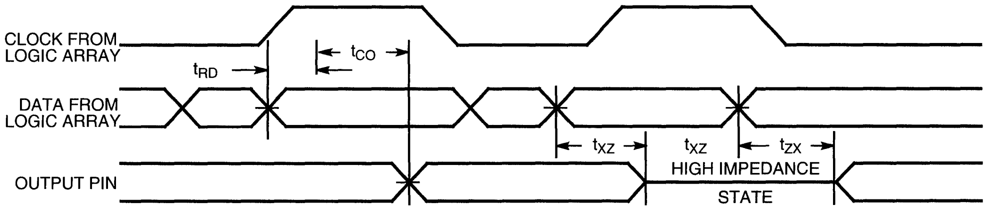
### Internal Synchronous



C341B-12

## Switching Waveforms (continued)

### Internal Synchronous



C341B-13

## Ordering Information

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 15 | CY7C341B-15HC/HI | H84 | 84-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C341B-15JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C341B-15RC/RI | R84 | 84-Lead Windowed Pin Grid Array | |
| 20 | CY7C341B-20HC/HI | H84 | 84-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C341B-20JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C341B-20RC/RI | R84 | 84-Lead Windowed Pin Grid Array | |
| | CY7C341B-20HMB | H84 | 84-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C341B-20RMB | R84 | 84-Lead Windowed Pin Grid Array | |
| 25 | CY7C341B-25HC/HI | H84 | 84-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C341B-25JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C341B-25RC/RI | R84 | 84-Lead Windowed Pin Grid Array | |
| | CY7C341B-25HMB | H84 | 84-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C341B-25RMB | R84 | 84-Lead Windowed Pin Grid Array | |
| 30 | CY7C341B-30HC/HI | H84 | 84-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C341B-30JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C341B-30RC/RI | R84 | 84-Lead Windowed Pin Grid Array | |
| | CY7C341B-30HMB | H84 | 84-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C341B-30RMB | R84 | 84-Lead Windowed Pin Grid Array | |
| 35 | CY7C341B-35HC/HI | H84 | 84-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C341B-35JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C341B-35RC/RI | R84 | 84-Lead Windowed Pin Grid Array | |
| | CY7C341B-35HMB | H84 | 84-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C341B-35RMB | R84 | 84-Lead Windowed Pin Grid Array | |

Shaded area contains preliminary information.

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD1}$ | 7, 8, 9, 10, 11 |
| $t_{PD2}$ | 7, 8, 9, 10, 11 |
| $t_{PD3}$ | 7, 8, 9, 10, 11 |
| $t_{CO1}$ | 7, 8, 9, 10, 11 |
| $t_{S1}$ | 7, 8, 9, 10, 11 |
| $t_{H}$ | 7, 8, 9, 10, 11 |
| $t_{ACO1}$ | 7, 8, 9, 10, 11 |
| $t_{ACO2}$ | 7, 8, 9, 10, 11 |
| $t_{AS1}$ | 7, 8, 9, 10, 11 |
| $t_{AH}$ | 7, 8, 9, 10, 11 |

Document #: 38-00137-G

**This is an abbreviated data sheet. Contact a Cypress Representative for complete specifications. For new designs please refer to the CY7C341B.**

fax id: 6105

CYPRESS

# CY7C341

# 192-Macrocell MAX® EPLD

## Features

- **92 macrocells in 12 LABs**
- **8 dedicated inputs, 64 bidirectional I/O pins**
- **0.8-micron double-metal CMOS EPROM technology**
- **Programmable interconnect array**
- **384 expander product terms**
- **Available in 84-pin HLCC, PLCC, and PGA packages**

## Functional Description

The CY7C341 is an Erasable Programmable Logic Device (EPLD) in which CMOS EPROM cells are used to configure logic functions within the device. The MAX architecture is 100% user configurable allowing the devices to accommodate a variety of independent logic functions.

The 192 macrocells in the CY7C341 are divided into 12 Logic Array Blocks (LABs), 16 per LAB. There are 384 expander product terms, 32 per LAB, to be used and shared by the macrocells within each LAB. Each LAB is interconnected with a programmable interconnect array, allowing all signals to be routed throughout the chip.

The speed and density of the CY7C341 allows them to be used in a wide range of applications, from replacement of large amounts of 7400 series TTL logic, to complex controllers and multifunction chips. With greater than 37 times the functionality of 20-pin PLDs, the CY7C341 allows the replacement of over 75 TTL devices. By replacing large amounts of logic, the CY7C341 reduces board space, part count, and increases system reliability.

Each LAB contains 16 macrocells. In LABs A, F, G, and L, 8 macrocells are connected to I/O pins and 8 are buried, while for LABs B, C, D, E, H, I, J, and K, 4 macrocells are connected to I/O pins and 12 are buried. Moreover, in addition to the I/O and buried macrocells, there are 32 single product term logic expanders in each LAB. Their use greatly enhances the capability of the macrocells without increasing the number of product terms in each macrocell.

## Logic Array Blocks

There are 12 logic array blocks in the CY7C341. Each LAB consists of a macrocell array containing 16 macrocells, an expander product term array containing 32 expanders, and an I/O block. The LAB is fed by the programmable interconnect array and the dedicated input bus. All macrocell feedbacks go to the macrocell array, the expander array, and the programmable interconnect array. Expanders feed themselves and the macrocell array. All I/O feedbacks go to the programmable interconnect array so that they may be accessed by macrocells in other LABs as well as the macrocells in the LAB in which they are situated.

Externally, the CY7C341 provides 8 dedicated inputs, one of which may be used as a system clock. There are 64 I/O pins that may be individually configured for input, output, or bidirectional data flow.

## Programmable Interconnect Array

The Programmable Interconnect Array (PIA) solves interconnect limitations by routing only the signals needed by each logic array block. The inputs to the PIA are the outputs of every macrocell within the device and the I/O pin feedback of every pin on the device.

Unlike masked or programmable gate arrays, which induce variable delay dependent on routing, the PIA has a fixed delay. This eliminates undesired skews among logic signals, which may cause glitches in internal or external logic. The fixed delay, regardless of programmable interconnect array configuration, simplifies design by assuring that internal signal skews or races are avoided. The result is ease of design implementation, often in a single pass, without the multiple internal logic placement and routing iterations required for a programmable gate array to achieve design timing objectives.

## Timing Delays

Timing delays within the CY7C341 may be easily determined using *Warp2*® or *Warp3*® software. The CY7C341 has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information, the *Warp3* software provides a timing simulator.

## Design Recommendations

For proper operation, input and output pins must be constrained to the range GND ≤ (VIN or VOUT) ≤ VCC. Unused inputs must always be tied to an appropriate logic level (either VCC or GND). Each set of VCC and GND pins must be connected together directly at the device. Power supply decoupling capacitors of at least 0.2 µF must be connected between VCC and GND. For the most effective decoupling, each VCC pin should be separately decoupled to GND, directly at the device. Decoupling capacitors should have good frequency response, such as monolithic ceramic types.

## Design Security

The CY7C341 contains a programmable design security feature that controls the access to the data programmed into the device. If this programmable feature is used, a proprietary design implemented in the device cannot be copied or retrieved. This enables a high level of design control to be obtained since programmed data within EPROM cells is invisible. The bit that controls this function, along with all other program data, may be reset simply by erasing the device.

## Selection Guide

| | | 7C341-25 | 7C341-30 | 7C341-35 |
|---|---|---|---|---|
| Maximum Access Time (ns) | | 25 | 30 | 35 |
| Maximum Operating Current (mA) | Commercial | 380 | 380 | 380 |
| | Industrial | 480 | 480 | 480 |
| | Military | 480 | 480 | 480 |
| Maximum Standby Current (mA) | Commercial | 360 | 360 | 360 |
| | Industrial | 435 | 435 | 435 |
| | Military | 435 | 435 | 435 |

MAX is a registered trademark of Altera Corporation.
*Warp2* and *Warp3* are trademarks of Cypress Semiconductor Corporation.

Document #: 38-00499

CYPRESS

# CY7C342B

# 128-Macrocell MAX® EPLDs

## Features

- **128 macrocells in 8 LABs**
- **8 dedicated inputs, 52 bidirectional I/O pins**
- **Programmable interconnect array**
- **Advanced 0.65-micron CMOS technology to increase performance**
- **Available in 68-pin HLCC, PLCC, and PGA**

## Functional Description

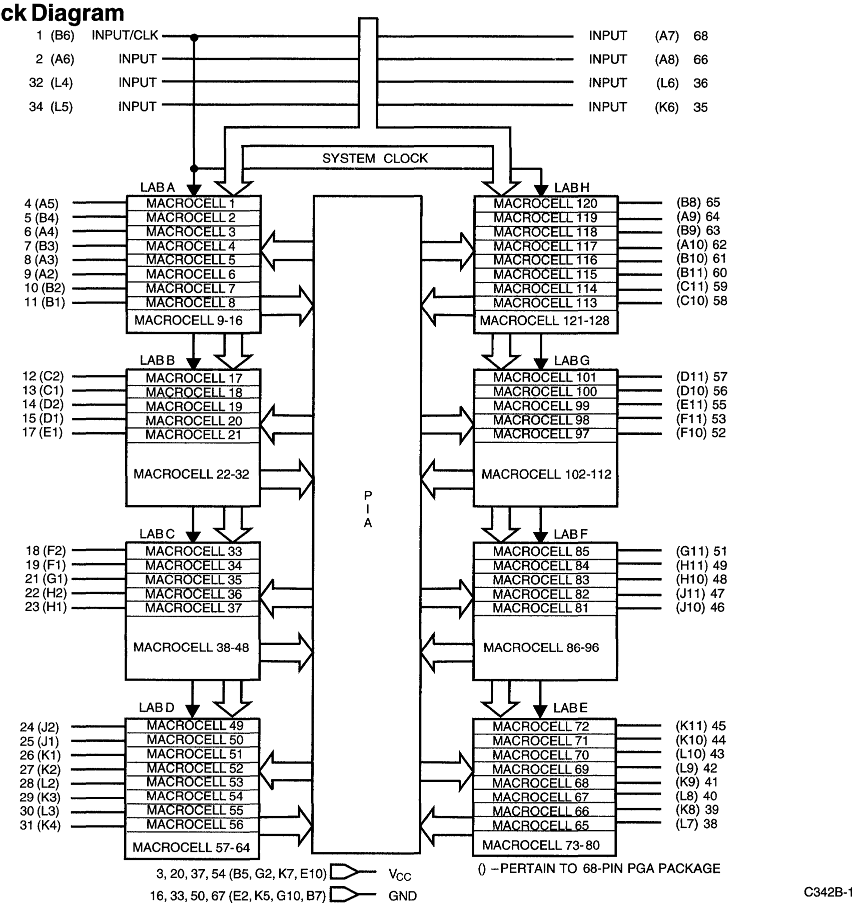
The CY7C342B is an Erasable Programmable Logic Device (EPLD) in which CMOS EPROM cells are used to configure logic functions within the device. The MAX architecture is 100% user configurable, allowing the devices to accommodate a variety of independent logic functions.

The 128 macrocells in the CY7C342B are divided into 8 Logic Array Blocks (LABs), 16 per LAB. There are 256 expander product terms, 32 per LAB, to be used and shared by the macrocells within each LAB.

Each LAB is interconnected with a programmable interconnect array, allowing all signals to be routed throughout the chip.

The speed and density of the CY7C342B allows it to be used in a wide range of applications, from replacement of large amounts of 7400-series TTL logic, to complex controllers and multifunction chips. With greater than 25 times the functionality of 20-pin PLDs, the CY7C342B allows the replacement of over 50 TTL devices. By replacing large amounts of logic, the CY7C342B reduces board space, part count, and increases system reliability.

### Logic Block Diagram



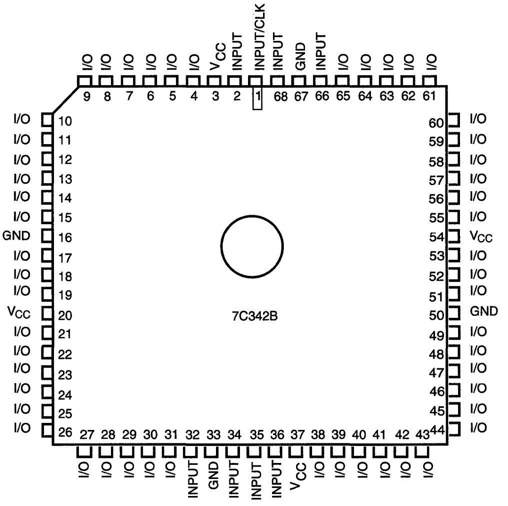
MAX is a registered trademark of Altera Corporation. *Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor.

## Selection Guide

| | | 7C342B–12 | 7C342B–15 | 7C342B–20 | 7C342B–25 | 7C342B–30 | 7C342B–35 |
|---|---|---|---|---|---|---|---|
| Maximum Access Time (ns) | | 12 | 15 | 20 | 25 | 30 | 35 |
| Maximum Operating Current (mA) | Commercial | 250 | 250 | 250 | 250 | 250 | 250 |
| | Military | | 320 | 320 | 320 | 320 | 320 |
| | Industrial | | 320 | 320 | 320 | 320 | 320 |
| Maximum Static Current (mA) | Commercial | 225 | 225 | 225 | 225 | 225 | 225 |
| | Military | | 275 | 275 | 275 | 275 | 275 |
| | Industrial | | 275 | 275 | 275 | 275 | 275 |

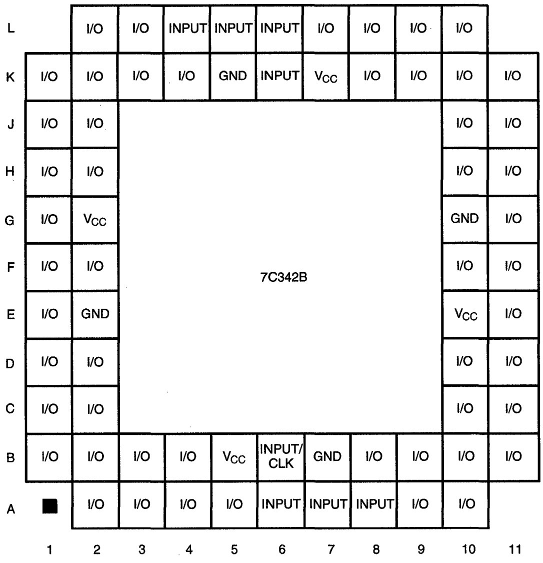
## Pin Configurations

PLCC
Top View



C342B-2

PGA
BottomView



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| L | | I/O | I/O | INPUT | INPUT | INPUT | I/O | I/O | I/O | I/O | |
| K | I/O | I/O | I/O | I/O | GND | INPUT | $V_{CC}$ | I/O | I/O | I/O | I/O |
| J | I/O | I/O | | | | | | | | I/O | I/O |
| H | I/O | I/O | | | | | | | | I/O | I/O |
| G | I/O | $V_{CC}$ | | | | | | | | GND | I/O |
| F | I/O | I/O | | | | 7C342B | | | | I/O | I/O |
| E | I/O | GND | | | | | | | | $V_{CC}$ | I/O |
| D | I/O | I/O | | | | | | | | I/O | I/O |
| C | I/O | I/O | | | | | | | | I/O | I/O |
| B | I/O | I/O | I/O | I/O | $V_{CC}$ | INPUT/ CLK | GND | I/O | I/O | I/O | I/O |
| A | ■ | I/O | I/O | I/O | I/O | INPUT | INPUT | INPUT | I/O | I/O | |

C342B-3

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Storage Temperature | −65°C to+150°C |
| Ambient Temperature with Power Applied | 0°C to+70°C |
| Maximum Junction Temperature (under bias) | 150°C |
| Supply Voltage to Ground Potential | −3.0V to+7.0V |
| Maximum Power Dissipation | 2500 mW |
| DC $V_{CC}$ or GND Current | 500 mA |
| DC Output Current per Pin | −25 mA to+25 mA |
| DC Input Voltage[1] | −3.0V to + 7.0V |
| DC Program Voltage | 13.0V |
| Static Discharge Voltage (per MIL-STD-883, Method 3015) | >1100V |

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | −40°C to +85°C | 5V ± 10% |
| Military | −55°C to +125°C (Case) | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $I_{OH}$ = −4.0 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $I_{OL}$ = 8.0 mA | | | 0.45 | V |
| $V_{IH}$ | Input HIGH Voltage | | | 2.2 | $V_{CC}$ +0.3 | V |
| $V_{IL}$ | Input LOW Voltage | | | −0.3 | 0.8 | V |
| $I_{IX}$ | Input Current | GND ≤ $V_{IN}$ ≤ $V_{CC}$ | | −10 | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = $V_{CC}$ or GND | | −40 | +40 | µA |
| $I_{OS}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[3, 4] | | −30 | −90 | mA |
| $I_{CC1}$ | Power Supply Current (Static) | $V_I$ = GND (No Load) | Com'l | | 225 | mA |
| | | | Mil/Ind | | 275 | |
| $I_{CC2}$ | Power Supply Current[5] | $V_I$ = $V_{CC}$ or GND (No Load)<br>f = 1.0 MHz[4] | Com'l | | 250 | mA |
| | | | Mil/Ind | | 320 | |
| $t_R$ | Recommended Input Rise Time | | | | 100 | ns |
| $t_F$ | Recommended Input Fall Time | | | | 100 | ns |

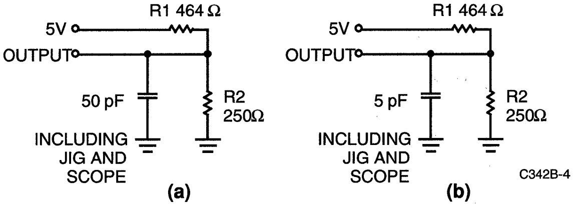
## Capacitance[6]

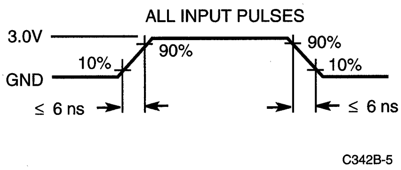
| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2V, f = 1.0 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2V, f = 1.0 MHz | 10 | pF |

**Notes:**

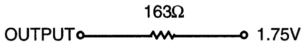
1. Minimum DC input is −0.3V. During transitions, the inputs may undershoot to −3.0V for periods less than 20 ns.
2. Typical values are for $T_A$ = 25°C and $V_{CC}$ = 5V.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
4. Guaranteed but not 100% tested.
5. This parameter is measured with device programmed as a 16-bit counter in each LAB.
6. Part (a) in AC Test Load and Waveforms is used for all parameters except $t_{ER}$ and $t_{XZ}$, which is used for part (b) in AC Test Load and Waveforms. All external timing parameters are measured referenced to external pins of the device.

## AC Test Loads and Waveforms[5]





Equivalent to: THÉVENIN EQUIVALENT (commercial/military)



## Logic Array Blocks

There are 8 logic array blocks in the CY7C342B. Each LAB consists of a macrocell array containing 16 macrocells, an expander product term array containing 32 expanders, and an I/O block. The LAB is fed by the programmable interconnect array and the dedicated input bus. All macrocell feedbacks go to the macrocell array, the expander array, and the programmable interconnect array. Expanders feed themselves and the macrocell array. All I/O feedbacks go to the programmable interconnect array so that they may be accessed by macrocells in other LABs as well as the macrocells in the LAB in which they are situated.

Externally, the CY7C342B provides eight dedicated inputs, one of which may be used as a system clock. There are 52 I/O pins that may be individually configured for input, output, or bidirectional data flow.

## Programmable Interconnect Array

The Programmable Interconnect Array (PIA) solves interconnect limitations by routing only the signals needed by each logic array block. The inputs to the PIA are the outputs of every macrocell within the device and the I/O pin feedback of every pin on the device.

Unlike masked or programmable gate arrays, which induce variable delay dependent on routing, the PIA has a fixed delay. This eliminates undesired skews among logic signals that may cause glitches in internal or external logic. The fixed delay, regardless of programmable interconnect array configuration, simplifies design by assuring that internal signal skews or races are avoided. The result is ease of design implementation, often in a signal pass, without the multiple internal logic placement and routing iterations required for a programmable gate array to achieve design timing objectives.

## Timing Delays

Timing delays within the CY7C342B may be easily determined using *Warp2*® or *Warp3*®software by the model shown in *Figure 1*. The CY7C342B has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information the *Warp3* software provides a timing simulator.

## Design Recommendations

Operation of the devices described herein with conditions above those listed under “Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this datasheet is not implied. Exposure to absolute maximum ratings conditions for extended periods of time may affect device reliability. The CY7C342B contains circuitry to protect device pins from high static voltages or electric fields, but normal precautions should be taken to avoid application of any voltage higher than the maximum rated voltages.

For proper operation, input and output pins must be constrained to the range GND $\leq$ ($V_{IN}$ or $V_{OUT}$) $\leq$ $V_{CC}$. Unused inputs must always be tied to an appropriate logic level (either $V_{CC}$ or GND). Each set of $V_{CC}$ and GND pins must be connected together directly at the device. Power supply decoupling capacitors of at least 0.2 µF must be connected between VCC and GND. For the most effective decoupling, each $V_{CC}$ pin should be separately decoupled to GND directly at the device. Decoupling capacitors should have good frequency response, such as monolithic ceramic types have.

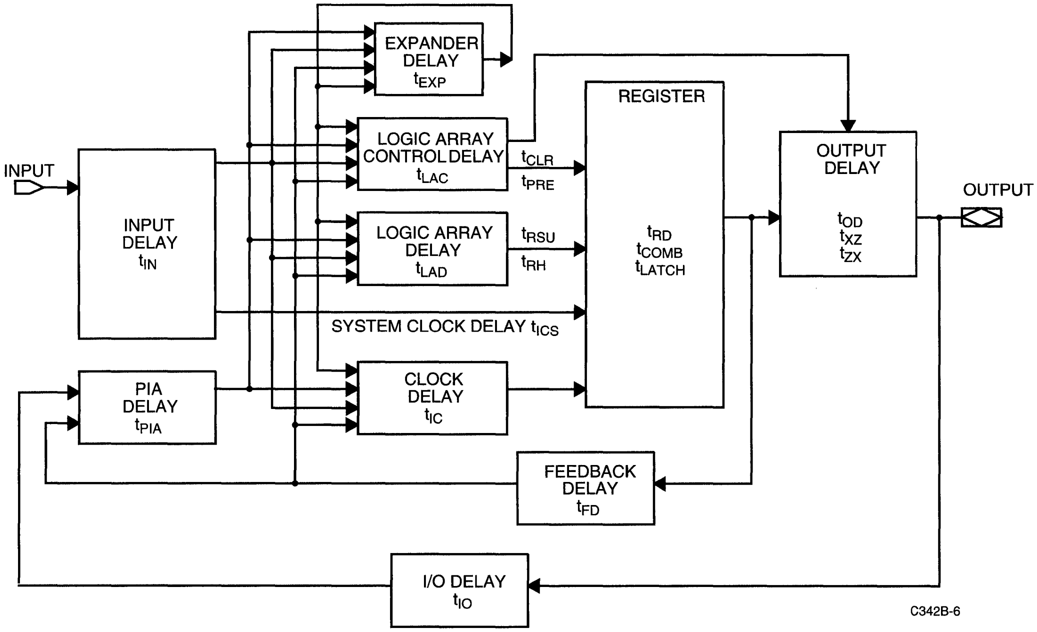


**Figure 1. CY7C342B Internal Timing Model**

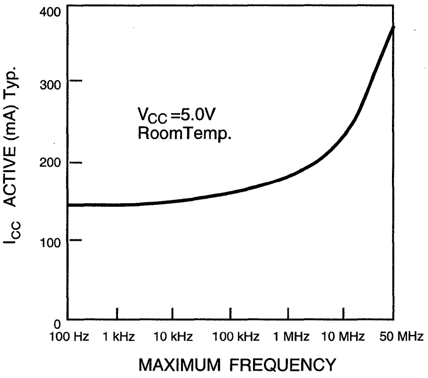
## Design Security

The CY7C342B contains a programmable design security feature that controls the access to the data programmed into the device. If this programmable feature is used, a proprietary design implemented in the device cannot be copied or retrieved. This enables a high level of design control to be obtained since programmed data within EPROM cells is invisible. The bit that controls this function, along with all other program data, may be reset simply by erasing the entire device.

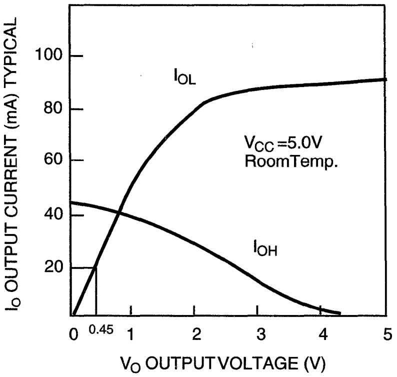
The CY7C342B is fully functionally tested and guaranteed through complete testing of each programmable EPROM bit and all internal logic elements thus ensuring 100% programming yield.

The erasable nature of these devices allows test programs to be used and erased during early stages of the production flow. The devices also contain on-board logic test circuitry to allow verification of function and AC specification once encapsulated in non-windowed packages.

## Typical $I_{CC}$ vs. $f_{MAX}$



## Output Drive Current



## Timing Considerations

Unless otherwise stated, propagation delays do not include expanders. When using expanders, add the maximum expander delay $t_{EXP}$ to the overall delay. Similarly, there is an additional $t_{PIA}$ delay for an input from an I/O pin when compared to a signal from straight input pin.

When calculating synchronous frequencies, use $t_{S1}$ if all inputs are on dedicated input pins. The parameter $t_{S2}$ should be used if data is applied at an I/O pin. If $t_{S2}$ is greater than $t_{CO1}$, $1/t_{S2}$ becomes the limiting frequency in the data path mode unless $1/(t_{WH} + t_{WL})$ is less than $1/t_{S2}$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{S1}$. Determine which of $1/(t_{WH} + t_{WL})$, $1/t_{CO1}$, or $1/(t_{EXP} + t_{S1})$ is the lowest frequency. The lowest of these frequencies is the maximum data path frequency for the synchronous configuration.

When calculating external asynchronous frequencies, use $t_{AS1}$ if all inputs are on the dedicated input pins. If any data is applied to an I/O pin, $t_{AS2}$ must be used as the required set-up time. If $(t_{AS2} + t_{AH})$ is greater than $t_{ACO1}$, $1/(t_{AS2} + t_{AH})$ becomes the limiting frequency in the data path mode unless $1/(t_{AWH} + t_{AWL})$ is less than $1/(t_{AS2} + t_{AH})$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{AS1}$. Determine which of $1/(t_{AWH} + t_{AWL})$, $1/t_{ACO1}$, or $1/(t_{EXP} + t_{AS1})$ is the lowest frequency. The lowest of these frequencies is the maximum data path frequency for the asynchronous configuration.

The parameter $t_{OH}$ indicates the system compatibility of this device when driving other synchronous logic with positive input hold times, which is controlled by the same synchronous clock. If $t_{OH}$ is greater than the minimum required input hold time of the subsequent synchronous logic, then the devices are guaranteed to function properly with a common synchronous clock under worst-case environmental and supply voltage conditions.

The parameter $t_{AOH}$ indicates the system compatibility of this device when driving subsequent registered logic with a positive hold time and using the same asynchronous clock as the CY7C342B.

In general, if $t_{AOH}$ is greater than the minimum required input hold time of the subsequent logic (synchronous or asynchronous) then the devices are guaranteed to function properly under worst-case environmental and supply voltage conditions, provided the clock signal source is the same. This also applies if expander logic is used in the clock signal path of the driving device, but not for the driven device. This is due to the expander logic in the second device's clock signal path adding an additional delay ($t_{EXP}$) causing the output data from the preceding device to change prior to the arrival of the clock signal at the following device's register.

## Commercial and Industrial External Synchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C342B–12 | | 7C342B–15 | | 7C342B–20 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | | 12 | | 15 | | 20 | ns |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | | 20 | | 25 | | 32 | ns |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | | 18 | | 23 | | 30 | ns |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4, 10] | | 26 | | 33 | | 42 | ns |
| $t_{EA}$ | Input to Output Enable Delay[4, 7] | | 12 | | 15 | | 20 | ns |
| $t_{ER}$ | Input to Output Disable Delay[4, 7] | | 12 | | 15 | | 20 | ns |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | | 6 | | 7 | | 8 | ns |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 11] | | 14 | | 17 | | 20 | ns |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[4, 12] | 8 | | 10 | | 13 | | ns |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7] | 16 | | 20 | | 24 | | ns |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | 0 | | 0 | | 0 | | ns |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | 4.5 | | 5 | | 7 | | ns |
| $t_{WL}$ | Synchronous Clock Input LOW Time | 4.5 | | 5 | | 7 | | ns |
| $t_{RW}$ | Asynchronous Clear Width[4, 7] | 12 | | 15 | | 20 | | ns |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4, 7] | 12 | | 15 | | 20 | | ns |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[7] | | 12 | | 15 | | 20 | ns |
| $t_{PW}$ | Asynchronous Preset Width[4, 7] | 12 | | 15 | | 20 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4, 7] | 12 | | 15 | | 20 | | ns |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[7] | | 12 | | 15 | | 20 | ns |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4, 13] | | 3 | | 3 | | 3 | ns |
| $t_P$ | External Synchronous Clock Period $(1/(f_{MAX3}))$[4] | 9 | | 12 | | 15 | | ns |
| $f_{MAX1}$ | External Feedback Maximum Frequency $(1/(t_{CO1} + t_{S1}))$[4, 14] | 71.4 | | 58.8 | | 47.6 | | MHz |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of $(1/(t_{S1} + t_{CF}))$ or $(1/t_{CO1})$[4, 15] | 90.9 | | 76.9 | | 62.5 | | MHz |

**Notes:**

[7] This specification is a measure of the delay from input signal applied to a dedicated input (68-pin PLCC input pin 1, 2, 32, 34, 35, 66, or 68) to combinatorial output on any output pin. This delay assumes no expander terms are used to form the logic function.

When this note is applied to any parameter specification it indicates that the signal (data, asynchronous clock, asynchronous clear, and/or asynchronous preset) is applied to a dedicated input only and no signal path (either clock or data) employs expander logic.

If an input signal is applied to an I/O pin an additional delay equal to $t_{PIA}$ should be added to the comparable delay for a dedicated input. If expanders are used, add the maximum expander delay $t_{EXP}$ to the overall delay for the comparable delay without expanders.

[8] This specification is a measure of the delay from input signal applied to an I/O macrocell pin to any output. This delay assumes no expander terms are used to form the logic function.

[9] This specification is a measure of the delay from an input signal applied to a dedicated input (68-pin PLCC input pin 1, 2, 32, 34, 35, 36, 66, or 68) to combinatorial output on any output pin. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic.

[10] This specification is a measure of the delay from an input signal applied to an I/O macrocell pin to any output. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.

[11] This specification is a measure of the delay from synchronous register clock to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used, register is synchronously clocked and all feedback is within the same LAB. This parameter is tested periodically by sampling production material.

[12] If data is applied to an I/O input for capture by a macrocell register, the I/O pin input set-up time minimums should be observed. These parameters are $t_{S2}$ for synchronous operation and $t_{AS2}$ for asynchronous operation.

[13] This specification is a measure of the delay associated with the internal register feedback path. This is the delay from synchronous clock to LAB logic array input. This delay plus the register set-up time, $t_{S1}$, is the minimum internal period for an internal synchronous state machine configuration. This delay is for feedback within the same LAB. This parameter is tested periodically by sampling production material.

[14] This specification indicates the guaranteed maximum frequency, in synchronous mode, at which a state machine configuration with external feedback can operate. It is assumed that all data inputs and feedback signals are applied to dedicated inputs. All feedback is assumed to be local originating within the same LAB.

[15] This specification indicates the guaranteed maximum frequency at which a state machine with internal-only feedback can operate. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{CO1}$.

## Commercial and Industrial External Synchronous Switching Characteristics[6] Over Operating Range

| | | 7C342B–12 | | 7C342B–15 | | 7C342B–20 | | |
|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $f_{MAX3}$ | Data Path Maximum Frequency, lesser of $(1/(t_{WL} + t_{WH}))$, $(1/(t_{S1} + t_H))$ or $(1/t_{CO1})$[4,16] | 111.1 | | 100 | | 71.4 | | MHz |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $(1/(t_{WL}+t_{WH}))$[4,17] | 111.1 | | 100 | | 71.4 | | MHz |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4,18] | 3 | | 3 | | 3 | | ns |

| | | 7C342B–25 | | 7C342B–30 | | 7C342B–35 | | |
|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | | 25 | | 30 | | 35 | ns |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | | 39 | | 46 | | 55 | ns |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | | 37 | | 44 | | 55 | ns |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4,8] | | 51 | | 60 | | 75 | ns |
| $t_{EA}$ | Input to Output Enable Delay[4,8] | | 25 | | 30 | | 35 | ns |
| $t_{ER}$ | Input to Output Disable Delay[4,7] | | 25 | | 30 | | 35 | ns |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | | 14 | | 16 | | 20 | ns |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4,9] | | 30 | | 35 | | 42 | ns |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[7,10] | 15 | | 20 | | 25 | | ns |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7] | 29 | | 36 | | 45 | | ns |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | 0 | | 0 | | 0 | | ns |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | 8 | | 10 | | 12.5 | | ns |
| $t_{WL}$ | Synchronous Clock Input LOW Time | 8 | | 10 | | 12.5 | | ns |
| $t_{RW}$ | Asynchronous Clear Width[4,7] | 25 | | 30 | | 35 | | ns |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4, 7] | 25 | | 30 | | 35 | | ns |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[7] | | 25 | | 30 | | 35 | ns |
| $t_{PW}$ | Asynchronous Preset Width[4,7] | 25 | | 30 | | 35 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4,7] | 25 | | 30 | | 35 | | ns |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[7] | | 25 | | 30 | | 35 | ns |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4,11] | | 3 | | 3 | | 6 | ns |
| $t_P$ | External Synchronous Clock Period $(1/(f_{MAX3}))$[4] | 16 | | 20 | | 25 | | ns |
| $f_{MAX1}$ | External Feedback Maximum Frequency $(1/(t_{CO1} + t_{S1}))$[4,12] | 34.5 | | 27.7 | | 22.2 | | MHz |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of $(1/(t_{S1} + t_{CF}))$ or $(1/t_{CO1})$[4,13] | 55.5 | | 43.4 | | 32.2 | | MHz |
| $f_{MAX3}$ | Data Path Maximum Frequency, lesser of $(1/(t_{WL} + t_{WH}))$, $(1/(t_{S1} + t_H))$ or $(1/t_{CO1})$[4,14] | 62.5 | | 50 | | 40 | | MHz |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $(1/(t_{WL}+t_{WH}))$[4,15] | 62.5 | | 50 | | 40 | | MHz |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4,16] | 3 | | 3 | | 3 | | ns |

**Notes:**

16. This frequency indicates the maximum frequency at which the device may operate in data path mode (dedicated input pin to output pin). This assumes data input signals are applied to dedicated input pins and no expander logic is used. If any of the data inputs are I/O pins, $t_{S2}$ is the appropriate $t_S$ for calculation.
17. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which an individual output or buried register can be cycled by a clock signal applied to the dedicated clock input pin.
18. This parameter indicates the minimum time after a synchronous register clock input that the previous register output data is maintained on the output pin.

## Commercial and Industrial External Asynchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C342B–12 | | 7C342B–15 | | 7C342B–20 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] | | 12 | | 15 | | 20 | ns |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | | 20 | | 25 | | 32 | ns |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | 4 | | 5 | | 5 | | ns |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | 12 | | 14.5 | | 17 | | ns |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | 4 | | 5 | | 6 | | ns |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | 8 | | 9 | | 10 | | ns |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | 6 | | 7 | | 8 | | ns |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4,21] | | 9 | | 11 | | 13 | ns |
| $t_{AP}$ | External Asynchronous Clock Period $(1/(f_{MAXA4}))$[4] | 14 | | 16 | | 18 | | ns |
| $f_{MAXA1}$ | External Feedback Maximum Frequency in Asynchronous Mode $(1/(t_{ACO1} + t_{AS1}))$[4,22] | 62.5 | | 50 | | 40 | | MHz |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4,23] | 71.4 | | 62.5 | | 55.5 | | MHz |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4,24] | 83.3 | | 66.6 | | 50 | | MHz |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4,25] | 71.4 | | 62.5 | | 55.5 | | MHz |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4,26] | 12 | | 12 | | 12 | | ns |

**Notes:**

[19] This specification is a measure of the delay from an asynchronous register clock input to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used in the logic of combinatorial output or the asynchronous clock input. The clock signal is applied to the dedicated clock input pin and all feedback is within a single LAB. This parameter is tested periodically by sampling production material.

[20] This parameter is measured with a positive-edge triggered clock at the register. For negative edge triggering, the $t_{AWH}$ and $t_{AWL}$ parameters must be swapped. If a given input is used to clock multiple registers with both positive and negative polarity, $t_{AWH}$ should be used for both $t_{AWH}$ and $t_{AWL}$.

[21] This specification is a measure of the delay associated with the internal register feedback path for an asynchronous clock to LAB logic array input. This delay plus the asynchronous register set-up time, $t_{AS1}$, is the minimum internal period for an internal asynchronously clocked state machine configuration. This delay is for feedback within the same LAB, assumes no expander logic in the clock path, and assumes that the clock input signal is applied to a dedicated input pin. This parameter is tested periodically by sampling production material.

[22] This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine configuration with external feedback can operate. It is assumed that all data inputs, clock inputs, and feedback signals are applied to dedicated inputs and that no expander logic is employed in the clock signal path or data path.

[23] This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine with internal-only feedback can operate. This parameter is determined by the lesser of $(1/(t_{ACF} + t_{AS1}))$ or $(1/(t_{AWH} + t_{AWL}))$. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{ACO1}$.

This specification assumes no expander logic is utilized, all data inputs and clock inputs are applied to dedicated inputs, and all state feedback is within a single LAB. This parameter is tested periodically by sampling production material.

[24] This frequency is the maximum frequency at which the device may operate in the asynchronously clocked data path mode. This specification is determined by the lesser of $1/(t_{AWH} + t_{AWL})$, $1/(t_{AS1} + t_{AH})$ or $1/t_{ACO1}$. It assumes data and clock input signals are applied to dedicated input pins and no expander logic is used.

[25] This specification indicates the guaranteed maximum frequency at which an individual output or buried register can be cycled in asynchronously clocked mode by a clock signal applied to an external dedicated input pin.

[26] This parameter indicates the minimum time that the previous register output data is maintained on the output after an asynchronous register clock input applied to an external dedicated input pin.

## Commercial and Industrial External Asynchronous Switching Characteristics[6] Over Operating Range (continued)

| Parameter | Description | 7C342B–25 |  | 7C342B–30 |  | 7C342B–35 |  | Unit |
|---|---|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. | Min. | Max. |  |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] |  | 25 |  | 30 |  | 35 | ns |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] |  | 39 |  | 46 |  | 55 | ns |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | 5 |  | 6 |  | 8 |  | ns |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | 19 |  | 22 |  | 28 |  | ns |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | 6 |  | 8 |  | 10 |  | ns |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | 11 |  | 14 |  | 16 |  | ns |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | 9 |  | 11 |  | 14 |  | ns |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4,21] |  | 15 |  | 18 |  | 22 | ns |
| $t_{AP}$ | External Asynchronous Clock Period $(1/(f_{MAXA4}))$[4] | 20 |  | 25 |  | 30 |  | ns |
| $f_{MAXA1}$ | External Feedback Maximum Frequency in Asynchronous Mode $(1/(t_{ACO1} + t_{AS1}))$[4,22] | 33.3 |  | 27.7 |  | 23.2 |  | MHz |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4,23] | 50 |  | 40 |  | 33.3 |  | MHz |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4,24] | 40 |  | 33.3 |  | 28.5 |  | MHz |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4,25] | 50 |  | 40 |  | 33.3 |  | MHz |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4,26] | 15 |  | 15 |  | 15 |  | ns |

## Commercial and Industrial Typical Internal Switching Characteristics Over Operating Range

| Parameter | Description | 7C342B–12 |  | 7C342B–15 |  | 7C342B–20 |  | Unit |
|---|---|---|---|---|---|---|---|---|
|  |  | Min. | Max. | Min. | Max. | Min. | Max. |  |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay |  | 2.5 |  | 3 |  | 4 | ns |
| $t_{IO}$ | I/O Input Pad and Buffer Delay |  | 2.5 |  | 3 |  | 4 | ns |
| $t_{EXP}$ | Expander Array Delay |  | 6 |  | 8 |  | 10 | ns |
| $t_{LAD}$ | Logic Array Data Delay |  | 6 |  | 8 |  | 10 | ns |
| $t_{LAC}$ | Logic Array Control Delay |  | 5 |  | 5 |  | 7 | ns |
| $t_{OD}$ | Output Buffer and Pad Delay |  | 3 |  | 3 |  | 3 | ns |
| $t_{ZX}$ | Output Buffer Enable Delay[27] |  | 5 |  | 5 |  | 5 | ns |
| $t_{XZ}$ | Output Buffer Disable Delay |  | 5 |  | 5 |  | 5 | ns |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | 2 |  | 4 |  | 5 |  | ns |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | 4 |  | 4 |  | 5 |  | ns |
| $t_{LATCH}$ | Flow Through Latch Delay |  | 1 |  | 1 |  | 2 | ns |
| $t_{RD}$ | Register Delay |  | 0.5 |  | 1 |  | 1 | ns |
| $t_{COMB}$ | Transparent Mode Delay[28] |  | 1 |  | 1 |  | 2 | ns |
| $t_{CH}$ | Clock HIGH Time | 3 |  | 4 |  | 6 |  | ns |
| $t_{CL}$ | Clock LOW Time | 3 |  | 4 |  | 6 |  | ns |
| $t_{IC}$ | Asynchronous Clock Logic Delay |  | 5 |  | 6 |  | 8 | ns |
| $t_{ICS}$ | Synchronous Clock Delay |  | 0.5 |  | 0.5 |  | 0.5 | ns |
| $t_{FD}$ | Feedback Delay |  | 1 |  | 1 |  | 1 | ns |
| $t_{PRE}$ | Asynchronous Register Preset Time |  | 3 |  | 3 |  | 3 | ns |

**Notes:**

[27] Sample tested only for an output change of 500 mV.

[28] This specification guarantees the maximum combinatorial delay associated with the macrocell register bypass when the macrocell is configured for combinatorial operation.

## Commercial and Industrial Typical Internal Switching Characteristics Over Operating Range (continued)

| | | 7C342B–12 | | 7C342B–15 | | 7C342B–20 | | |
|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $t_{CLR}$ | Asynchronous Register Clear Time | | 3 | | 3 | | 3 | ns |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | 2 | | 3 | | 4 | | ns |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | 2 | | 3 | | 4 | | ns |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | | 8 | | 10 | | 12 | ns |

## Commercial and Industrial Typical Internal Switching Characteristics Over Operating Range (continued)

| | | 7C342B–25 | | 7C342B–30 | | 7C342B–35 | | |
|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | | 5 | | 7 | | 9 | ns |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | | 6 | | 6 | | 9 | ns |
| $t_{EXP}$ | Expander Array Delay | | 12 | | 14 | | 20 | ns |
| $t_{LAD}$ | Logic Array Data Delay | | 12 | | 14 | | 16 | ns |
| $t_{LAC}$ | Logic Array Control Delay | | 10 | | 12 | | 13 | ns |
| $t_{OD}$ | Output Buffer and Pad Delay | | 5 | | 5 | | 6 | ns |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | | 10 | | 11 | | 13 | ns |
| $t_{XZ}$ | Output Buffer Disable Delay | | 10 | | 11 | | 13 | ns |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | 6 | | 8 | | 10 | | ns |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | 6 | | 8 | | 10 | | ns |
| $t_{LATCH}$ | Flow Through Latch Delay | | 3 | | 4 | | 4 | ns |
| $t_{RD}$ | Register Delay | | 1 | | 2 | | 2 | ns |
| $t_{COMB}$ | Transparent Mode Delay[28] | | 3 | | 4 | | 4 | ns |
| $t_{CH}$ | Clock HIGH Time | 8 | | 10 | | 12.5 | | ns |
| $t_{CL}$ | Clock LOW Time | 8 | | 10 | | 12.5 | | ns |
| $t_{IC}$ | Asynchronous Clock Logic Delay | | 14 | | 16 | | 18 | ns |
| $t_{ICS}$ | Synchronous Clock Delay | | 2 | | 2 | | 3 | ns |
| $t_{FD}$ | Feedback Delay | | 1 | | 1 | | 2 | ns |
| $t_{PRE}$ | Asynchronous Register Preset Time | | 5 | | 6 | | 7 | ns |
| $t_{CLR}$ | Asynchronous Register Clear Time | | 5 | | 6 | | 7 | ns |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | 5 | | 6 | | 7 | | ns |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | 5 | | 6 | | 7 | | ns |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | | 14 | | 16 | | 20 | ns |

## Military External Synchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C342B–15 | | 7C342B–20 | | 7C342B–25 | | 7C342B–30 | | 7C342B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | | 25 | | 32 | | 39 | | 46 | | 55 | ns |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | | 23 | | 30 | | 37 | | 44 | | 55 | ns |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4,10] | | 33 | | 42 | | 51 | | 60 | | 75 | ns |
| $t_{EA}$ | Input to Output Enable Delay[4,7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{ER}$ | Input to Output Disable Delay[4,7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | | 7 | | 8 | | 14 | | 16 | | 20 | ns |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 11] | | 17 | | 20 | | 30 | | 35 | | 42 | ns |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[7,12] | 10 | | 13 | | 15 | | 20 | | 25 | | ns |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7] | 20 | | 24 | | 29 | | 36 | | 45 | | ns |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | 0 | | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | 5 | | 7 | | 8 | | 10 | | 12.5 | | ns |
| $t_{WL}$ | Synchronous Clock Input LOW Time | 5 | | 7 | | 8 | | 10 | | 12.5 | | ns |
| $t_{RW}$ | Asynchronous Clear Width[4, 7] | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4, 7] | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{PW}$ | Asynchronous Preset Width[4,7] | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4,7] | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4,13] | | 3 | | 3 | | 3 | | 3 | | 6 | ns |
| $t_P$ | External Synchronous Clock Period ($1/(f_{MAX3})$)[4] | 12 | | 14 | | 16 | | 20 | | 25 | | ns |
| $f_{MAX1}$ | External Feedback Maximum Frequency ($1/(t_{CO1} + t_{S1})$)[4,14] | 58.8 | | 47.6 | | 34.5 | | 27.7 | | 22.2 | | MHz |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of ($1/(t_{S1} + t_{CF})$) or ($1/t_{CO1}$)[4,15] | 76.9 | | 62.5 | | 55.5 | | 43.4 | | 32.2 | | MHz |

## Military External Synchronous Switching Characteristics[6] Over Operating Range (continued)

| Parameter | Description | 7C342B–15 | | 7C342B–20 | | 7C342B–25 | | 7C342B–30 | | 7C342B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $f_{MAX3}$ | Data Path Maximum Frequency, lesser of $(1/(t_{WL} + t_{WH}))$, $(1/(t_{S1} + t_H))$ or $(1/t_{CO1})$[4,16] | 100 | | 71.4 | | 62.5 | | 50 | | 40 | | MHz |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $(1/(t_{WL} + t_{WH}))$[4,17] | 100 | | 71.4 | | 62.5 | | 50 | | 40 | | MHz |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4,18] | 3 | | 3 | | 3 | | 3 | | 3 | | ns |

## Military External Asynchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C342B–15 | | 7C342B–20 | | 7C342B–25 | | 7C342B–30 | | 7C342B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | | 25 | | 32 | | 39 | | 46 | | 55 | ns |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | 5 | | 5 | | 5 | | 6 | | 8 | | ns |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | 14.5 | | 17 | | 19 | | 22 | | 28 | | ns |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | 5 | | 6 | | 6 | | 8 | | 10 | | ns |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | 9 | | 10 | | 11 | | 14 | | 16 | | ns |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | 7 | | 8 | | 9 | | 11 | | 14 | | ns |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4,21] | | 11 | | 13 | | 15 | | 18 | | 22 | ns |
| $t_{AP}$ | External Asynchronous Clock Period $(1/(f_{MAXA4}))$[4] | 16 | | 18 | | 20 | | 25 | | 30 | | ns |
| $f_{MAXA1}$ | External Feedback Maximum Frequency in Asynchronous Mode $(1/(t_{ACO1} + t_{AS1}))$[4,22] | 50.0 | | 40 | | 33.3 | | 27.7 | | 23.2 | | MHz |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4,23] | 62.5 | | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4,24] | 66.6 | | 50 | | 40 | | 33.3 | | 28.5 | | MHz |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4,25] | 62.5 | | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4,26] | 12 | | 12 | | 15 | | 15 | | 15 | | ns |

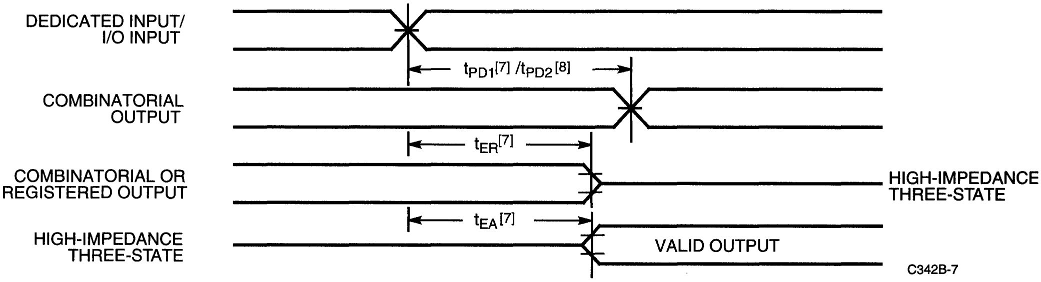
3

## Military Typical Internal Switching Characteristics Over Operating Range

| | | 7C342B–15 | | 7C342B–20 | | 7C342B–25 | | 7C342B–30 | | 7C342B–35 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | | 3 | | 4 | | 5 | | 7 | | 9 | ns |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | | 3 | | 4 | | 6 | | 6 | | 9 | ns |
| $t_{EXP}$ | Expander Array Delay | | 8 | | 10 | | 12 | | 14 | | 20 | ns |
| $t_{LAD}$ | Logic Array Data Delay | | 8 | | 10 | | 12 | | 14 | | 16 | ns |
| $t_{LAC}$ | Logic Array Control Delay | | 5 | | 7 | | 10 | | 12 | | 13 | ns |
| $t_{OD}$ | Output Buffer and Pad Delay | | 3 | | 3 | | 5 | | 5 | | 6 | ns |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | | 5 | | 5 | | 10 | | 11 | | 13 | ns |
| $t_{XZ}$ | Output Buffer Disable Delay | | 5 | | 5 | | 10 | | 11 | | 13 | ns |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | 4 | | 5 | | 6 | | 8 | | 10 | | ns |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | 4 | | 5 | | 6 | | 8 | | 10 | | ns |
| $t_{LATCH}$ | Flow Through Latch Delay | | 1 | | 2 | | 3 | | 4 | | 4 | ns |
| $t_{RD}$ | Register Delay | | 1 | | 1 | | 1 | | 2 | | 2 | ns |
| $t_{COMB}$ | Transparent Mode Delay[28] | | 1 | | 2 | | 3 | | 4 | | 4 | ns |
| $t_{CH}$ | Clock HIGH Time | 4 | | 6 | | 8 | | 10 | | 12.5 | | ns |
| $t_{CL}$ | Clock LOW Time | 4 | | 6 | | 8 | | 10 | | 12.5 | | ns |
| $t_{IC}$ | Asynchronous Clock Logic Delay | | 6 | | 8 | | 14 | | 16 | | 18 | ns |
| $t_{ICS}$ | Synchronous Clock Delay | | 0.5 | | 0.5 | | 2 | | 2 | | 3 | ns |
| $t_{FD}$ | Feedback Delay | | 1 | | 1 | | 1 | | 1 | | 2 | ns |
| $t_{PRE}$ | Asynchronous Register Preset Time | | 3 | | 3 | | 5 | | 6 | | 7 | ns |
| $t_{CLR}$ | Asynchronous Register Clear Time | | 3 | | 3 | | 5 | | 6 | | 7 | ns |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | 3 | | 4 | | 5 | | 6 | | 7 | | ns |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | 3 | | 4 | | 5 | | 6 | | 7 | | ns |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | | 10 | | 12 | | 14 | | 16 | | 20 | ns |

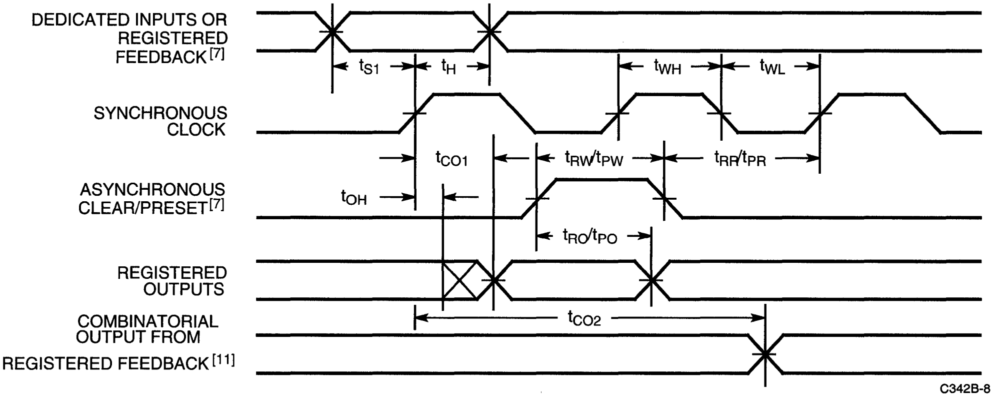
## Switching Waveforms

### External Combinatorial



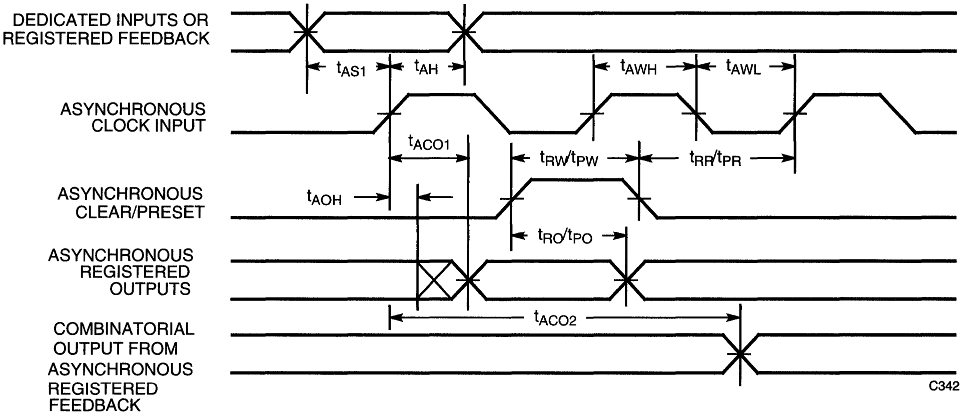
C342B-7

### External Synchronous



C342B-8

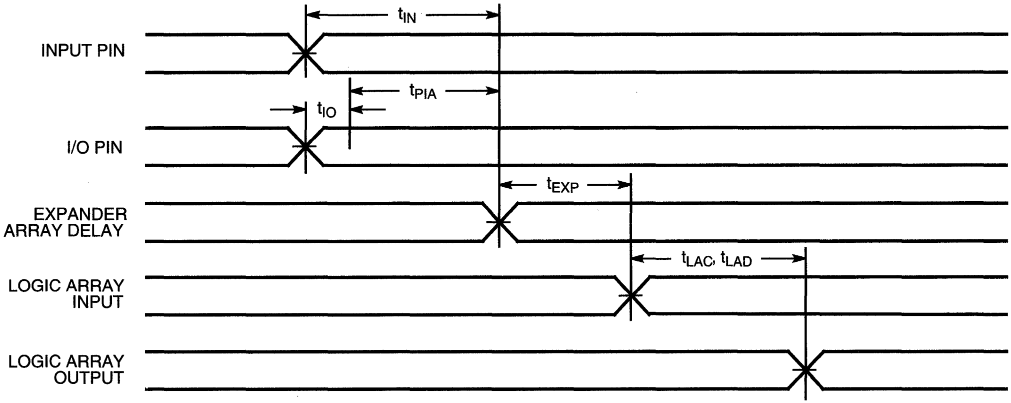
### External Asynchronous



C342B-9

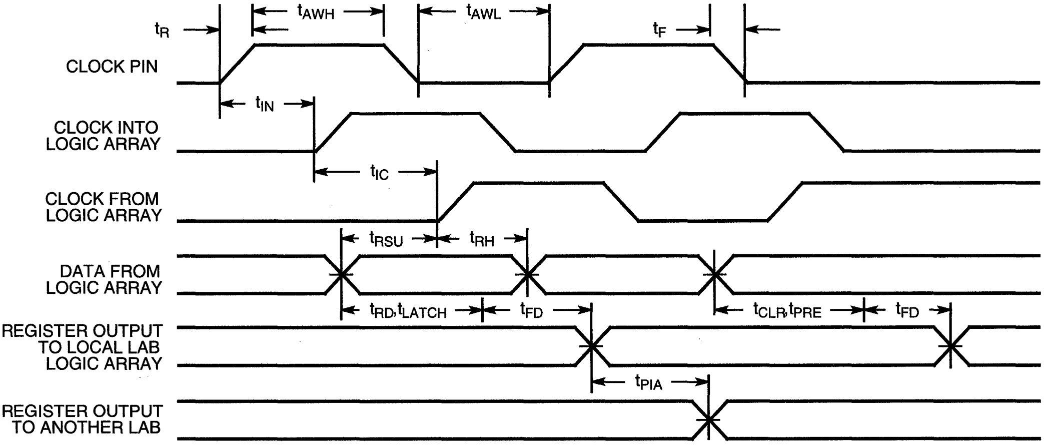
## Switching Waveforms (continued)

### Internal Combinatorial



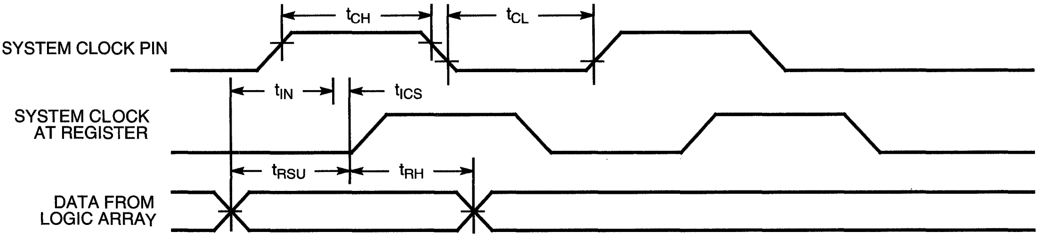
C342B-10

### Internal Asynchronous



C342B-11

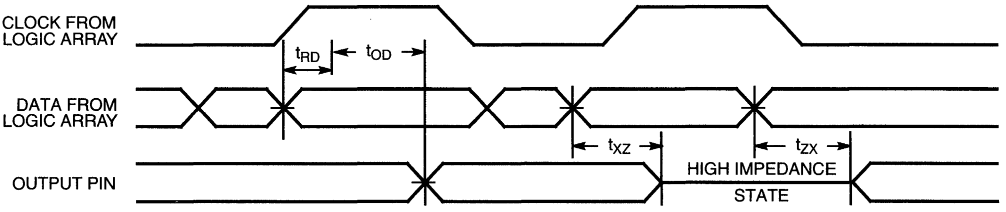
### InternalSynchronous



C342B-12

## Switching Waveforms (continued)

### Internal Synchronous



C342B-13

## Ordering Information

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 12 | CY7C342B–12HC | H81 | 68-Pin Windowed Leaded Chip Carrier | Commercial |
| | CY7C342B–12JC | J81 | 68-Lead Plastic Leaded Chip Carrier | |
| | CY7C342B–12RC | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| 15 | CY7C342B–15HC/HI | H81 | 68-Pin Windowed Leaded Chip Carrier | Commercial/ Industrial |
| | CY7C342B–15JC/JI | J81 | 68-Lead Plastic Leaded Chip Carrier | |
| | CY7C342B–15RC/RI | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C342B–15HMB | H81 | 68-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C342B–15RMB | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| 20 | CY7C342B–20HC/HI | H81 | 68-Pin Windowed Leaded Chip Carrier | Commercial/ Industrial |
| | CY7C342B–20JC/JI | J81 | 68-Lead Plastic Leaded Chip Carrier | |
| | CY7C342B–20RC/RI | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C342B–20HMB | H81 | 68-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C342B–20RMB | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| 25 | CY7C342B–25HC/HI | H81 | 68-Pin Windowed Leaded Chip Carrier | Commercial/ Industrial |
| | CY7C342B–25JC/JI | J81 | 68-Lead Plastic Leaded Chip Carrier | |
| | CY7C342B–25RC | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C342B–25HMB | H81 | 68-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C342B–25RMB | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| 30 | CY7C342B–30HC/HI | H81 | 68-Pin Windowed Leaded Chip Carrier | Commercial/ Industrial |
| | CY7C342B–30JC/JI | J81 | 68-Lead Plastic Leaded Chip Carrier | |
| | CY7C342B–30RC/RI | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C342B–30HMB | H81 | 68-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C342B–30RMB | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| 35 | CY7C342B–35HC/HI | H81 | 68-Pin Windowed Leaded Chip Carrier | Commercial/ Industrial |
| | CY7C342B–35JC/JI | J81 | 68-Lead Plastic Leaded Chip Carrier | |
| | CY7C342B–35RC/RI | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C342B–35HMB | H81 | 68-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C342B–35RMB | R68 | 68-Pin Windowed Ceramic Pin Grid Array | |

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD1}$ | 7, 8, 9, 10, 11 |
| $t_{PD2}$ | 7, 8, 9, 10, 11 |
| $t_{PD3}$ | 7, 8, 9, 10, 11 |
| $t_{CO1}$ | 7, 8, 9, 10, 11 |
| $t_{S1}$ | 7, 8, 9, 10, 11 |
| $t_{S2}$ | 7, 8, 9, 10, 11 |
| $t_H$ | 7, 8, 9, 10, 11 |
| $t_{WH}$ | 7, 8, 9, 10, 11 |
| $t_{WL}$ | 7, 8, 9, 10, 11 |
| $t_{RO}$ | 7, 8, 9, 10, 11 |
| $t_{PO}$ | 7, 8, 9, 10, 11 |
| $t_{ACO1}$ | 7, 8, 9, 10, 11 |
| $t_{AS1}$ | 7, 8, 9, 10, 11 |
| $t_{AH}$ | 7, 8, 9, 10, 11 |
| $t_{AWH}$ | 7, 8, 9, 10, 11 |
| $t_{AWL}$ | 7, 8, 9, 10, 11 |

Document #: 38–00119–G

**This is an abbreviated datasheet. Contact a Cypress Representative for complete specifications. For new designs, please refer to the CY7C342B.**

fax id: 6103

CYPRESS

# CY7C342

# 128-Macrocell MAX® EPLDs

## Features

- **128 macrocells in 8 LABs**
- **8 dedicated inputs, 52 bidirectional I/O pins**
- **Programmable interconnect array**
- **0.8-micron double-metal CMOS EPROM technology**
- **Available in 68-pin HLCC, PLCC, and PGA packages**

## Functional Description

The CY7C342 is an Erasable Programmable Logic Device (EPLD) in which CMOS EPROM cells are used to configure logic functions within the device. The MAX architecture is 100% user configurable, allowing the devices to accommodate a variety of independent logic functions.

The 128 macrocells in the CY7C342 are divided into 8 Logic Array Blocks (LABs), 16 per LAB. There are 256 expander product terms, 32 per LAB, to be used and shared by the macrocells within each LAB.

Each LAB is interconnected with a programmable interconnect array, allowing all signals to be routed throughout the chip.

The speed and density of the CY7C342 allows it to be used in a wide range of applications, from replacement of large amounts of 7400-series TTL logic, to complex controllers and multifunction chips. With greater than 25 times the functionality of 20-pin PLDs, the CY7C342 allows the replacement of over 50 TTL devices. By replacing large amounts of logic, the CY7C342 reduces board space, part count, and increases system reliability.

## Logic Array Blocks

There are 8 logic array blocks in the CY7C342. Each LAB consists of a macrocell array containing 16 macrocells, an expander product term array containing 32 expanders, and an I/O block. The LAB is fed by the programmable interconnect array and the dedicated input bus. All macrocell feedbacks go to the macrocell array, the expander array, and the programmable interconnect array. Expanders feed themselves and the macrocell array. All I/O feedbacks go to the programmable interconnect array so that they may be accessed by macrocells in other LABs as well as the macrocells in the LAB in which they are situated.

Externally, the CY7C342 provides eight dedicated inputs, one of which may be used as a system clock. There are 52 I/O pins that may be individually configured for input, output, or bidirectional data flow.

## Programmable Interconnect Array

The Programmable Interconnect Array (PIA) solves interconnect limitations by routing only the signals needed by each logic array block. The inputs to the PIA are the outputs of every macrocell within the device and the I/O pin feedback of every pin on the device.

Unlike masked or programmable gate arrays, which induce variable delay dependent on routing, the PIA has a fixed delay. This eliminates undesired skews among logic signals that may cause glitches in internal or external logic. The fixed delay, regardless of programmable interconnect array configuration, simplifies design by assuring that internal signal skews or races are avoided. The result is ease of design implementation, often in a signal pass, without the multiple internal logic placement and routing iterations required for a programmable gate array to achieve design timing objectives.

## Timing Delays

Timing delays within the CY7C342 may be easily determined using *Warp2®*, *Warp2Sim*™ or *Warp3®* software. The CY7C342 has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information the *Warp3* software provides a timing simulator.

## Selection Guide

| | | 7C342–25 | 7C342–30 | 7C342–35 |
|---|---|---|---|---|
| Maximum Access Time (ns) | | 25 | 30 | 35 |
| Maximum Operating Current (mA) | Commercial | 250 | 250 | 250 |
| | Military | 320 | 320 | 320 |
| | Industrial | 320 | 320 | 320 |
| Maximum Static Current (mA) | Commercial | 225 | 225 | 225 |
| | Military | 275 | 275 | 275 |
| | Industrial | 275 | 275 | 275 |

MAX is a registered trademark of Altera Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Coroporation.
*Warp2Sim* is a trademark of Cypress Secmiconductor Coroporation.

Document #: 38–00500

CYPRESS

# CY7C343
# CY7C343B

# 64-Macrocell MAX® EPLD

## Features

- **64 MAX macrocells in 4 LABs**
- **8 dedicated inputs, 24 bidirectional I/O pins**
- **Programmable interconnect array**
- **0.8-micron double-metal CMOS EPROM technology (CY7C343)**
- **Advanced 0.65-micron CMOS technology to increase performance (CY7C343B)**
- **Available in 44-pin HLCC, PLCC**
- **Lowest power MAX device**

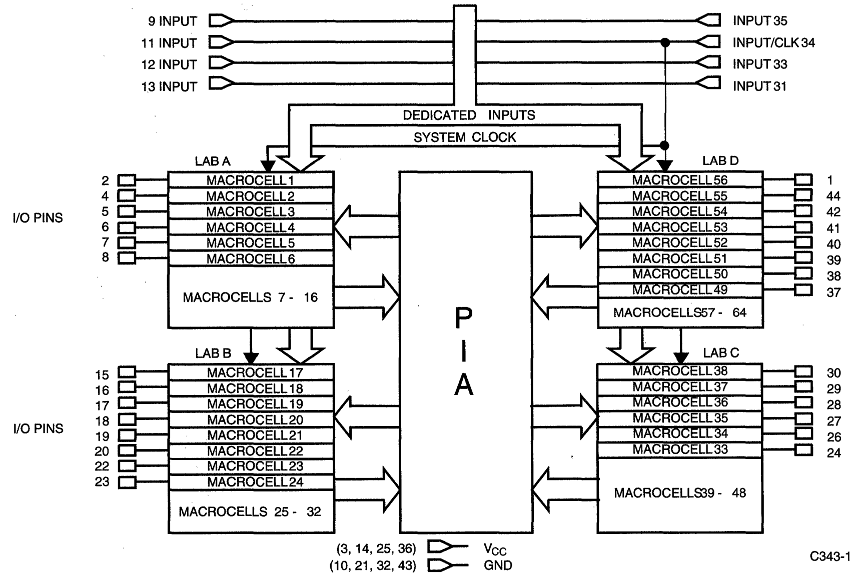
## Functional Description

The CY7C343/CY7C343B is a high-performance, high-density erasable programmable logic device, available in 44-pin PLCC and HLCC packages.

The CY7C343/CY7C343B contains 64 highly flexible macrocells and 128 expander product terms. These resources are divided into four Logic Array Blocks (LABs) connected through the Programmable Inter-connect Array (PIA). There are 8 input pins, one that doubles as a clock pin when needed. The CY7C343/CY7C343B also has 28 I/O pins, each connected to a macrocell (6 for LABs A and C, and 8 for LABs B and D). The remaining 36 macrocells are used for embedded logic.

The CY7C343/CY7C343B is excellent for a wide range of both synchronous and asynchronous applications.

## Logic Block Diagram



MAX is a registered trademark of Altera Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Coroporation.
*Warp2Sim* is a trademark of Cypress Secmiconductor Coroporation.

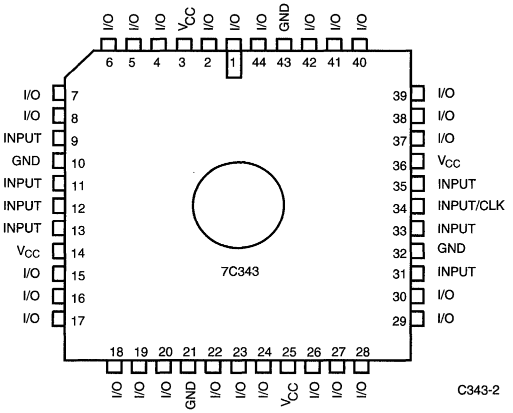
## Selection Guide

| | | 7C343-12<br>7C343B-12 | 7C343-15<br>7C343B-15 | 7C343-20<br>7C343B-20 | 7C343-25<br>7C343B-25 | 7C343-30<br>7C343B-30 | 7C343-35<br>7C343B-35 |
|---|---|---|---|---|---|---|---|
| Maximum Access Time (ns) | | 12 | 15 | 20 | 25 | 30 | 35 |
| Maximum Operating Current (mA) | Commercial | 135 | 135 | 135 | 135 | 135 | 135 |
| | Military | | 225 | 225 | 225 | 225 | 225 |
| | Industrial | 225 | 225 | 225 | 225 | 225 | 225 |
| Maximum Standby Current (mA) | Commercial | 125 | 125 | 125 | 125 | 125 | 125 |
| | Military | | 200 | 200 | 200 | 200 | 200 |
| | Industrial | 200 | 200 | 200 | 200 | 200 | 200 |

Shaded area contains preliminary information.

## Pin Configuration

HLCC
Top View



C343-2

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ..................................–65°C to+150°C

Ambient Temperature with
Power Applied ....................................................0°C to+70°C

Maximum Junction Temperature
(Under Bias)................................................................ 150°C

Supply Voltage to Ground Potential ................–2.0V to+7.0V

Maximum Power Dissipation..................................2500 mW

DC $V_{CC}$ or GND Current...........................................500 mA

DC Output Current, per Pin......................–25 mA to +25 mA

DC Input Voltage[1]........................................–3.0V to +7.0V

DC Program Voltage...................................................13.0V

Static Discharge Voltage..........................................>1100V
(per MIL–STD–883, method 3015)

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ±5% |
| Industrial | –40°C to +85°C | 5V ±10% |
| Military | –55°C to +125°C (Case) | 5V ±10% |

**Note:**

[1] Minimum DC input is –0.3V. During transitions, the inputs may undershoot to –2.0V for periods less than 20 ns.

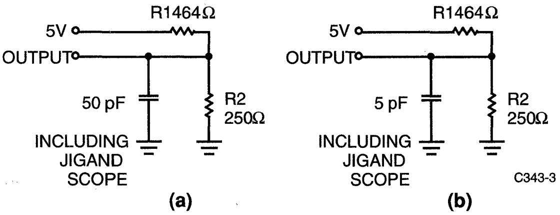
## Electrical Characteristics Over the Operating Range[2]

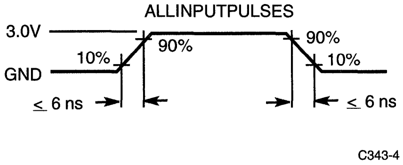
| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $I_{OH}$ = –4.0 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $I_{OL}$ = 8 mA | | | 0.45 | V |
| $V_{IH}$ | Input HIGH Level | | | 2.2 | $V_{CC}$+0.3 | V |
| $V_{IL}$ | Input LOW Level | | | –0.3 | 0.8 | V |
| $I_{IX}$ | Input Current | GND ≤ $V_{IN}$ ≤ $V_{CC}$ | | –10 | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = $V_{CC}$ or GND | | –40 | +40 | µA |
| $I_{OS}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[3, 4] | | –30 | –90 | mA |
| $I_{CC1}$ | Power Supply Current (Standby) | $V_I$ = $V_{CC}$ or GND (No Load) | Commercial | | 125 | mA |
| | | | Military/Industrial | | 200 | mA |
| $I_{CC2}$ | Power Supply Current[5] | $V_I$ = $V_{CC}$ or GND (No Load) f = 1.0 MHz[4, 5] | Commercial | | 135 | mA |
| | | | Military/Industrial | | 225 | mA |
| $t_R$ | Recommended Input Rise Time | | | | 100 | ns |
| $t_F$ | Recommended Input Fall Time | | | | 100 | ns |

## Capacitance[6]

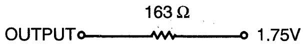
| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2V, f = 1.0 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V, f = 1.0 MHz | 10 | pF |

## AC Test Loads and Waveforms[6]





Equivalent to: THÉVENIN EQUIVALENT(commercial/military)



**Notes:**

2. Typical values are for $T_A$ = 25°C and $V_{CC}$ = 5V.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
4. Guaranteed but not 100% tested.
5. Measured with device programmed as a 16-bit counter in each LAB. This parameter is tested periodically by sampling production material.
6. Part (a) in AC Test Load and Waveforms is used for all parameters except $t_{ER}$ and $t_{XZ}$, which is used for part (b) in AC Test Load and Waveforms. All external timing parameters are measured referenced to external pins of the device.

## Programmable Interconnect Array

The Programmable Interconnect Array (PIA) solves interconnect limitations by routing only the signals needed by each logic array block. The inputs to the PIA are the outputs of every macrocell within the device and the I/O pin feedback of every pin on the device.

Unlike masked or programmable gate arrays, which induce variable delay dependent on routing, the PIA has a fixed delay. This eliminates undesired skews among logic signals, which may cause glitches in internal or external logic. The fixed delay, regardless of programmable interconnect array configuration, simplifies design by ensuring that internal signal skews or races are avoided. The result is simpler design implementation, often in a single pass, without the multiple internal logic placement and routing iterations required for a programmable gate array to achieve design timing objectives.

## Timing Delays

Timing delays within the CY7C343/CY7C343B may be easily determined using *Warp2*®, *Warp2Sim*™ or *Warp3*® software or by the model shown in *Figure 1*. The CY7C343/CY7C343B has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information, the *Warp3* software provides a timing simulator.

## Design Recommendations

Operation of the devices described herein with conditions above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this data sheet is not implied. Exposure to absolute maximum ratings conditions for extended periods of time may affect device reliability.The CY7C343/CY7C343B contains circuitry to protect device pins from high static voltages or electric fields; however, normal precautions should be taken to avoid applying any voltage higher than maximum rated voltages.

For proper operation, input and output pins must be constrained to the range GND ≤ ($V_{IN}$ or $V_{OUT}$) ≤ $V_{CC}$. Unused inputs must always be tied to an appropriate logic level (either $V_{CC}$ or GND). Each set of $V_{CC}$ and GND pins must be connected together directly at the device. Power supply decoupling capacitors of at least 0.2 µF must be connected between $V_{CC}$ and GND. For the most effective decoupling, each $V_{CC}$ pin should be separately decoupled to GND, directly at the device. Decoupling capacitors should have good frequency response, such as monolithic ceramic types.

## Timing Considerations

Unless otherwise stated, propagation delays do not include expanders. When using expanders, add the maximum expander delay $t_{EXP}$ to the overall delay. Similarly, there is an additional $t_{PIA}$ delay for an input from an I/O pin when compared to a signal from a straight input pin.

When calculating synchronous frequencies, use $t_{S1}$ if all inputs are on the input pins. $t_{S2}$ should be used if data is applied at an I/O pin. If $t_{S2}$ is greater than $t_{CO1}$, $1/t_{S2}$ becomes the limiting frequency in the data path mode unless $1/(t_{WH} + t_{WL})$ is less than $1/t_{S2}$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{S1}$. Determine which of $1/(t_{WH} + t_{WL})$, $1/t_{CO1}$, or $1/(t_{EXP} + t_{S1})$ is the lowest frequency. The lowest of these frequencies is the maximum data path frequency for the synchronous configuration.

When calculating external asynchronous frequencies, use $t_{AS1}$ if all inputs are on dedicated input pins. If any data is applied to an I/O pin, $t_{AS2}$ must be used as the required set-up time. If ($t_{AS2} + t_{AH}$) is greater than $t_{ACO1}$, $1/(t_{AS2} + t_{AH})$ becomes the limiting frequency in the data path mode unless $1/(t_{AWH} + t_{AH})$ is less than $1/(t_{AS2} + t_{AH})$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{AS1}$. Determine which of $1/(t_{AWH} + t_{AWL})$, $1/t_{ACO1}$, or $1/(t_{EXP} + t_{AS1})$ is the lowest frequency. The lowest of these frequencies is the maximum data path frequency for the asynchronous configuration.

The parameter $t_{OH}$ indicates the system compatibility of this device when driving other synchronous logic with positive input hold times, which is controlled by the same synchronous clock. If $t_{OH}$ is greater than the minimum required input hold time of the subsequent synchronous logic, then the devices are guaranteed to function properly with a common synchronous clock under worst-case environmental and supply voltage conditions.

The parameter $t_{AOH}$ indicates the system compatibility of this device when driving subsequent registered logic with a positive hold time and using the same clock as the CY7C343/CY7C343B.

In general, if $t_{AOH}$ is greater than the minimum required input hold time of the subsequent logic (synchronous or asynchronous), then the devices are guaranteed to function properly under worst-case environmental and supply voltage conditions, provided the clock signal source is the same. This also applies if expander logic is used in the clock signal path of the driving device, but not for the driven device. This is due to the expander logic in the second device's clock signal path adding an additional delay ($t_{EXP}$), causing the output data from the preceding device to change prior to the arrival of the clock signal at the following device's register.

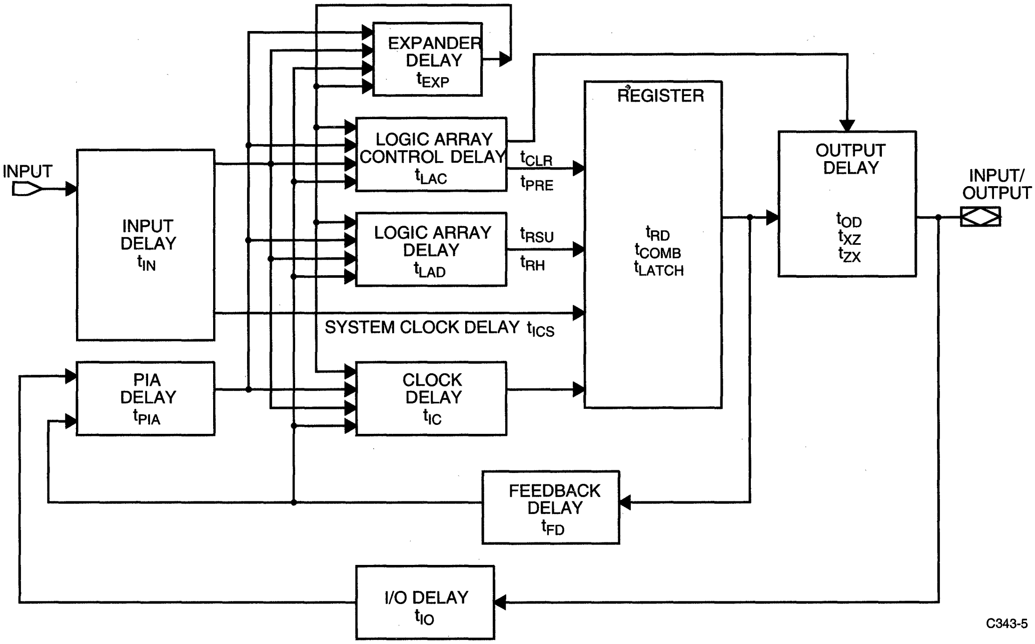


**Figure 1. CY7C343/CY7C343B Internal Timing Model**

## External Synchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | | 7C343-12<br>7C343B-12 | | 7C343-15<br>7C343B-15 | | 7C343-20<br>7C343B-20 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | Com'l /Ind | | 12 | | 15 | | 20 | ns |
| | | Mil | | | | 15 | | 20 | |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | Com'l /Ind | | 20 | | 25 | | 32 | ns |
| | | Mil | | | | 25 | | 32 | |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | Com'l /Ind | | 18 | | 23 | | 30 | ns |
| | | Mil | | | | 23 | | 30 | |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4, 10] | Com'l /Ind | | 26 | | 33 | | 42 | ns |
| | | | | | | 33 | | 42 | |
| $t_{EA}$ | Input to Output Enable Delay[4, 7] | Com'l /Ind | | 12 | | 15 | | 20 | ns |
| | | Mil | | | | 15 | | 20 | |
| $t_{ER}$ | Input to Output Disable Delay[4, 7] | Com'l /Ind | | 12 | | 15 | | 20 | ns |
| | | Mil | | | | 15 | | 20 | |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | Com'l /Ind | | 6 | | 7 | | 12 | ns |
| | | Mil | | | | 7 | | 12 | |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 11] | Com'l /Ind | | 14 | | 17 | | 25 | ns |
| | | Mil | | | | 17 | | | |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[7] | Com'l /Ind | 8 | | 10 | | 12 | | ns |
| | | Mil | | | 10 | | | | |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7, 12] | Com'l /Ind | 16 | | 20 | | 24 | | ns |
| | | Mil | | | 20 | | 24 | | |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | Com'l /Ind | 0 | | 0 | | 0 | | ns |
| | | Mil | | | 0 | | 0 | | |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | Com'l /Ind | 4.5 | | 5 | | 6 | | ns |
| | | Mil | | | 5 | | 6 | | |
| $t_{WL}$ | Synchronous Clock Input LOW Time | Com'l /Ind | 4.5 | | 5 | | 6 | | ns |
| | | Mil | | | 5 | | 6 | | |
| $t_{RW}$ | Asynchronous Clear Width[4, 7] | Com'l /Ind | 12 | | 15 | | 20 | | ns |
| | | Mil | | | 15 | | 20 | | |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4, 7] | Com'l /Ind | 12 | | 15 | | 20 | | ns |
| | | Mil | | | 15 | | 20 | | |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[7] | Com'l /Ind | | 12 | | 15 | | 20 | ns |
| | | Mil | | | | 15 | | 20 | |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4, 7] | Com'l /Ind | 12 | | 15 | | 20 | | ns |
| | | Mil | | | 15 | | 20 | | |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[7] | Com'l /Ind | | 12 | | 15 | | 20 | ns |
| | | Mil | | | | 15 | | 20 | |

## External Synchronous Switching Characteristics[6] Over Operating Range (continued)

| Parameter | Description | | 7C343-12<br>7C343B-12 | | 7C343-15<br>7C343B-15 | | 7C343-20<br>7C343B-20 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4, 13] | Com'l /Ind | | 3 | | 3 | | 3 | ns |
| | | Mil | | | | 3 | | 3 | |
| $t_P$ | External Synchronous Clock Period ($1/f_{MAX3}$)[4] | Com'l /Ind | 9 | | 10 | | 12 | | ns |
| | | Mil | | | 10 | | 12 | | |
| $f_{MAX1}$ | External Maximum Frequency ($1/(t_{CO1} + t_{S1})$)[4, 14] | Com'l /Ind | 71.4 | | 58.8 | | 41.6 | | MHz |
| | | Mil | | | 58.8 | | 41.6 | | |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of ($1/(t_{S1} + t_{CF})$) or ($1/t_{CO1}$)[4, 15] | Com'l /Ind | 90.9 | | 76.9 | | 66.6 | | MHz |
| | | Mil | | | 76.9 | | 66.6 | | |
| $f_{MAX3}$ | Data Path Maximum Frequency, least of $1/(t_{WL} + t_{WH})$, $1/(t_{S1} + t_H)$, or ($1/t_{CO1}$)[4, 16] | Com'l /Ind | 111.1 | | 100 | | 83.3 | | MHz |
| | | Mil | | | 100 | | 83.3 | | |
| $f_{MAX4}$ | Maximum Register Toggle Frequency ($1/(t_{WL}+t_{WH})$)[4, 17] | Com'l /Ind | 111.1 | | 100 | | 83.3 | | MHz |
| | | Mil | | | 100 | | 83.3 | | |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4, 18] | Com'l /Ind | 3 | | 3 | | 3 | | ns |
| | | Mil | | | 3 | | 3 | | |
| $t_{PW}$ | Asynchronous Preset Width[4, 7] | Com'l /Ind | 12 | | 15 | | 20 | | ns |
| | | Mil | | | 15 | | 20 | | |

Shaded area contains preliminary information.

**Notes:**

7. This specification is a measure of the delay from input signal applied to a dedicated input (44-pin PLCC input pin 9, 11, 12, 13, 31, 33, 34, or 35) to combinatorial output on any output pin. This delay assumes no expander terms are used to form the logic function. When this note is applied to any parameter specification it indicates that the signal (data, asynchronous clock, asynchronous clear, and/or asynchronous preset) is applied to a dedicated input only and no signal path (either clock or data) employs expander logic. If an input signal is applied to an I/O pin, an additional delay equal to $t_{PIA}$ should be added to the comparable delay for a dedicated input. If expanders are used, add the maximum expander delay $t_{EXP}$ to the overall delay for the comparable delay without expanders.
8. This specification is a measure of the delay from input signal applied to an I/O macrocell pin to any output. This delay assumes no expander terms are used to form the logic function.
9. This specification is a measure of the delay from an input signal applied to a dedicated input (44-pin PLCC input pin 9, 11, 12, 13, 31, 33, 34, or 35) to combinatorial output on any output pin. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.
10. This specification is a measure of the delay from an input signal applied to an I/O macrocell pin to any output. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.
11. This specification is a measure of the delay from synchronous register clock to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used, register is synchronously clocked and all feedback is within the same LAB. This parameter is tested periodically by sampling production material.
12. If data is applied to an I/O input for capture by a macrocell register, the I/O pin set-up time minimums should be observed. These parameters are $t_{S2}$ for synchronous operation and $t_{AS2}$ for asynchronous operation.
13. This specification is a measure of the delay associated with the internal register feedback path. This is the delay from synchronous clock to LAB logic array input. This delay plus the register set-up time, $t_{S1}$, is the minimum internal period for an internal synchronous state machine configuration. This delay is for feedback within the same LAB. This parameter is tested periodically by sampling production material.
14. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which a state machine configuration with external feedback can operate. It is assumed that all data inputs and feedback signals are applied to dedicated inputs.
15. This specification indicates the guaranteed maximum frequency at which a state machine, with internal-only feedback, can operate. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{CO1}$. All feedback is assumed to be local, originating within the same LAB..
16. This frequency indicates the maximum frequency at which the device may operate in data path mode. This delay assumes data input signals are applied to dedicated inputs and no expander logic is used.
17. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which an individual output or buried register can be cycled.
18. This parameter indicates the minimum time after a synchronous register clock input that the previous register output data is maintained on the output pin.

## External Synchronous Switching Characteristics[6] Over Operating Range (continued)

| Parameter | Description | | 7C343-25<br>7C343B-25 | | 7C343-30<br>7C343B-30 | | 7C343-35<br>7C343B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | Com'l /Ind | | 25 | | 30 | | 35 | ns |
| | | Mil | | 25 | | 30 | | 35 | |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | Com'l /Ind | | 39 | | 44 | | 53 | ns |
| | | Mil | | 39 | | 44 | | 53 | |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | Com'l /Ind | | 37 | | 44 | | 55 | ns |
| | | Mil | | 37 | | 44 | | 55 | |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4, 10] | Com'l/ Ind | | 51 | | 58 | | 73 | ns |
| | | Mil | | 51 | | 58 | | 73 | |
| $t_{EA}$ | Input to Output Enable Delay[4, 7] | Com'l /Ind | | 25 | | 30 | | 35 | ns |
| | | Mil | | 25 | | 30 | | 35 | |
| $t_{ER}$ | Input to Output Disable Delay[4, 7] | Com'l/ Ind | | 25 | | 30 | | 35 | ns |
| | | Mil | | 25 | | 30 | | 35 | |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | Com'l/ Ind | | 14 | | 16 | | 20 | ns |
| | | Mil | | 14 | | 16 | | 20 | |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 11] | Com'l/ Ind | | 30 | | 35 | | 42 | ns |
| | | Mil | | 30 | | 35 | | 42 | |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[7] | Com'l/ Ind | 15 | | 20 | | 25 | | ns |
| | | Mil | 15 | | 20 | | 25 | | |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7, 12] | Com'l/ Ind | 30 | | 35 | | 42 | | ns |
| | | Mil | 30 | | 35 | | 42 | | |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | Com'l/ Ind | 0 | | 0 | | 0 | | ns |
| | | Mil | 0 | | 0 | | 0 | | |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | Com'l/ Ind | 8 | | 10 | | 12.5 | | ns |
| | | Mil | 8 | | 10 | | 12.5 | | |
| $t_{WL}$ | Synchronous Clock Input LOW Time | Com'l/ Ind | 8 | | 10 | | 12.5 | | ns |
| | | Mil | 8 | | 10 | | 12.5 | | |
| $t_{RW}$ | Asynchronous Clear Width[4, 7] | Com'l /Ind | 25 | | 30 | | 35 | | ns |
| | | Mil | 25 | | 30 | | 35 | | |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4, 7] | Com'l/ Ind | 25 | | 30 | | 35 | | ns |
| | | Mil | 25 | | 30 | | 35 | | |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[7] | Com'l/ Ind | | 25 | | 30 | | 35 | ns |
| | | Mil | | 25 | | 30 | | 35 | |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4, 7] | Com'l/ Ind | 25 | | 30 | | 35 | | ns |
| | | Mil | 25 | | 30 | | 35 | | |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[7] | Com'l/ Ind | | 25 | | 30 | | 35 | ns |
| | | Mil | | 25 | | 30 | | 35 | |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4, 13] | Com'l/ Ind | | 3 | | 3 | | 5 | ns |
| | | Mil | | 3 | | 3 | | 5 | |
| $t_P$ | External Synchronous Clock Period ($1/f_{MAX3}$)[4] | Com'l/ Ind | 16 | | 20 | | 25 | | ns |
| | | Mil | 16 | | 20 | | 25 | | |

## External Synchronous Switching Characteristics[6] Over Operating Range (continued)

| Parameter | Description | | 7C343-25<br>7C343B-25 | | 7C343-30<br>7C343B-30 | | 7C343-35<br>7C343B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $f_{MAX1}$ | External Maximum Frequency $(1/(t_{CO1}+t_{S1}))$[4, 14] | Com'l/ Ind | 34 | | 27 | | 22.2 | | MHz |
| | | Mil | 34 | | 27 | | 22.2 | | |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of $(1/(t_{S1}+t_{CF}))$ or $(1/t_{CO1})$[4, 15] | Com'l /Ind | 55 | | 43 | | 33 | | MHz |
| | | Mil | 55 | | 43 | | 33 | | |
| $f_{MAX3}$ | Data Path Maximum Frequency, least of $1/(t_{WL}+t_{WH})$, $1/(t_{S1}+t_{H})$, or $(1/t_{CO1})$[4, 16] | Com'l /Ind | 62.5 | | 50 | | 40 | | MHz |
| | | Mil | 62.5 | | 50 | | 40 | | |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $(1/(t_{WL}+t_{WH}))$[4, 17] | Com'l/Ind | 62.5 | | 50 | | 40 | | MHz |
| | | Mil | 62.5 | | 50 | | 40 | | |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4, 18] | Com'l/ Ind | 3 | | 3 | | 3 | | ns |
| | | Mil | 3 | | 3 | | 3 | | |
| $t_{PW}$ | Asynchronous Preset Width[4, 7] | Com'l/ Ind | 25 | | 30 | | 35 | | ns |
| | | Mil | 25 | | 30 | | 35 | | |

## External Asynchronous Switching Characteristics Over Operating Range[6]

| Parameter | Description | | 7C343-12<br>7C343B-12 | | 7C343-15<br>7C343B-15 | | 7C343-20<br>7C343B-20 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] | Com'l/ Ind | | 12 | | 15 | | 20 | ns |
| | | Mil | | | | 15 | | 20 | |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | Com'l/ Ind | | 20 | | 25 | | 32 | ns |
| | | Mil | | | | 25 | | 32 | |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | Com'l/ Ind | 3 | | 3.5 | | 4 | | ns |
| | | Mil | | | 3.5 | | 4 | | |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | Com'l/ Ind | 12 | | 13.5 | | 15 | | ns |
| | | Mil | | | 13.5 | | 15 | | |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | Com'l/ Ind | 4 | | 4.5 | | 5 | | ns |
| | | Mil | | | 4.5 | | 5 | | |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | Com'l /Ind | 8 | | 8.5 | | 9 | | ns |
| | | Mil | | | 8.5 | | 9 | | |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | Com'l/ Ind | 6 | | 6.5 | | 7 | | ns |
| | | Mil | | | 6.5 | | 7 | | |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4, 21] | Com'l /Ind | | 9 | | 11 | | 13 | ns |
| | | Mil | | | | 11 | | 13 | |
| $t_{AP}$ | External Asynchronous Clock Period $(1/f_{MAXA4})$[4] | Com'l/ Ind | 14 | | 15 | | 16 | | ns |
| | | Mil | | | 15 | | 16 | | |

**Notes:**

[19] This specification is a measure of the delay from an asynchronous register clock input to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used in the logic of combinatorial output or the asynchronous clock input. The clock signal is applied to a dedicated input pin and all feedback is within a single LAB. This parameter is tested periodically by sampling production material.

[20] This parameter is measured with a positive-edge triggered clock at the register. For negative edge triggering, the $t_{AWH}$ and $t_{AWL}$ parameters must be swapped. If a given input is used to clock multiple registers with both positive and negative polarity, $t_{AWH}$ should be used for both $t_{AWH}$ and $t_{AWL}$.

[21] This specification is a measure of the delay associated with the internal register feedback path for an asynchronous clock to LAB logic array input. This delay plus the asynchronous register set-up time, $t_{AS1}$, is the minimum internal period for an internal asynchronously clocked state machine configuration. This delay is for feedback within the same LAB, assumes no expander logic in the clock path, and assumes that the clock input signal is applied to a dedicated input pin. This parameter is tested periodically by sampling production material.

## External Asynchronous Switching Characteristics Over Operating Range[6] (continued)

| Parameter | Description | | 7C343-12<br>7C343B-12 | | 7C343-15<br>7C343B-15 | | 7C343-20<br>7C343B-20 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $f_{MAXA1}$ | External Maximum Frequency in Asynchronous Mode 1/($t_{ACO1}$ + $t_{AS1}$)[4, 22] | Com'l/ Ind | 66.6 | | 54.0 | | 41.6 | | MHz |
| | | Mil | | | 54.0 | | 41.6 | | |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4, 23] | Com'l/ Ind | 71.4 | | 66.6 | | 58.8 | | MHz |
| | | Mil | | | 66.6 | | 58.8 | | |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4, 24] | Com'l/ Ind | 71.4 | | 66.6 | | 50 | | MHz |
| | | Mil | | | 66.6 | | 50 | | |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency 1/($t_{AWH}$ + $t_{AWL}$)[4, 25] | Com'l/ Ind | 71.4 | | 66.6 | | 62.5 | | MHz |
| | | Mil | | | 66.6 | | 62.5 | | |

Shaded area contains advanced information.

## External Asynchronous Switching Characteristics Over Operating Range[6]

| Parameter | Description | | 7C343-25<br>7C343B-25 | | 7C343-30<br>7C343B-30 | | 7C343-35<br>7C343B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4, 26] | Com'l/ Ind | 12 | | 12 | | 15 | | ns |
| | | Mil | | | 12 | | 15 | | |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] | Com'l/ Ind | | 25 | | 30 | | 35 | ns |
| | | Mil | | 25 | | 30 | | 35 | |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | Com'l/ Ind | | 40 | | 46 | | 55 | ns |
| | | Mil | | 40 | | 46 | | 55 | |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | Com'l/ Ind | 5 | | 6 | | 8 | | ns |
| | | Mil | 5 | | 6 | | 8 | | |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | Com'l/ Ind | 20 | | 25 | | 30 | | ns |
| | | Mil | 20 | | 25 | | 30 | | |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | Com'l/ Ind | 6 | | 8 | | 10 | | ns |
| | | Mil | 6 | | 8 | | 10 | | |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | Com'l/ Ind | 11 | | 14 | | 16 | | ns |
| | | Mil | 11 | | 14 | | 16 | | |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | Com'l/ Ind | 9 | | 11 | | 14 | | ns |
| | | Mil | 9 | | 11 | | 14 | | |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4, 21] | Com'l/ Ind | | 15 | | 18 | | 22 | ns |
| | | Mil | | 15 | | 18 | | 22 | |
| $t_{AP}$ | External Asynchronous Clock Period (1/$f_{MAXA4}$)[4] | Com'l/ Ind | 20 | | 25 | | 30 | | ns |
| | | Mil | 20 | | 25 | | 30 | | |

**Notes:**

[22] This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine configuration with external feedback can operate. It is assumed that all data inputs, clock inputs, and feedback signals are applied to dedicated inputs, and that no expander logic is employed in the clock signal path or data path.

[23] This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine with internal-only feedback can operate. This parameter is determined by the lesser of (1/$t_{ACF}$ + $t_{AS1}$)) or (1/($t_{AWH}$ +$t_{AWL}$)). If register output states must also control external points, this frequency can still be observed as long as this frequency is less than 1/$t_{ACO1}$.

[24] This frequency is the maximum frequency at which the device may operate in the asynchronously clocked data path mode. This specification is determined by the least of 1/($t_{AWH}$ + $t_{AWL}$), 1/($t_{AS1}$ + $t_{AH}$) or 1/$t_{ACO1}$. It assumes data and clock input signals are applied to dedicated input pins and no expander logic is used.

[25] This specification indicates the guaranteed maximum frequency at which an individual output or buried register can be cycled in asynchronously clocked mode by a clock signal applied to an external dedicated input pin.

[26] This parameter indicates the minimum time that the previous register output data is maintained on the output after an asynchronous register clock input.

## External Asynchronous Switching Characteristics Over Operating Range[6] (continued)

| Parameter | Description | | 7C343-25<br>7C343B-25 | | 7C343-30<br>7C343B-30 | | 7C343-35<br>7C343B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $f_{MAXA1}$ | External Maximum Frequency in Asynchronous Mode $1/(t_{ACO1} + t_{AS1})$[4, 22] | Com'l/ Ind | 33 | | 27 | | 23 | | MHz |
| | | Mil | 33 | | 27 | | 23 | | |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4, 23] | Com'l/ Ind | 50 | | 40 | | 33 | | MHz |
| | | Mil | 50 | | 40 | | 33 | | |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4, 24] | Com'l/ Ind | 40 | | 33 | | 28 | | MHz |
| | | Mil | 40 | | 33 | | 28 | | |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4, 25] | Com'l/ Ind | 50 | | 40 | | 33 | | MHz |
| | | Mil | 50 | | 40 | | 33 | | |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4, 26] | Com'l / Ind | 15 | | 15 | | 15 | | ns |
| | | Mil | 15 | | 15 | | 15 | | |

## Internal Switching Characteristics Over Operating Range[6]

| Parameter | Description | | 7C343-12<br>7C343B-12 | | 7C343-15<br>7C343B-15 | | 7C343-20<br>7C343B-20 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | Com'l / Ind | | 2.5 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | Com'l/ Ind | | 2.5 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_{EXP}$ | Expander Array Delay | Com'l/ Ind | | 6 | | 8 | | 10 | ns |
| | | Mil | | | | 8 | | 10 | |
| $t_{LAD}$ | Logic Array Data Delay | Com'l/ Ind | | 6 | | 8 | | 10 | ns |
| | | Mil | | | | 8 | | 10 | |
| $t_{LAC}$ | Logic Array Control Delay | Com'l/ Ind | | 5 | | 6 | | 8 | ns |
| | | Mil | | | | 6 | | 8 | |
| $t_{OD}$ | Output Buffer and Pad Delay | Com'l/ Ind | | 3 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | Com'l/ Ind | | 5 | | 6 | | 8 | ns |
| | | Mil | | | | 6 | | 8 | |
| $t_{XZ}$ | Output Buffer Disable Delay | Com'l/ Ind | | 5 | | 6 | | 8 | ns |
| | | Mil | | | | 6 | | 8 | |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | Com'l/ Ind | 2 | | 3 | | 4 | | ns |
| | | Mil | | | 3 | | 4 | | |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | Com'l/ Ind | 3 | | 3.5 | | 4 | | ns |
| | | Mil | | | 3.5 | | 4 | | |
| $t_{LATCH}$ | Flow-Through Latch Delay | Com'l /Ind | | 1 | | 1 | | 2 | ns |
| | | Mil | | | | 1 | | 2 | |
| $t_{RD}$ | Register Delay | Com'l/ Ind | | 1 | | 1 | | 1 | ns |
| | | Mil | | | | 1 | | 1 | |
| $t_{COMB}$ | Transparent Mode Delay[28] | Com'l/ Ind | | 1 | | 1 | | 2 | ns |
| | | Mil | | | | 1 | | 2 | |

Shaded area contains advanced information.

## Internal Switching Characteristics Over Operating Range[6] (continued)

| Parameter | Description | | 7C343-12<br>7C343B-12 | | 7C343-15<br>7C343B-15 | | 7C343-20<br>7C343B-20 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{CH}$ | Clock HIGH Time | Com'l/ Ind | 3 | | 4 | | 6 | | ns |
| | | Mil | | | 4 | | 6 | | |
| $t_{CL}$ | Clock LOW Time | Com'l/ Ind | 3 | | 4 | | 6 | | ns |
| | | Mil | | | 4 | | 6 | | |
| $t_{IC}$ | Asynchronous Clock Logic Delay | Com'l/ Ind | | 5 | | 7 | | 12 | ns |
| | | Mil | | | | 7 | | 12 | |
| $t_{ICS}$ | Synchronous Clock Delay | Com'l/ Ind | | 0.5 | | 0.5 | | 2 | ns |
| | | Mil | | | | 0.5 | | 2 | |
| $t_{FD}$ | Feedback Delay | Com'l/ Ind | | 1 | | 1 | | 1 | ns |
| | | Mil | | | | 1 | | 1 | |
| $t_{PRE}$ | Asynchronous Register Preset Time | Com'l/ Ind | | 3 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_{CLR}$ | Asynchronous Register Clear Time | Com'l/ Ind | | 3 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | Com'l /Ind | 2 | | 3 | | 4 | | ns |
| | | Mil | | | 3 | | 4 | | |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | Com'l/ Ind | 2 | | 3 | | 4 | | ns |
| | | Mil | | | 3 | | 4 | | |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | Com'l/ Ind | | 8 | | 10 | | 12 | ns |
| | | Mil | | | | 10 | | 12 | |

Shaded area contains preliminary information.

## Internal Switching Characteristics Over Operating Range[6]

| Parameter | Description | | 7C343-25<br>7C343B-25 | | 7C343-30<br>7C343B-30 | | 7C343-35<br>7C343B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | Com'l /Ind | | 5 | | 7 | | 9 | ns |
| | | Mil | | 5 | | 7 | | 9 | |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | Com'l/ Ind | | 5 | | 5 | | 7 | ns |
| | | Mil | | 5 | | 5 | | 7 | |
| $t_{EXP}$ | Expander Array Delay | Com'l/ Ind | | 12 | | 14 | | 20 | ns |
| | | Mil | | 12 | | 14 | | 20 | |
| $t_{LAD}$ | Logic Array Data Delay | Com'l/ Ind | | 12 | | 14 | | 16 | ns |
| | | Mil | | 12 | | 14 | | 16 | |
| $t_{LAC}$ | Logic Array Control Delay | Com'l/ Ind | | 10 | | 12 | | 13 | ns |
| | | Mil | | 10 | | 12 | | 13 | |
| $t_{OD}$ | Output Buffer and Pad Delay | Com'l /Ind | | 5 | | 5 | | 6 | ns |
| | | Mil | | 5 | | 5 | | 6 | |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | Com'l /Ind | | 10 | | 11 | | 13 | ns |
| | | Mil | | 10 | | 11 | | 13 | |
| $t_{XZ}$ | Output Buffer Disable Delay | Com'l /Ind | | 10 | | 11 | | 13 | ns |
| | | Mil | | 10 | | 11 | | 13 | |

## Internal Switching Characteristics Over Operating Range[6] (continued)

| Parameter | Description | | 7C343-25<br>7C343B-25 | | 7C343-30<br>7C343B-30 | | 7C343-35<br>7C343B-35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | Com'l/ Ind | 6 | | 8 | | 10 | | ns |
| | | Mil | 6 | | 8 | | 10 | | |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | Com'l/ Ind | 6 | | 8 | | 12 | | ns |
| | | Mil | 6 | | 8 | | 12 | | |
| $t_{LATCH}$ | Flow-Through Latch Delay | Com'l /Ind | | 3 | | 4 | | 4 | ns |
| | | Mil | | 3 | | 4 | | 4 | |
| $t_{RD}$ | Register Delay | Com'l /Ind | | 1 | | 2 | | 2 | ns |
| | | Mil | | 1 | | 2 | | 2 | |
| $t_{COMB}$ | Transparent Mode Delay[28] | Com'l/ Ind | | 3 | | 4 | | 4 | ns |
| | | Mil | | 3 | | 4 | | 4 | |
| $t_{CH}$ | Clock HIGH Time | Com'l /Ind | 8 | | 10 | | 12.5 | | ns |
| | | Mil | 8 | | 10 | | 12.5 | | |
| $t_{CL}$ | Clock LOW Time | Com'l /Ind | 8 | | 10 | | 12.5 | | ns |
| | | Mil | 8 | | 10 | | 12.5 | | |
| $t_{IC}$ | Asynchronous Clock Logic Delay | Com'l /Ind | | 14 | | 16 | | 18 | ns |
| | | Mil | | 14 | | 16 | | 18 | |
| $t_{ICS}$ | Synchronous Clock Delay | Com'l /Ind | | 2 | | 2 | | 3 | ns |
| | | Mil | | 2 | | 2 | | 3 | |
| $t_{FD}$ | Feedback Delay | Com'l /Ind | | 1 | | 1 | | 2 | ns |
| | | Mil | | 1 | | 1 | | 2 | |
| $t_{PRE}$ | Asynchronous Register Preset Time | Com'l /Ind | | 5 | | 6 | | 7 | ns |
| | | Mil | | 5 | | 6 | | 7 | |
| $t_{CLR}$ | Asynchronous Register Clear Time | Com'l /Ind | | 5 | | 6 | | 7 | ns |
| | | Mil | | 5 | | 6 | | 7 | |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | Com'l /Ind | 5 | | 6 | | 7 | | ns |
| | | Mil | 5 | | 6 | | 7 | | |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | Com'l/ Ind | 5 | | 6 | | 7 | | ns |
| | | Mil | 5 | | 6 | | 7 | | |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | Com'l/ Ind | | 14 | | 16 | | 20 | ns |
| | | Mil | | 14 | | 16 | | 20 | |

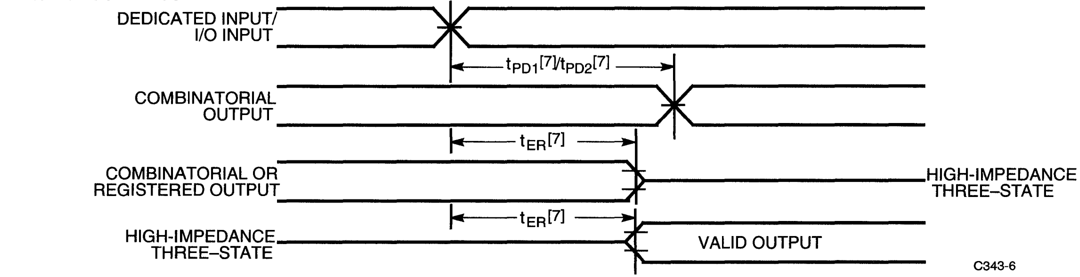
**Notes:**

[27] Sample tested only for an output change of 500 mV.

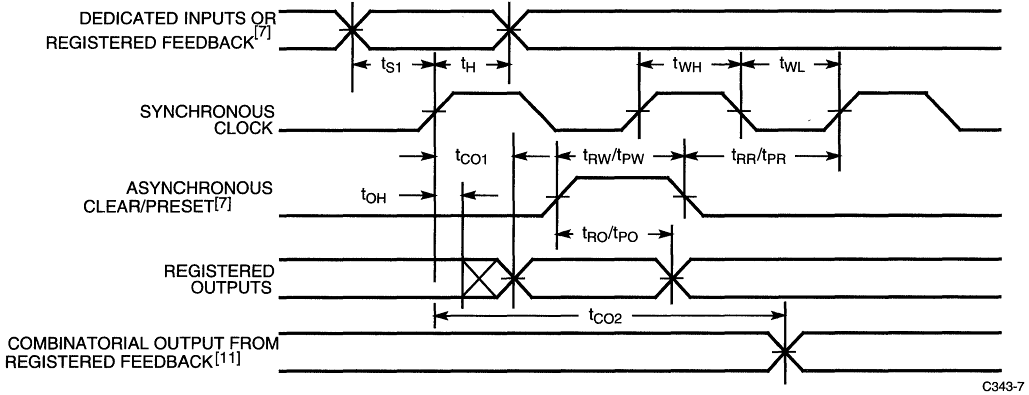
[28] This specification guarantees the maximum combinatorial delay associated with the macrocell register bypass when the macrocell is configured for combinatorial operation.

## Switching Waveforms

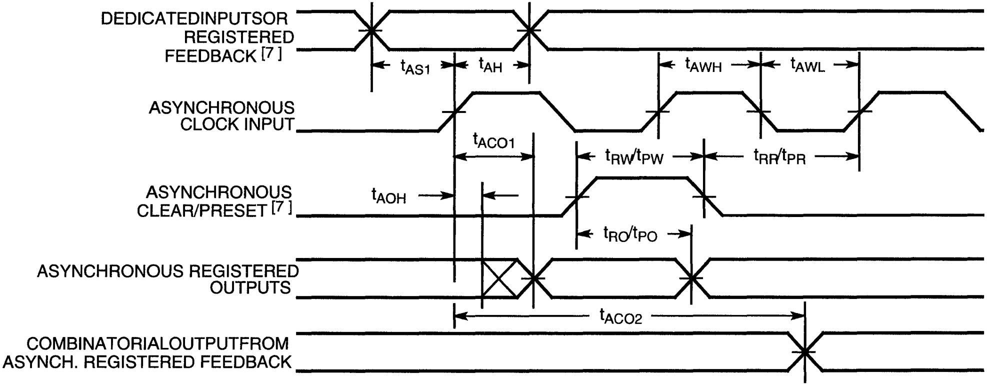
### External Combinatorial



### External Synchronous

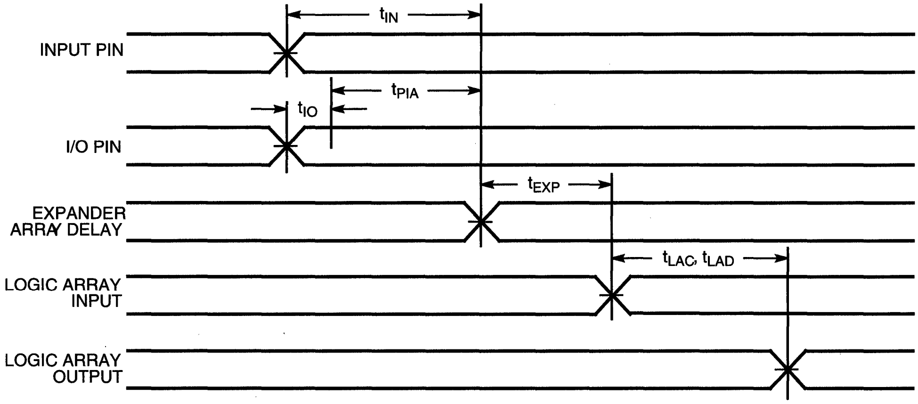


### External Asynchronous

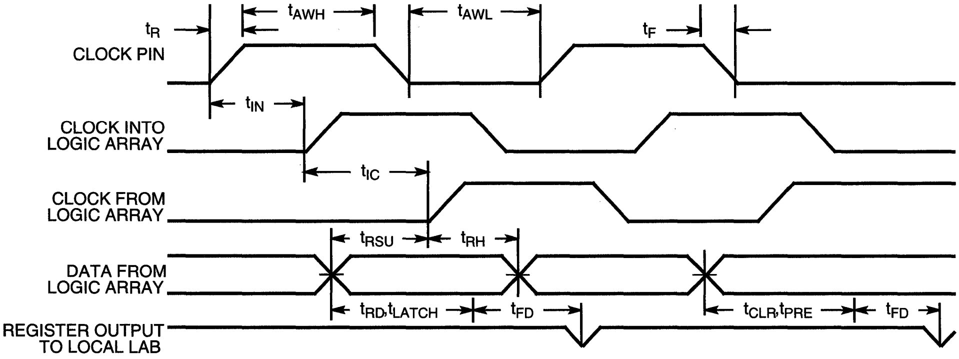


## Switching Waveforms (continued)

### Internal Combinatorial

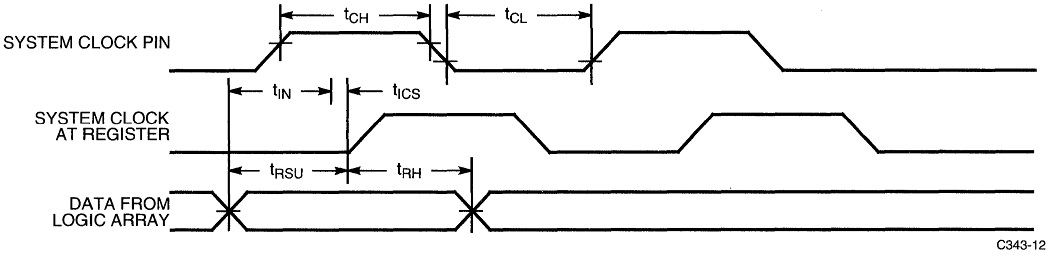


### Internal Asynchronous

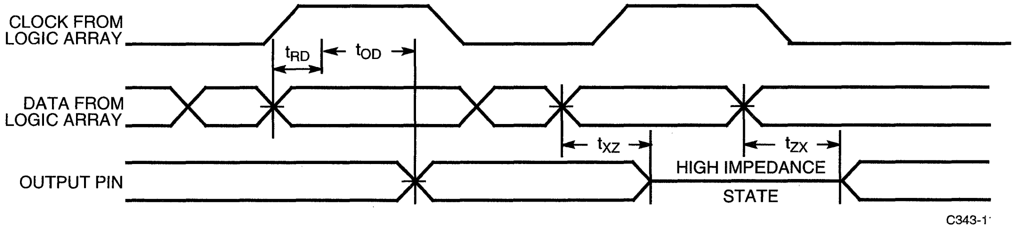


## Switching Waveforms (continued)

### Internal Synchronous



### Output Mode



## Ordering Information

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 12 | CY7C343B-12HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C343B-12JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| 15 | CY7C343B-15HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C343B-15JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343B-15HMB | H67 | 44-Pin Windowed Leaded Chip Carrier | Military |
| 20 | CY7C343-20JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | Commercial/Industrial |
| | CY7C343B-20HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | |
| | CY7C343B-20JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343B-20HMB | H67 | 44-Pin Windowed Leaded Chip Carrier | Military |
| 25 | CY7C343-25HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C343-25JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343B-25HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | |
| | CY7C343B-25JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| 30 | CY7C343-30HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C343-30JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343B-30HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | |
| | CY7C343B-30JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343-30HMB | H67 | 44-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C343B-30HMB | H67 | 44-Pin Windowed Leaded Chip Carrier | |
| 35 | CY7C343-35HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C343-35JC | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343B-35HC/HI | H67 | 44-Pin Windowed Leaded Chip Carrier | |
| | CY7C343B-35JC/JI | J67 | 44-Lead Plastic Leaded Chip Carrier | |
| | CY7C343-35HMB | H67 | 44-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C343B-35HMB | H67 | 44-Pin Windowed Leaded Chip Carrier | |

Shaded area contains preliminary information.

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameters | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

Document #: 38-00128-G

### Switching Characteristics

| Parameters | Subgroups |
|---|---|
| $t_{PD1}$ | 7, 8, 9, 10, 11 |
| $t_{PD2}$ | 7, 8, 9, 10, 11 |
| $t_{PD3}$ | 7, 8, 9, 10, 11 |
| $t_{CO1}$ | 7, 8, 9, 10, 11 |
| $t_S$ | 7, 8, 9, 10, 11 |
| $t_H$ | 7, 8, 9, 10, 11 |
| $t_{ACO1}$ | 7, 8, 9, 10, 11 |
| $t_{ACO2}$ | 7, 8, 9, 10, 11 |
| $t_{AS}$ | 7, 8, 9, 10, 11 |
| $t_{AH}$ | 7, 8, 9, 10, 11 |

CYPRESS

# CY7C344
# CY7C344B

## 32-Macrocell MAX® EPLD

### Features

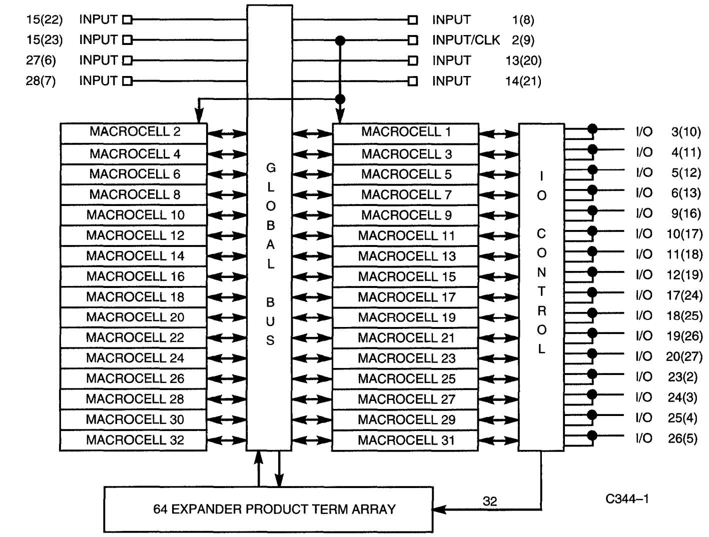
- High-performance, high-density replacement for TTL, 74HC, and custom logic
- 32 macrocells, 64 expander product terms in one LAB
- 8 dedicated inputs, 16 I/O pins
- 0.8-micron double-metal CMOS EPROM technology (CY7C344)
- Advanced 0.65-micron CMOS EPROM technology to increase performance (CY7C344B)
- 28-pin 300-mil DIP, cerDIP or 28-pin HLCC, PLCC package

### Functional Description

Available in a 28-pin 300-mil DIP or windowed J-leaded ceramic chip carrier (HLCC), the CY7C344/CY7C344B represents the densest EPLD of this size. Eight dedicated inputs and 16 bidirectional I/O pins communicate to one logic array block. In the CY7C344 LAB there are 32 macrocells and 64 expander product terms. When an I/O macrocell is used as an input, two expanders are used to create an input path. Even if all of the I/O pins are driven by macrocell registers, there are still 16 "buried" registers available. All inputs, macrocells, and I/O pins are interconnected within the LAB.

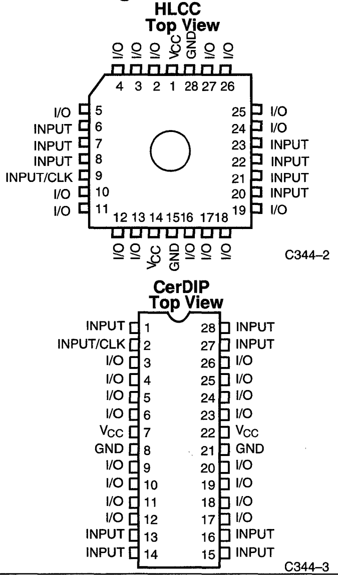
The speed and density of the CY7C344/CY7C344B makes it a natural for all types of applications. With just this one device, the designer can implement complex state machines, registered logic, and combinatorial "glue" logic, without using multiple chips. This architectural flexibility allows the CY7C344/CY7C344B to replace multichip TTL solutions, whether they are synchronous, asynchronous, combinatorial, or all three.

### Logic Block Diagram[1]



C344–1

### Pin Configurations



C344–2

C344–3

### Selection Guide

| | | 7C344B–10 | 7C344B–12 | 7C344–15<br>7C344B–15 | 7C344–20<br>7C344B–20 | 7C344–25<br>7C344B–25 |
|---|---|---|---|---|---|---|
| Maximum Access Time (ns) | | 10 | 12 | 15 | 20 | 25 |
| Maximum Operating Current (mA) | Commercial | 200 | 200 | 200 | 200 | 200 |
| | Military | | 220 | | 220 | 220 |
| | Industrial | | 220 | 220 | 220 | 220 |
| Maximum Standby Current (mA) | Commercial | 150 | 150 | 150 | 150 | 150 |
| | Military | | 170 | | 170 | 170 |
| | Industrial | | 170 | 170 | 170 | 170 |

Shaded area contains preliminary information.

**Note:**

[1] Numbers in () refer to J-leaded packages.

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ................................ –65°C to +150°C

Ambient Temperature with
Power Applied ................................................ 0°C to +70°C

Maximum Junction Temperature (Under Bias)............ 150°C

Supply Voltage to Ground Potential ............... –2.0V to +7.0V

Maximum Power Dissipation .................................. 1500 mW

DC $V_{CC}$ or GND Current ........................................... 500 mA

Static Discharge Voltage
(per MIL-STD-883, Method 3015) ............................. >2001V

DC Output Current, per Pin ..................... –25 mA to +25 mA

DC Input Voltage[2] ......................................... –3.0V to +7.0V

DC Program Voltage ................................................ +13.0V

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ±5% |
| Industrial | –40°C to +85°C | 5V ±10% |
| Military | –55°C to +125°C (Case) | 5V ±10% |

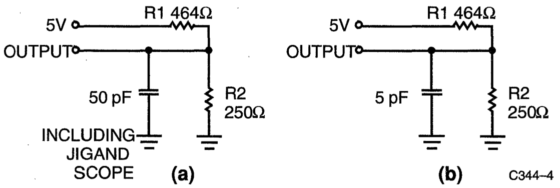
## Electrical Characteristics Over the Operating Range[3]

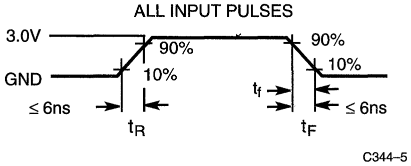
| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $I_{OH}$ = –4.0 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $I_{OL}$ = 8 mA | | | 0.45 | V |
| $V_{IH}$ | Input HIGH Level | | | 2.2 | $V_{CC}$+0.3 | V |
| $V_{IL}$ | Input LOW Level | | | –0.3 | 0.8 | V |
| $I_{IX}$ | Input Current | GND ≤ $V_{IN}$ ≤ $V_{CC}$ | | –10 | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = $V_{CC}$ or GND | | –40 | +40 | µA |
| $I_{OS}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[4, 5] | | –30 | –90 | mA |
| $I_{CC1}$ | Power Supply Current (Standby) | $V_I$ = $V_{CC}$ or GND (No Load) | Commercial | | 150 | mA |
| | | | Military/Industrial | | 170 | mA |
| $I_{CC2}$ | Power Supply Current | $V_I$ = $V_{CC}$ or GND (No Load) f = 1.0 MHz[4,6] | Commercial | | 200 | mA |
| | | | Military/Industrial | | 220 | mA |
| $t_R$ | Recommended Input Rise Time | | | | 100 | ns |
| $t_F$ | Recommended Input Fall Time | | | | 100 | ns |

## Capacitance

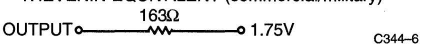
| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2V, f = 1.0 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0V, f = 1.0 MHz | 10 | pF |

## AC Test Loads and Waveforms[7]





Equivalent to: THÉVENIN EQUIVALENT (commercial/military)



**Notes:**

[2] Minimum DC input is –0.3V. During transitions, the inputs may undershoot to –2.0V for periods less than 20 ns.

[3] Typical values are for $T_A$ = 25°C and $V_{CC}$ = 5V.

[4] Guaranteed by design but not 100% tested.

[5] Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.

[6] Measured with device programmed as a 16-bit counter.

[7] Part (a) in AC Test Load and Waveforms is used for all parameters except $t_{ER}$ and $t_{XZ}$, which is used for part (b) in AC Test Load and Waveforms. All external timing parameters are measured referenced to external pins of the device.

## Timing Delays

Timing delays within the CY7C344/CY7C344B may be easily determined using *Warp2*®, *Warp2Sim*™, or *Warp3*® software or by the model shown in *Figure 1*. The CY7C344/CY7C344B has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information, the *Warp3* software provides a timing simulator.

## Design Recommendations

Operation of the devices described herein with conditions above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this datasheet is not implied. Exposure to absolute maximum ratings conditions for extended periods of time may affect device reliability. The CY7C344/CY7C344B contains circuitry to protect device pins from high-static voltages or electric fields; however, normal precautions should be taken to avoid applying any voltage higher than maximum rated voltages.

For proper operation, input and output pins must be constrained to the range GND ≤ (VIN or VOUT) ≤ VCC. Unused inputs must always be tied to an appropriate logic level (either VCC or GND). Each set of VCC and GND pins must be connected together directly at the device. Power supply decoupling capacitors of at least 0.2 µF must be connected between VCC and GND. For the most effective decoupling, each VCC pin should be separately decoupled.

## Timing Considerations

Unless otherwise stated, propagation delays do not include expanders. When using expanders, add the maximum expander delay $t_{EXP}$ to the overall delay.

When calculating synchronous frequencies, use $t_{S1}$ if all inputs are on the input pins. $t_{S2}$ should be used if data is applied at an I/O pin. If $t_{S2}$ is greater than $t_{CO1}$, $1/t_{S2}$ becomes the limiting frequency in the data-path mode unless $1/(t_{WH} + t_{WL})$ is less than $1/t_{S2}$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{S1}$. Determine which of $1/(t_{WH} + t_{WL})$, $1/t_{CO1}$, or $1/(t_{EXP} + t_{S1})$ is the lowest frequency. The lowest of these frequencies is the maximum data-path frequency for the synchronous configuration.

When calculating external asynchronous frequencies, use $t_{AS1}$ if all inputs are on dedicated input pins. If any data is applied to an I/O pin, $t_{AS2}$ must be used as the required set-up time. If $(t_{AS2} + t_{AH})$ is greater than $t_{ACO1}$, $1/(t_{AS2} + t_{AH})$ becomes the limiting frequency in the data-path mode unless $1/(t_{AWH} + t_{AWL})$ is less than $1/(t_{AS2} + t_{AH})$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{AS1}$. Determine which of $1/(t_{AWH} + t_{AWL})$, $1/t_{ACO1}$, or $1/(t_{EXP} + t_{AS1})$ is the lowest frequency. The lowest of these frequencies is the maximum data-path frequency for the asynchronous configuration.

The parameter $t_{OH}$ indicates the system compatibility of this device when driving other synchronous logic with positive input hold times, which is controlled by the same synchronous clock. If $t_{OH}$ is greater than the minimum required input hold time of the subsequent synchronous logic, then the devices are guaranteed to function properly with a common synchronous clock under worst-case environmental and supply voltage conditions.

The parameter $t_{AOH}$ indicates the system compatibility of this device when driving subsequent registered logic with a positive hold time and using the same clock as the CY7C344/CY7C344B.In general, if $t_{AOH}$ is greater than the minimum required input hold time of the subsequent logic (synchronous or asynchronous), then the devices are guaranteed to function properly under worst-case environmental and supply voltage conditions, provided the clock signal source is the same. This also applies if expander logic is used in the clock signal path of the driving device, but not for the driven device. This is due to the expander logic in the second device's clock signal path adding an additional delay ($t_{EXP}$), causing the output data from the preceding device to change prior to the arrival of the clock signal at the following device's register.

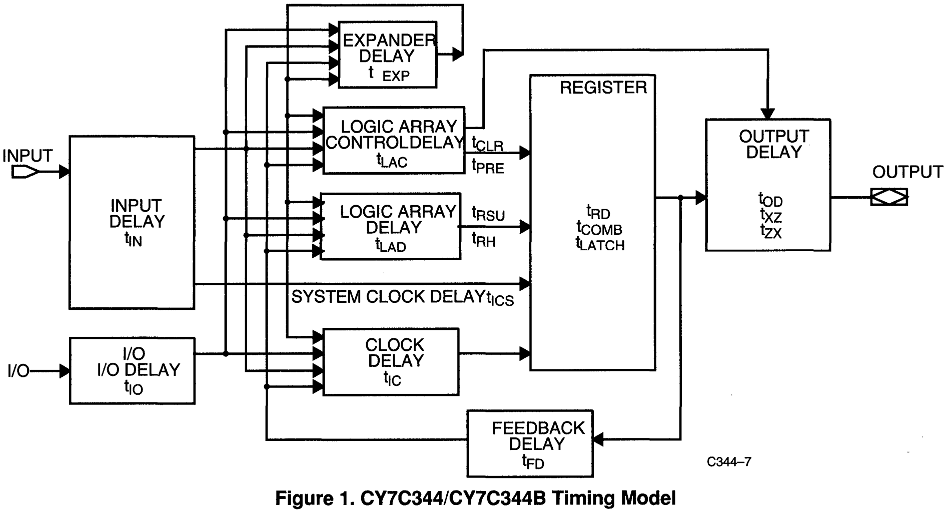


**Figure 1. CY7C344/CY7C344B Timing Model**

## External Synchronous Switching Characteristics[7] Over Operating Range

| | | | 7C344B–10 | | 7C344B–12 | | 7C344–15<br>7C344B–15 | | |
|---|---|---|---|---|---|---|---|---|---|
| Parameter | Description | | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[8] | Com'l /Ind | | 10 | | 12 | | 15 | ns |
| | | Mil | | | | 12 | | 15 | |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[9] | Com'l/Ind | | 10 | | 12 | | 15 | ns |
| | | Mil | | | | 12 | | 15 | |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[10] | Com'l /Ind | | 16 | | 18 | | 30 | ns |
| | | Mil | | | | 18 | | 30 | |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4, 11] | Com'l/Ind | | 16 | | 18 | | 30 | ns |
| | | Mil | | | | 18 | | 30 | |
| $t_{EA}$ | Input to Output Enable Delay[4] | Com'l/Ind | | 10 | | 12 | | 20 | ns |
| | | Mil | | | | 12 | | 20 | |
| $t_{ER}$ | Input to Output Disable Delay[4] | Com'l /Ind | | 10 | | 12 | | 20 | ns |
| | | Mil | | | | 12 | | 20 | |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | Com'l /Ind | | 5 | | 6 | | 10 | ns |
| | | Mil | | | | 6 | | 10 | |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 12] | Com'l /Ind | | 10 | | 12 | | 20 | ns |
| | | Mil | | | | 12 | | 20 | |
| $t_S$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input | Com'l/Ind | 6 | | 8 | | 10 | | ns |
| | | Mil | | | 8 | | 10 | | |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | Com'l /Ind | 0 | | 0 | | 0 | | ns |
| | | Mil | | | 0 | | 0 | | |
| $t_{WH}$ | Synchronous Clock Input HIGH Time[4] | Com'l/Ind | 4 | | 4.5 | | 6 | | ns |
| | | Mil | | | 4.5 | | 6 | | |
| $t_{WL}$ | Synchronous Clock Input LOW Time[4] | Com'l /Ind | 4 | | 4.5 | | 6 | | ns |
| | | Mil | | | 4.5 | | 6 | | |
| $t_{RW}$ | Asynchronous Clear Width[4] | Com'l /Ind | 10 | | 12 | | 20 | | ns |
| | | Mil | | | 12 | | 20 | | |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4] | Com'l /Ind | 10 | | 12 | | 20 | | ns |
| | | Mil | | | 12 | | 20 | | |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[4] | Com'l /Ind | | 10 | | 12 | | 15 | ns |
| | | Mil | | | | 12 | | 15 | |
| $t_{PW}$ | Asynchronous Preset Width[4] | Com'l /Ind | 10 | | 12 | | 20 | | ns |
| | | Mil | | | 12 | | 20 | | |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4] | Com'l /Ind | 10 | | 12 | | 20 | | ns |
| | | Mil | | | 12 | | 20 | | |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[4] | Com'l /Ind | | 10 | | 12 | | 15 | ns |
| | | Mil | | | | 12 | | 15 | |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4, 13] | Com'l /Ind | | 3 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_P$ | External Synchronous Clock Period ($1/f_{MAX3}$)[4] | Com'l/Ind | 8 | | 9 | | 13 | | ns |
| | | Mil | | | 9 | | 13 | | |
| $f_{MAX1}$ | External Maximum Frequency($1/(t_{CO1} + t_S)$)[4, 14] | Com'l/Ind | 90.9 | | 71.4 | | 50.0 | | MHz |
| | | Mil | | | 71.4 | | 50.0 | | |

Shaded area contains preliminary information.

## External Synchronous Switching Characteristics[7] Over Operating Range (continued)

| Parameter | Description | | 7C344B–10 | | 7C344B–12 | | 7C344–15<br>7C344B–15 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $f_{MAX2}$ | Maximum Frequency with Internal Only Feedback $(1/(t_{CF} + t_S))$[4, 15] | Com'l/Ind | 111.1 | | 90.9 | | 71.4 | | MHz |
| | | Mil | | | 90.9 | | 71.4 | | |
| $f_{MAX3}$ | Data Path Maximum Frequency, least of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or $(1/t_{CO1})$[4, 16] | Com'l/Ind | 125.0 | | 111.1 | | 83.3 | | MHz |
| | | Mil | | | 111.1 | | 83.3 | | |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $1/(t_{WL} + t_{WH})$[4, 17] | Com'l/Ind | 125.0 | | 111.1 | | 83.3 | | MHz |
| | | Mil | | | 111.1 | | 83.3 | | |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4, 18] | Com'l/ Ind | 3 | | 3 | | 3 | | ns |
| | | Mil | | | 3 | | 3 | | |

Shaded area contains preliminary information.

**Notes:**

8. This parameter is the delay from an input signal applied to a dedicated input pin to a combinatorial output on any output pin. This delay assumes no expander terms are used to form the logic function.
9. This parameter is the delay associated with an input signal applied to an I/O macrocell pin to any output. This delay assumes no expander terms are used to form the logic function.
10. This parameter is the delay associated with an input signal applied to a dedicated input pin to combinatorial output on any output pin. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.
11. This parameter is the delay associated with an input signal applied to an I/O macrocell pin to any output pin. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.
12. This specification is a measure of the delay from synchronous register clock input to internal feedback of the register output signal to a combinatorial output for which the registered output signal is used as an input. This parameter assumes no expanders are used in the logic of the combinatorial output and the register is synchronously clocked. This parameter is tested periodically by sampling production material.
13. This specification is a measure of the delay associated with the internal register feedback path. This delay plus the register set-up time, $t_S$, is the minimum internal period for an internal state machine configuration. This parameter is tested periodically by sampling production material.
14. This specification indicates the guaranteed maximum frequency at which a state machine configuration with external only feedback can operate.
15. This specification indicates the guaranteed maximum frequency at which a state machine with internal-only feedback can operate. If register output states must also control external points, this frequency can still be observed as long as it is less than $1/t_{CO1}$. This specification assumes no expander logic is used. This parameter is tested periodically by sampling production material.
16. This frequency indicates the maximum frequency at which the device may operate in data-path mode (dedicated input pin to output pin). This assumes that no expander logic is used.
17. This specification indicates the guaranteed maximum frequency in synchronous mode, at which an individual output or buried register can be cycled by a clock signal applied to either a dedicated input pin or an I/O pin.
18. This parameter indicates the minimum time after a synchronous register clock input that the previous register output data is maintained on the output pin.

## External Synchronous Switching Characteristics[7] Over Operating Range (continued)

| Parameter | Description | | 7C344–20<br>7C344B–20 | | 7C344–25<br>7C344B–25 | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[8] | Com'l /Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[9] | Com'l/Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[10] | Com'l /Ind | | 30 | | 40 | ns |
| | | Mil | | 30 | | 40 | |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4, 11] | Com'l/Ind | | 30 | | 40 | ns |
| | | Mil | | 30 | | 40 | |
| $t_{EA}$ | Input to Output Enable Delay[4] | Com'l/Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{ER}$ | Input to Output Disable Delay[4] | Com'l /Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | Com'l /Ind | | 12 | | 15 | ns |
| | | Mil | | 12 | | 15 | |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 12] | Com'l /Ind | | 22 | | 29 | ns |
| | | Mil | | 22 | | 29 | |
| $t_S$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input | Com'l/Ind | 12 | | 15 | | ns |
| | | Mil | 12 | | 15 | | |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | Com'l /Ind | 0 | | 0 | | ns |
| | | Mil | 0 | | 0 | | |
| $t_{WH}$ | Synchronous Clock Input HIGH Time[4] | Com'l/Ind | 7 | | 8 | | ns |
| | | Mil | 7 | | 8 | | |
| $t_{WL}$ | Synchronous Clock Input LOW Time[4] | Com'l /Ind | 7 | | 8 | | ns |
| | | Mil | 7 | | 8 | | |
| $t_{RW}$ | Asynchronous Clear Width[4] | Com'l /Ind | 20 | | 25 | | ns |
| | | Mil | 20 | | 25 | | |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4] | Com'l /Ind | 20 | | 25 | | ns |
| | | Mil | 20 | | 25 | | |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[4] | Com'l /Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{PW}$ | Asynchronous Preset Width[4] | Com'l /Ind | 20 | | 25 | | ns |
| | | Mil | 20 | | 25 | | |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4] | Com'l /Ind | 20 | | 25 | | ns |
| | | Mil | 20 | | 25 | | |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[4] | Com'l /Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4, 13] | Com'l /Ind | | 4 | | 7 | ns |
| | | Mil | | 4 | | 7 | |
| $t_P$ | External Synchronous Clock Period ($1/f_{MAX3}$)[4] | Com'l/Ind | 14 | | 16 | | ns |
| | | Mil | 14 | | 16 | | |

## External Synchronous Switching Characteristics[7] Over Operating Range (continued)

| Parameter | Description | | 7C344–20<br>7C344B–20 | | 7C344–25<br>7C344B–25 | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| $f_{MAX1}$ | External Maximum Frequency($1/(t_{CO1} + t_S)$)[4, 14] | Com'l/Ind | 41.6 | | 33.3 | | MHz |
| | | Mil | 41.6 | | 33.3 | | |
| $f_{MAX2}$ | Maximum Frequency with Internal Only Feedback ($1/(t_{CF} + t_S)$)[4, 15] | Com'l/Ind | 62.5 | | 45.4 | | MHz |
| | | Mil | 62.5 | | 45.4 | | |
| $f_{MAX3}$ | Data Path Maximum Frequency, least of $1/(t_{WL} + t_{WH})$, $1/(t_S + t_H)$, or ($1/t_{CO1}$)[4, 16] | Com'l/Ind | 71.4 | | 62.5 | | MHz |
| | | Mil | 71.4 | | 62.5 | | |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $1/(t_{WL} + t_{WH})$[4, 17] | Com'l/Ind | 71.4 | | 62.5 | | MHz |
| | | Mil | 71.4 | | 62.5 | | |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4, 18] | Com'l/ Ind | 3 | | 3 | | ns |
| | | Mil | 3 | | 3 | | |

## External Asynchronous Switching Characteristics Over Operating Range[7]

| Parameter | Description | | 7C344B–10 | | 7C344B–12 | | 7C344–15<br>7C344B–15 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay | Com'l/ Ind | | 10 | | 12 | | 15 | ns |
| | | Mil | | | | 12 | | 15 | |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | Com'l/Ind | | 15 | | 18 | | 30 | ns |
| | | Mil | | | | 18 | | 30 | |
| $t_{AS}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input | Com'l/Ind | 4 | | 4 | | 7 | | ns |
| | | Mil | | | 4 | | 7 | | |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input | Com'l/Ind | 3 | | 4 | | 7 | | ns |
| | | Mil | | | 4 | | 7 | | |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[4, 20] | Com'l/Ind | 4 | | 5 | | 6 | | ns |
| | | Mil | | | 5 | | 6 | | |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[4] | Com'l/Ind | 5 | | 6 | | 7 | | ns |
| | | Mil | | | 6 | | 7 | | |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4, 21] | Com'l /Ind | | 7 | | 9 | | 18 | ns |
| | | Mil | | | | 9 | | 18 | |
| $t_{AP}$ | External Asynchronous Clock Period ($1/f_{MAX4}$)[4] | Com'l/Ind | 12 | | 12.5 | | 13 | | ns |
| | | Mil | | | 12.5 | | 13 | | |
| $f_{MAXA1}$ | External Maximum Frequency in Asynchronous Mode $1/(t_{ACO1} + t_{AS})$[4, 22] | Com'l/Ind | 71.4 | | 62.5 | | 45.4 | | MHz |
| | | Mil | | | 62.5 | | 45.4 | | |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency $1/(t_{ACF} + t_{AS})$ or $1/(t_{AWH} + t_{AWL})$[4, 23] | Com'l/Ind | 90.9 | | 76.9 | | 40 | | MHz |
| | | Mil | | | 76.9 | | 40 | | |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4, 24] | Com'l/Ind | 100.0 | | 83.3 | | 66.6 | | MHz |
| | | Mil | | | 83.3 | | 66.6 | | |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4, 25] | Com'l /Ind | 111.1 | | 90.9 | | 76.9 | | MHz |
| | | Mil | | | 90.9 | | 76.9 | | |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4, 26] | Com'l/Ind | 12 | | 12 | | 15 | | ns |
| | | Mil | | | | | 15 | | |

Shaded area contains preliminary information.

## External Asynchronous Switching Characteristics Over Operating Range[7] (continued)

| | | | 7C344–20<br>7C344B–20 | | 7C344–25<br>7C344B–25 | | |
|---|---|---|---|---|---|---|---|
| Parameter | Description | | Min. | Max. | Min. | Max. | Unit |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay | Com'l/ Ind | | 20 | | 25 | ns |
| | | Mil | | 20 | | 25 | |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | Com'l/Ind | | 30 | | 37 | ns |
| | | Mil | | 30 | | 37 | |
| $t_{AS}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input | Com'l/Ind | 9 | | 12 | | ns |
| | | Mil | 9 | | 12 | | |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input | Com'l/Ind | 9 | | 12 | | ns |
| | | Mil | 9 | | 12 | | |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[4, 20] | Com'l/Ind | 7 | | 9 | | ns |
| | | Mil | 7 | | 9 | | |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[4] | Com'l/Ind | 9 | | 11 | | ns |
| | | Mil | 9 | | 11 | | |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4, 21] | Com'l /Ind | | 18 | | 21 | ns |
| | | Mil | | 18 | | 21 | |
| $t_{AP}$ | External Asynchronous Clock Period ($1/f_{MAX4}$)[4] | Com'l/Ind | 16 | | 20 | | ns |
| | | Mil | 16 | | 20 | | |
| $f_{MAXA1}$ | External Maximum Frequency in Asynchronous Mode $1/(t_{ACO1} + t_{AS})$[4, 22] | Com'l/Ind | 34.4 | | 27 | | MHz |
| | | Mil | 34.4 | | 27 | | |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency $1/(t_{ACF} + t_{AS})$ or $1/(t_{AWH} + t_{AWL})$[4, 23] | Com'l/Ind | 37 | | 30.3 | | MHz |
| | | Mil | 37 | | 30.3 | | |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4, 23] | Com'l/Ind | 50 | | 40 | | MHz |
| | | Mil | 50 | | 40 | | |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4, 25] | Com'l /Ind | 62.5 | | 50 | | MHz |
| | | Mil | 62.5 | | 50 | | |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4, 26] | Com'l/Ind | 15 | | 15 | | ns |
| | | Mil | 15 | | 15 | | |

**Notes:**

[19] This specification is a measure of the delay from an asynchronous register clock input to internal feedback of the registered output signal to a combinatorial output for which the registered output signal is used as an input. Assumes no expanders are used in logic of combinatorial output or the asynchronous clock input. This parameter is tested periodically by sampling production material.

[20] This parameter is measured with a positive-edge-triggered clock at the register. For negative edge triggering, the $t_{AWH}$ and $t_{AWL}$ parameters must be swapped. If a given input is used to clock multiple registers with both positive and negative polarity, $t_{AWH}$ should be used for both $t_{AWH}$ and $t_{AWL}$.

[21] This specification is a measure of the delay associated with the internal register feedback path for an asynchronously clocked register. This delay plus the asynchronous register set-up time, $t_{AS}$, is the minimum internal period for an asynchronously clocked state machine configuration. This delay assumes no expander logic in the asynchronous clock path. This parameter is tested periodically by sampling production material.

[22] This parameter indicates the guaranteed maximum frequency at which an asynchronously clocked state machine configuration with external feedback can operate. It is assumed that no expander logic is employed in the clock signal path or data path.

[23] This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine with internal-only feedback can operate. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{ACO1}$. This specification assumes no expander logic is utilized. This parameter is tested periodically by sampling production material.

[24] This specification indicates the guaranteed maximum frequency at which an individual output or buried register can be cycled in asynchronously clocked mode. This frequency is least of $1/(t_{AWH} + t_{AWL})$, $1/(t_{AS} + t_{AH})$, or $1/t_{ACO1}$. It also indicates the maximum frequency at which the device may operate in the asynchronously clocked data-path mode. Assumes no expander logic is used.

[25] This specification indicates the guaranteed maximum frequency at which an individual output or buried register can be cycled in asynchronously clocked mode by a clock signal applied to an external dedicated input or an I/O pin.

[26] This parameter indicates the minimum time that the previous register output data is maintained on the output pin after an asynchronous register clock input to an external dedicated input or I/O pin.

## Typical Internal Switching Characteristics Over Operating Range[7]

| | | | 7C344B–10 | | 7C344B–12 | | 7C344–15<br>7C344B–15 | | |
|---|---|---|---|---|---|---|---|---|---|
| Parameter | Description | | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | Com'l/Ind | | 2 | | 2.5 | | 4 | ns |
| | | Mil | | | | 2.5 | | 4 | |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | Com'l/Ind | | 2 | | 2.5 | | 4 | ns |
| | | Mil | | | | 2.5 | | 4 | |
| $t_{EXP}$ | Expander Array Delay | Com'l/Ind | | 6 | | 6 | | 8 | ns |
| | | Mil | | | | 6 | | 8 | |
| $t_{LAD}$ | Logic Array Data Delay | Com'l/Ind | | 5 | | 6 | | 7 | ns |
| | | Mil | | | | 6 | | 7 | |
| $t_{LAC}$ | Logic Array Control Delay | Com'l/Ind | | 5 | | 5 | | 5 | ns |
| | | Mil | | | | 5 | | 5 | |
| $t_{OD}$ | Output Buffer and Pad Delay | Com'l/Ind | | 3 | | 3 | | 4 | ns |
| | | Mil | | | | 3 | | 4 | |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | Com'l /Ind | | 5 | | 5 | | 7 | ns |
| | | Mil | | | | 5 | | 7 | |
| $t_{XZ}$ | Output Buffer Disable Delay | Com'l/Ind | | 5 | | 5 | | 7 | ns |
| | | Mil | | | | 5 | | 7 | |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | Com'l/Ind | 2 | | 2 | | 5 | | ns |
| | | Mil | | | 2 | | 5 | | |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | Com'l/Ind | 4 | | 5 | | 7 | | ns |
| | | Mil | | | 5 | | 7 | | |
| $t_{LATCH}$ | Flow-Through Latch Delay | Com'l/Ind | | 0.5 | | 0.5 | | 1 | ns |
| | | Mil | | | | 0.5 | | 1 | |
| $t_{RD}$ | Register Delay | Com'l/Ind | | 0.5 | | 0.5 | | 1 | ns |
| | | Mil | | | | 0.5 | | 1 | |
| $t_{COMB}$ | Transparent Mode Delay[28] | Com'l/Ind | | 0.5 | | 0.5 | | 1 | ns |
| | | Mil | | | | 0.5 | | 1 | |
| $t_{CH}$ | Clock HIGH Time | Com'l/Ind | 3 | | 4 | | 6 | | ns |
| | | Mil | | | 4 | | 6 | | |
| $t_{CL}$ | Clock LOW Time | Com'l/Ind | 3 | | 4 | | 6 | | ns |
| | | Mil | | | 4 | | 6 | | |
| $t_{IC}$ | Asynchronous Clock Logic Delay | Com'l/Ind | | 5 | | 6 | | 7 | ns |
| | | Mil | | | | 6 | | 7 | |
| $t_{ICS}$ | Synchronous Clock Delay | Com'l/Ind | | 0.5 | | 0.5 | | 1 | ns |
| | | Mil | | | | 0.5 | | 1 | |
| $t_{FD}$ | Feedback Delay | Com'l/Ind | | 1 | | 1 | | 1 | ns |
| | | Mil | | | | 1 | | 1 | |
| $t_{PRE}$ | Asynchronous Register Preset Time | Com'l/Ind | | 2 | | 3 | | 5 | ns |
| | | Mil | | | | 3 | | 5 | |
| $t_{CLR}$ | Asynchronous Register Clear Time | Com'l/Ind | | 2 | | 3 | | 5 | ns |
| | | Mil | | | | 3 | | 5 | |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | Com'l/Ind | 2 | | 3 | | 5 | | ns |
| | | Mil | | | 3 | | 5 | | |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | Com'l/Ind | 2 | | 3 | | 5 | | ns |
| | | Mil | | | 3 | | 5 | | |

Shaded area contains preliminary information.

**Notes:**

[27] Sample tested only for an output change of 500 mV.

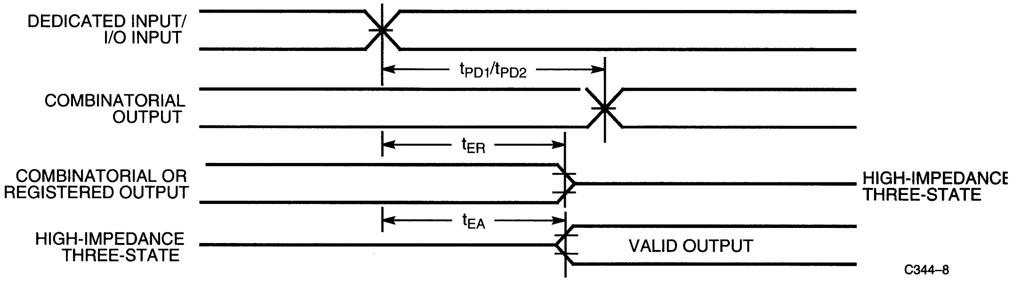
[28] This specification guarantees the maximum combinatorial delay associated with the macrocell register bypass when the macrocell is configured for combinatorial operation.

## Typical Internal Switching Characteristics Over Operating Range[7] (continued)

| Parameter | Description | | 7C344–20<br>7C344B–20 | | 7C344–25<br>7C344B–25 | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | Com'l/Ind | | 5 | | 7 | ns |
| | | Mil | | 5 | | 7 | |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | Com'l/Ind | | 5 | | 7 | ns |
| | | Mil | | 5 | | 7 | |
| $t_{EXP}$ | Expander Array Delay | Com'l/Ind | | 10 | | 15 | ns |
| | | Mil | | 10 | | 15 | |
| $t_{LAD}$ | Logic Array Data Delay | Com'l/Ind | | 9 | | 10 | ns |
| | | Mil | | 9 | | 10 | |
| $t_{LAC}$ | Logic Array Control Delay | Com'l/Ind | | 7 | | 7 | ns |
| | | Mil | | 7 | | 7 | |
| $t_{OD}$ | Output Buffer and Pad Delay | Com'l/Ind | | 5 | | 5 | ns |
| | | Mil | | 5 | | 5 | |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | Com'l /Ind | | 8 | | 11 | ns |
| | | Mil | | 8 | | 11 | |
| $t_{XZ}$ | Output Buffer Disable Delay | Com'l/Ind | | 8 | | 11 | ns |
| | | Mil | | 8 | | 11 | |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | Com'l/Ind | 5 | | 8 | | ns |
| | | Mil | 5 | | 8 | | |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | Com'l/Ind | 9 | | 12 | | ns |
| | | Mil | 9 | | 12 | | |
| $t_{LATCH}$ | Flow-Through Latch Delay | Com'l/Ind | | 1 | | 3 | ns |
| | | Mil | | 1 | | 3 | |
| $t_{RD}$ | Register Delay | Com'l/Ind | | 1 | | 1 | ns |
| | | Mil | | 1 | | 1 | |
| $t_{COMB}$ | Transparent Mode Delay[28] | Com'l/Ind | | 1 | | 3 | ns |
| | | Mil | | 1 | | 3 | |
| $t_{CH}$ | Clock HIGH Time | Com'l/Ind | 7 | | 8 | | ns |
| | | Mil | 7 | | 8 | | |
| $t_{CL}$ | Clock LOW Time | Com'l/Ind | 7 | | 8 | | ns |
| | | Mil | 7 | | 8 | | |
| $t_{IC}$ | Asynchronous Clock Logic Delay | Com'l/Ind | | 8 | | 10 | ns |
| | | Mil | | 8 | | 10 | |
| $t_{ICS}$ | Synchronous Clock Delay | Com'l/Ind | | 2 | | 3 | ns |
| | | Mil | | 2 | | 3 | |
| $t_{FD}$ | Feedback Delay | Com'l/Ind | | 1 | | 1 | ns |
| | | Mil | | 1 | | 1 | |
| $t_{PRE}$ | Asynchronous Register Preset Time | Com'l/Ind | | 6 | | 9 | ns |
| | | Mil | | 6 | | 9 | |
| $t_{CLR}$ | Asynchronous Register Clear Time | Com'l/Ind | | 6 | | 9 | ns |
| | | Mil | | 6 | | 9 | |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | Com'l/Ind | 5 | | 7 | | ns |
| | | Mil | 5 | | 7 | | |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | Com'l/Ind | 5 | | 7 | | ns |
| | | Mil | 5 | | 7 | | |

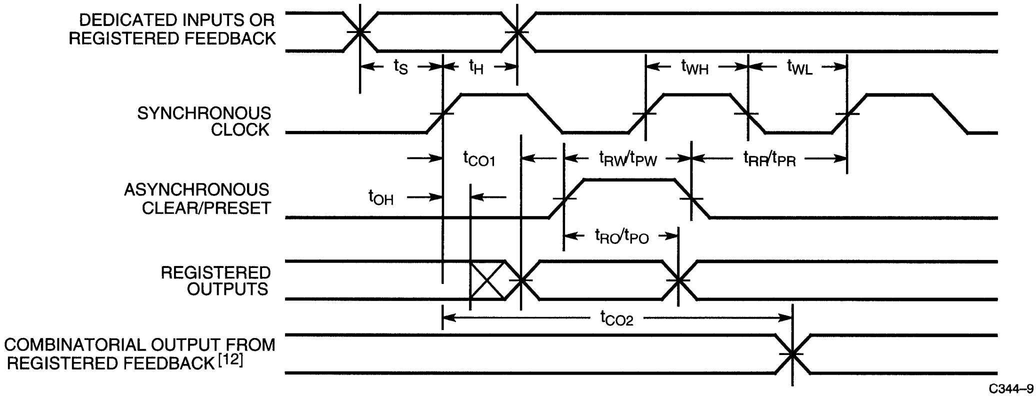
## Switching Waveforms

### External Combinatorial



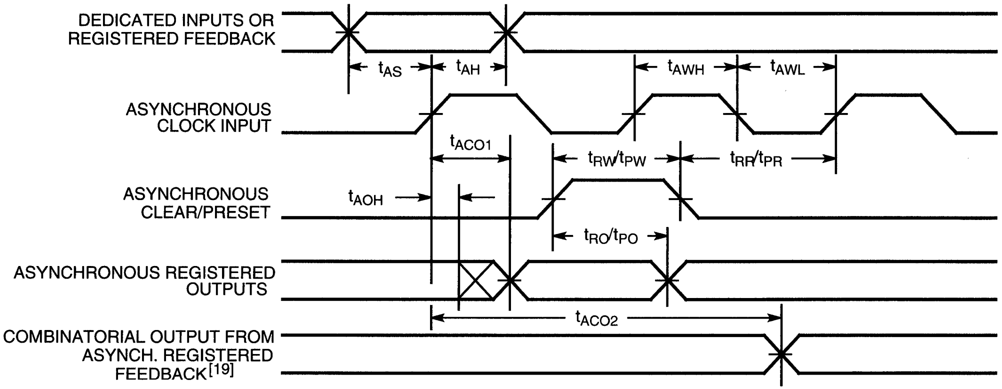
C344–8

### External Synchronous



C344–9

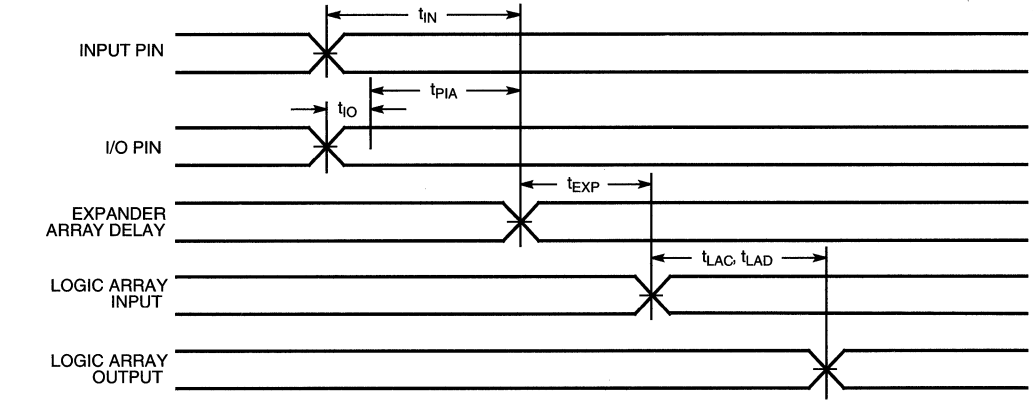
### External Asynchronous



C344–10

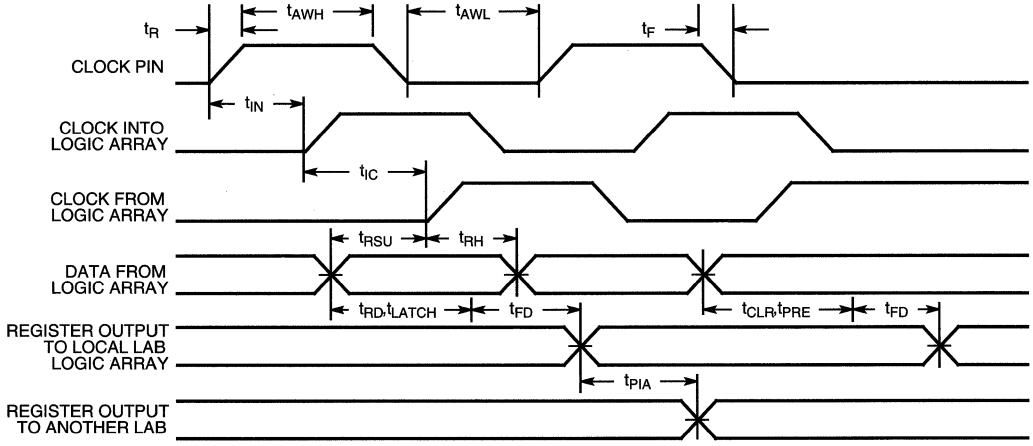
## Switching Waveforms (Continued)

### Internal Combinatorial



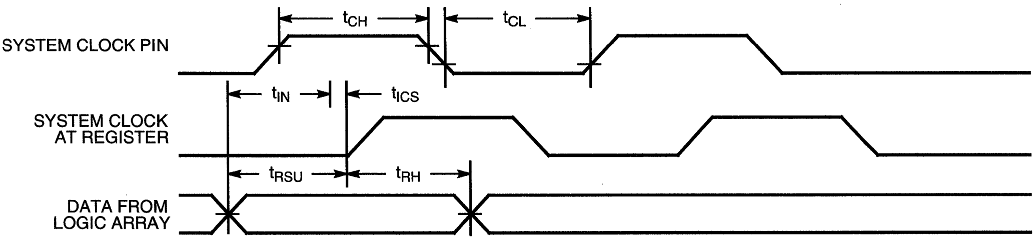
C344–11

### Internal Asynchronous



C344–12

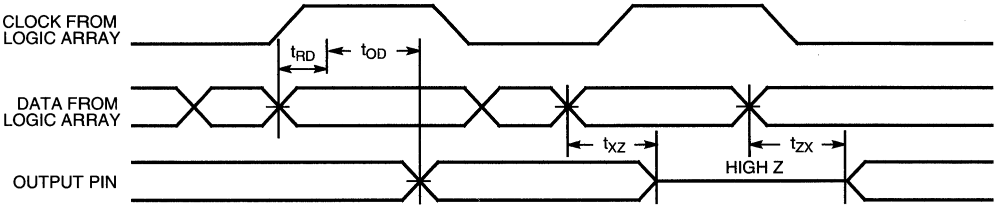
### Internal Synchronous (Input Path)



C344–13

## Switching Waveforms (Continued)

### Internal Synchronous (Output Path)



C344–14

## Ordering Information

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 10 | CY7C344B–10HC | H64 | 28-Lead Windowed Leaded Chip Carrier | Commercial |
| | CY7C344B–10JC | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344B–10PC | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344B–10WC | W22 | 28-Lead Windowed CerDIP | |
| 12 | CY7C344B–12HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C344B–12JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344B–12PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344B–12WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–12HMB | H64 | 28-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C344B–12WMB | W22 | 28-Lead Windowed CerDIP | |
| 15 | CY7C344–15HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C344–15JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344–15PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344–15WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–15HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | |
| | CY7C344B–15JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344B–15PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344B–15WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–15HMB | H64 | 28-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C344B–15WMB | W22 | 28-Lead Windowed CerDIP | |
| 20 | CY7C344–20HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C344–20JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344–20PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344–20WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–20HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | |
| | CY7C344B–20JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344B–20PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344B–20WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344–20HMB | H64 | 28-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C344–20WMB | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–20HMB | H64 | 28-Lead Windowed Leaded Chip Carrier | |
| | CY7C344B–20WMB | W22 | 28-Lead Windowed CerDIP | |

Shaded area contains preliminary information.

## Ordering Information (continued)

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 25 | CY7C344–25HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C344–25JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344–25PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344–25WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–25HC/HI | H64 | 28-Lead Windowed Leaded Chip Carrier | |
| | CY7C344B–25JC/JI | J64 | 28-Lead Plastic Leaded Chip Carrier | |
| | CY7C344B–25PC/PI | P21 | 28-Lead (300-Mil) Molded DIP | |
| | CY7C344B–25WC/WI | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344–25HMB | H64 | 28-Lead Windowed Leaded Chip Carrier | Military |
| | CY7C344–25WMB | W22 | 28-Lead Windowed CerDIP | |
| | CY7C344B–25HMB | H64 | 28-Lead Windowed Leaded Chip Carrier | |
| | CY7C344B–25WMB | W22 | 28-Lead Windowed CerDIP | |

Shaded area contains preliminary information.

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD1}$ | 7, 8, 9, 10, 11 |
| $t_{PD2}$ | 7, 8, 9, 10, 11 |
| $t_{PD3}$ | 7, 8, 9, 10, 11 |
| $t_{CO1}$ | 7, 8, 9, 10, 11 |
| $t_S$ | 7, 8, 9, 10, 11 |
| $t_H$ | 7, 8, 9, 10, 11 |
| $t_{ACO1}$ | 7, 8, 9, 10, 11 |
| $t_{ACO1}$ | 7, 8, 9, 10, 11 |
| $t_{AS}$ | 7, 8, 9, 10, 11 |
| $t_{AH}$ | 7, 8, 9, 10, 11 |

Document #: 38–00127–G

MAX is a registered trademark of Altera Corporation.
*Warp2*, and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.
*Warp2Sim* is a trademark of Cypress Semiconductor Corporation.

CYPRESS

# CY7C346
# CY7C346B

# 128-Macrocell MAX® EPLDs

## Features

- 128 macrocells in 8 LABs
- 20 dedicated inputs, up to 64 bidirectional I/O pins
- Programmable interconnect array
- 0.8-micron double-metal CMOS EPROM technology (CY7C346)
- Advanced 0.65-micron CMOS technology to increase performance (CY7C346B)
- Available in 84-pin CLCC, PLCC, and 100-pin PGA, PQFP

## Functional Description

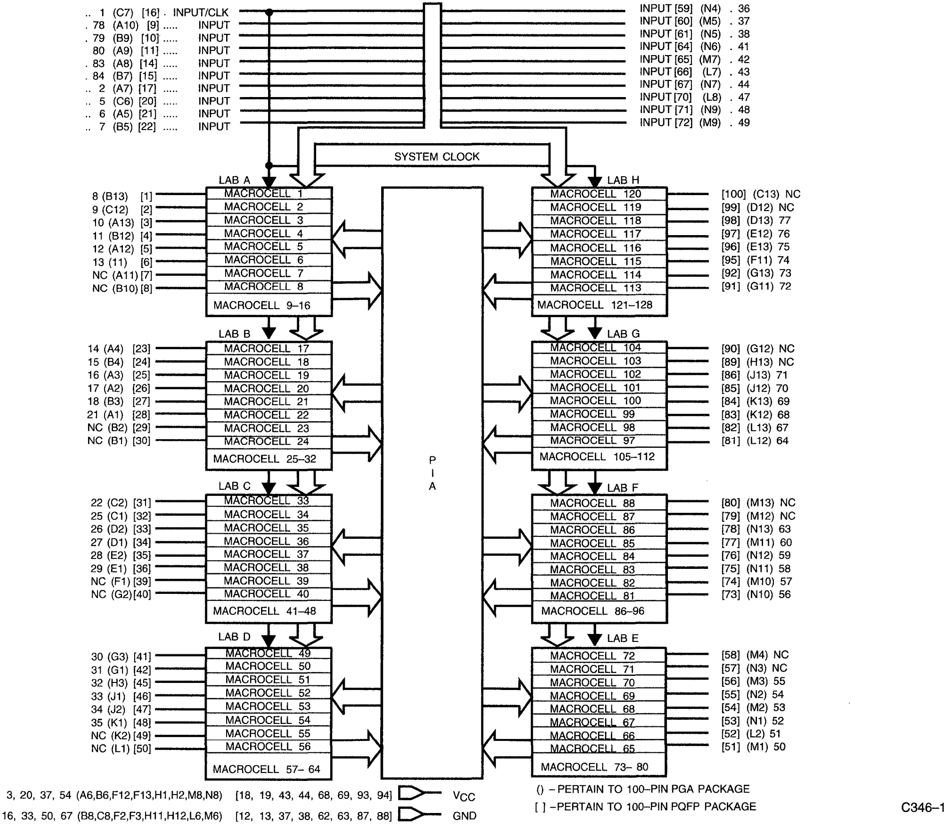
The CY7C346/CY7C346B is an Erasable Programmable Logic Device (EPLD) in which CMOS EPROM cells are used to configure logic functions within the device. The MAX architecture is 100% user configurable, allowing the devices to accommodate a variety of independent logic functions.

The 128 macrocells in the CY7C346/CY7C346B are divided into 8 Logic Array Blocks (LABs), 16 per LAB. There are 256 expander product terms, 32 per LAB, to be used and shared by the macrocells within each LAB.

Each LAB is interconnected through the programmable interconnect array, allowing all signals to be routed throughout the chip.

The speed and density of the CY7C346/CY7C346B allow it to be used in a wide range of applications, from replacement of large amounts of 7400-series TTL logic, to complex controllers and multifunction chips. With greater than 25 times the functionality of 20-pin PLDs, the CY7C346/CY7C346B allows the replacement of over 50 TTL devices. By replacing large amounts of logic, the CY7C346/CY7C346B reduces board space, part count, and increases system reliability.

## Logic Block Diagram



C346–1

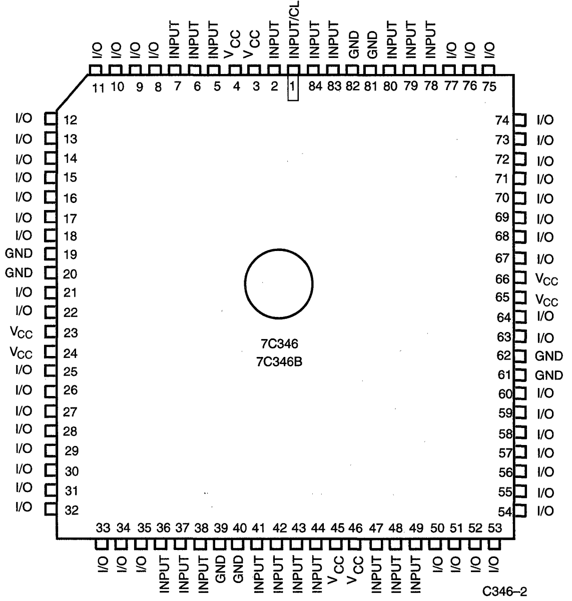
## Selection Guide

| | | 7C346B–15 | 7C346B–20 | 7C346–25<br>7C346B–25 | 7C346–30<br>7C346B–30 | 7C346–35<br>7C346B–35 |
|---|---|---|---|---|---|---|
| Maximum Access Time (ns) | | 15 | 20 | 25 | 30 | 35 |
| Maximum Operating Current (mA) | Commercial | 250 | 250 | 250 | 250 | 250 |
| | Military | | 320 | 325 | 320 | 320 |
| | Industrial | 320 | 320 | 320 | 320 | 320 |
| Maximum Standby Current (mA) | Commercial | 225 | 225 | 225 | 225 | 225 |
| | Military | | 275 | 275 | 275 | 275 |
| | Industrial | 275 | 275 | 275 | 275 | 275 |

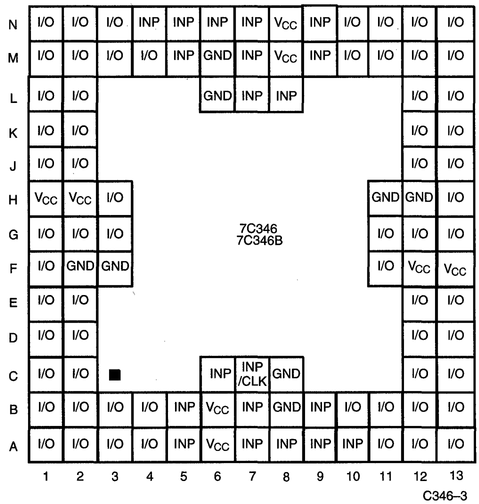
Shaded area contains preliminary information.

## Pin Configurations

**PLCC/CLCC**
**Top View**



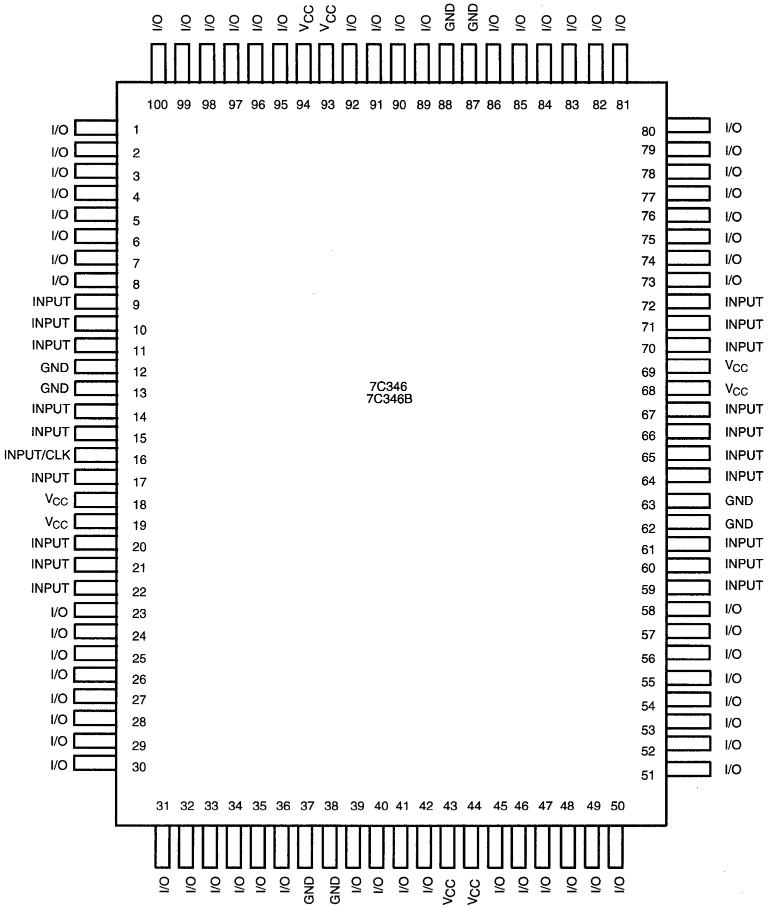
**PGA**
**BottomView**



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| N | I/O | I/O | I/O | INP | INP | INP | INP | $V_{CC}$ | INP | I/O | I/O | I/O | I/O |
| M | I/O | I/O | I/O | I/O | INP | GND | INP | $V_{CC}$ | INP | I/O | I/O | I/O | I/O |
| L | I/O | I/O | | | | GND | INP | INP | | | | I/O | I/O |
| K | I/O | I/O | | | | | | | | | | I/O | I/O |
| J | I/O | I/O | | | | | | | | | | I/O | I/O |
| H | $V_{CC}$ | $V_{CC}$ | I/O | | | | | | | | GND | GND | I/O |
| G | I/O | I/O | I/O | | | | | | | | I/O | I/O | I/O |
| F | I/O | GND | GND | | | | | | | | I/O | $V_{CC}$ | $V_{CC}$ |
| E | I/O | I/O | | | | | | | | | | I/O | I/O |
| D | I/O | I/O | | | | | | | | | | I/O | I/O |
| C | I/O | I/O | ■ | | | INP | INP<br>/CLK | GND | | | | I/O | I/O |
| B | I/O | I/O | I/O | I/O | INP | $V_{CC}$ | INP | GND | INP | I/O | I/O | I/O | I/O |
| A | I/O | I/O | I/O | I/O | INP | $V_{CC}$ | INP | INP | INP | INP | I/O | I/O | I/O |

## Pin Configurations (continued)

**PQFP**
**Top View**



C346–4

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Storage Temperature ....................................–65°C to+150°C

Ambient Temperature with
Power Applied .............................................–55°C to+125°C

Maximum Junction Temperature
(under bias)................................................................ 150°C

Supply Voltage to Ground Potential ................–2.0V to+7.0V

Maximum Power Dissipation..................................2500 mW

DC $V_{CC}$ or GND Current...........................................500 mA

DC Output Current per Pin........................–25 mA to+25 mA

DC Input Voltage[1] ...................................... –3.0V to + 7.0V

DC Program Voltage..................................................... 13.0V

Static Discharge Voltage............................................>1100V
(per MIL-STD-883, Method 3015)

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0°C to +70°C | 5V ± 5% |
| Industrial | –40°C to +85°C | 5V ± 10% |
| Military | –55°C to +125°C (Case) | 5V ± 10% |

## Electrical Characteristics Over the Operating Range[2]

| Parameter | Description | Test Conditions | | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $I_{OH}$ = –4.0 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $I_{OL}$ = 8.0 mA | | | 0.45 | V |
| $V_{IH}$ | Input HIGH Voltage | | | 2.2 | $V_{CC}$ +0.3 | V |
| $V_{IL}$ | Input LOW Voltage | | | –0.3 | 0.8 | V |
| $I_{IX}$ | Input Current | GND ≤ $V_{IN}$ ≤ $V_{CC}$ | | –10 | +10 | µA |
| $I_{OZ}$ | Output Leakage Current | $V_O$ = $V_{CC}$ or GND | | –40 | +40 | µA |
| $I_{OS}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5V[3, 4] | | –30 | –90 | mA |
| $I_{CC1}$ | Power Supply Current (Standby) | $V_I$ = GND (No Load) | Com'l | | 225 | mA |
| | | | Mil/Ind | | 275 | |
| $I_{CC2}$ | Power Supply Current[5] | $V_I$ = $V_{CC}$ or GND (No Load)<br>f = 1.0 MHz[4] | Com'l | | 250 | mA |
| | | | Mil/Ind | | 320 | |
| $t_R$ | Recommended Input Rise Time | | | | 100 | ns |
| $t_F$ | Recommended Input Fall Time | | | | 100 | ns |

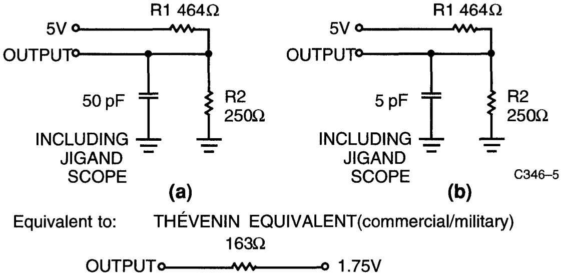
## Capacitance[6]

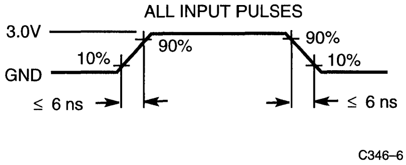
| Parameter | Description | Test Conditions | Max. | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2V, f = 1.0 MHz | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2V, f = 1.0 MHz | 20 | pF |

**Notes:**

1. Minimum DC input is –0.3V. During transitions, the inputs may undershoot to –3.0V for periods less than 20 ns.
2. Typical values are for $T_A$ = 25°C and $V_{CC}$ = 5V.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
4. Guaranteed by design but not 100% tested.
5. This parameter is measured with device programmed as a 16-bit counter in each LAB.
6. Part (a) in AC Test Load and Waveforms is used for all parameters except $t_{ER}$ and $t_{XZ}$, which is used for part (b) in AC Test Load and Waveforms. All external timing parameters are measured referenced to external pins of the device.

## AC Test Loads and Waveforms[6]





## Logic Array Blocks

There are 8 logic array blocks in the CY7C346/CY7C346B. Each LAB consists of a macrocell array containing 16 macrocells, an expander product term array containing 32 expanders, and an I/O block. The LAB is fed by the programmable interconnect array and the dedicated input bus. All macrocell feedbacks go to the macrocell array, the expander array, and the programmable interconnect array. Expanders feed themselves and the macrocell array. All I/O feedbacks go to the programmable interconnect array so that they may be accessed by macrocells in other LABs as well as the macrocells in the LAB in which they are situated.

Externally, the CY7C346/CY7C346B provides 20 dedicated inputs, one of which may be used as a system clock. There are 64 I/O pins that may be individually configured for input, output, or bidirectional data flow.

## Programmable Interconnect Array

The Programmable Interconnect Array (PIA) solves interconnect limitations by routing only the signals needed by each logic array block. The inputs to the PIA are the outputs of every macrocell within the device and the I/O pin feedback of every pin on the device.

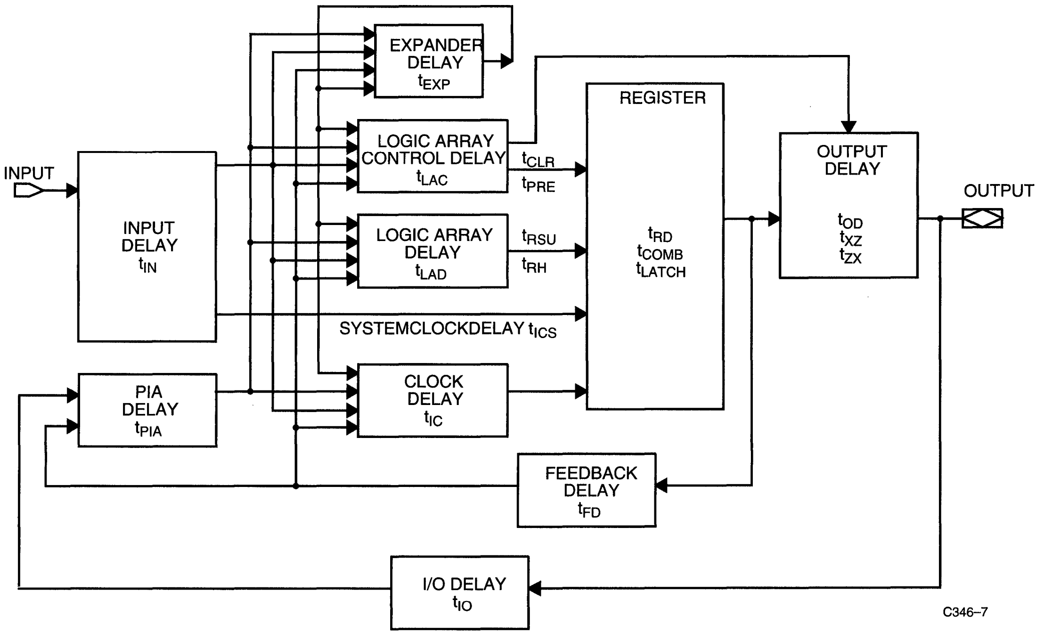


**Figure 1. CY7C346/CY7C346B Internal Timing Model**

## Timing Delays

Timing delays within the CY7C346/CY7C346B may be easily determined using *Warp2*® or *Warp3*® software or by the model shown in or *Figure 1*. The CY7C346 /CY7C346B has fixed internal delays, allowing the user to determine the worst case timing delays for any design. For complete timing information, *Warp3* software provides a timing simulator.

## Design Recommendations

Operation of the devices described herein with conditions above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this datasheet is not implied. Exposure to absolute maximum ratings conditions for extended periods of time may affect device reliability. The CY7C346/CY7C346B contains circuitry to protect device pins from high static voltages or electric fields, but normal precautions should be taken to avoid application of any voltage higher than the maximum rated voltages.

For proper operation, input and output pins must be constrained to the range GND ≤ (VIN or VOUT) ≤ VCC. Unused inputs must always be tied to an appropriate logic level (either VCC or GND). Each set of VCC and GND pins must be connected together directly at the device. Power supply decoupling capacitors of at least 0.2 µF must be connected between VCC and GND. For the most effective decoupling, each VCC pin should be separately decoupled to GND directly at the device. Decoupling capacitors should have good frequency response, such as monolithic ceramic types have.

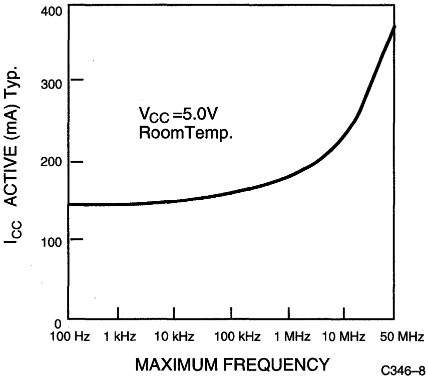
## Design Security

The CY7C346/CY7C346B contains a programmable design security feature that controls the access to the data programmed into the device. If this programmable feature is used, a proprietary design implemented in the device cannot be copied or retrieved. This enables a high level of design control to be obtained since programmed data within EPROM cells is invisible. The bit that controls this function, along with all other program data, may be reset simply by erasing the entire device.

The CY7C346/CY7C346B is fully functionally tested and guaranteed through complete testing of each programmable EPROM bit and all internal logic elements thus ensuring 100% programming yield.

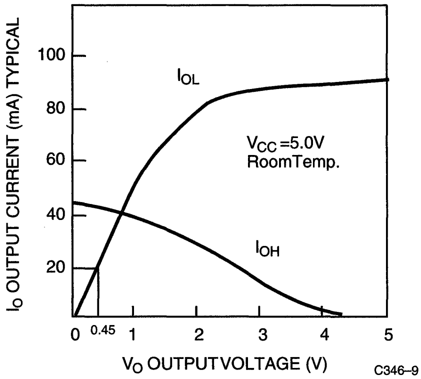
The erasable nature of these devices allows test programs to be used and erased during early stages of the production flow. The devices also contain on-board logic test circuitry to allow verification of function and AC specification once encapsulated in non-windowed packages.

## Typical $I_{CC}$ vs. $f_{MAX}$



C346–8

## Output Drive Current



C346–9

## Timing Considerations

Unless otherwise stated, propagation delays do not include expanders. When using expanders, add the maximum expander delay $t_{EXP}$ to the overall delay. Similarly, there is an additional $t_{PIA}$ delay for an input from an I/O pin when compared to a signal from straight input pin.

When calculating synchronous frequencies, use $t_{S1}$ if all inputs are on dedicated input pins. The parameter $t_{S2}$ should be used if data is applied at an I/O pin. If $t_{S2}$ is greater than $t_{CO1}$, $1/t_{S2}$ becomes the limiting frequency in the data path mode unless $1/(t_{WH} + t_{WL})$ is less than $1/t_{S2}$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{S1}$. Determine which of $1/(t_{WH} + t_{WL})$, $1/t_{CO1}$, or $1/(t_{EXP} + t_{S1})$ is the lowest frequency. The lowest of these frequencies is the maximum data path frequency for the synchronous configuration.

When calculating external asynchronous frequencies, use $t_{AS1}$ if all inputs are on the dedicated input pins. If any data is applied to an I/O pin, $t_{AS2}$ must be used as the required set-up time. If $(t_{AS2} + t_{AH})$ is greater than $t_{ACO1}$, $1/(t_{AS2} + t_{AH})$ becomes the limiting frequency in the data path mode unless $1/(t_{AWH} + t_{AWL})$ is less than $1/(t_{AS2} + t_{AH})$.

When expander logic is used in the data path, add the appropriate maximum expander delay, $t_{EXP}$ to $t_{AS1}$. Determine which of $1/(t_{AWH} + t_{AWL})$, $1/t_{ACO1}$, or $1/(t_{EXP} + t_{AS1})$ is the lowest frequency. The lowest of these frequencies is the maximum data path frequency for the asynchronous configuration.

The parameter $t_{OH}$ indicates the system compatibility of this device when driving other synchronous logic with positive input hold times, which is controlled by the same synchronous clock. If $t_{OH}$ is greater than the minimum required input hold time of the subsequent synchronous logic, then the devices are guaranteed to function properly with a common synchronous clock under worst-case environmental and supply voltage conditions.

The parameter $t_{AOH}$ indicates the system compatibility of this device when driving subsequent registered logic with a positive hold time and using the same asynchronous clock as the CY7C346/CY7C346B.

In general, if $t_{AOH}$ is greater than the minimum required input hold time of the subsequent logic (synchronous or asynchronous) then the devices are guaranteed to function properly under worst-case environmental and supply voltage conditions, provided the clock signal source is the same. This also applies if expander logic is used in the clock signal path of the driving device, but not for the driven device. This is due to the expander logic in the second device's clock signal path adding an additional delay ($t_{EXP}$) causing the output data from the preceding device to change prior to the arrival of the clock signal at the following device's register.

## Commercial and Industrial External Synchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C346B–15 | | 7C346B–20 | | 7C346–25<br>7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | | 25 | | 32 | | 40 | | 45 | | 55 | ns |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | | 23 | | 30 | | 37 | | 44 | | 55 | ns |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4, 10] | | 33 | | 42 | | 52 | | 59 | | 75 | ns |
| $t_{EA}$ | Input to Output Enable Delay[4, 7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{ER}$ | Input to Output Disable Delay[4, 7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | | 7 | | 8 | | 14 | | 16 | | 20 | ns |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4, 11] | | 17 | | 20 | | 30 | | 35 | | 42 | ns |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[7, 12] | 10 | | 13 | | 15 | | 20 | | 25 | | ns |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7] | 20 | | 24 | | 30 | | 36 | | 45 | | ns |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | 0 | | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | 5 | | 7 | | 8 | | 10 | | 12.5 | | ns |
| $t_{WL}$ | Synchronous Clock Input LOW Time | 5 | | 7 | | 8 | | 10 | | 12.5 | | ns |
| $t_{RW}$ | Asynchronous Clear Width[4, 7] | 16 | | 22 | | 25 | | 30 | | 35 | | ns |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4,7] | 16 | | 22 | | 25 | | 30 | | 35 | | ns |
| $t_{RO}$ | Asynchronous Clear to Regis- tered Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{PW}$ | Asynchronous Preset Width[4, 7] | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4,7] | 15 | | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{PO}$ | Asynchronous Preset to Regis- tered Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4, 13] | | 3 | | 3 | | 3 | | 3 | | 6 | ns |
| $t_P$ | External Synchronous Clock Period $(1/(f_{MAX3}))$[4] | 12 | | 15 | | 16 | | 20 | | 25 | | ns |
| $f_{MAX1}$ | External Feedback Maximum Frequency $(1/(t_{CO1} + t_{S1}))$[4, 14] | 58.8 | | 47.6 | | 34.5 | | 27.7 | | 22.2 | | MHz |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of $(1/(t_{S1} + t_{CF}))$ or $(1/t_{CO1})$[4, 15] | 76.9 | | 62.5 | | 55.5 | | 43.4 | | 32.2 | | MHz |

Shaded area contains preliminary information.

## Commercial and Industrial External Synchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C346B–15 | | 7C346B–20 | | 7C346–25<br>7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $f_{MAX3}$ | Data Path Maximum Frequency, lesser of $(1/(t_{WL} + t_{WH}))$, $(1/(t_{S1} + t_H))$ or $(1/t_{CO1})$[4, 16] | 100 | | 71.4 | | 62.5 | | 50 | | 40 | | MHz |
| $f_{MAX4}$ | Maximum Register Toggle Frequency $(1/(t_{WL} + t_{WH}))$[4, 17] | 100 | | 71.4 | | 62.5 | | 50 | | 40 | | MHz |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4, 18] | 3 | | 3 | | 3 | | 3 | | 3 | | ns |

Shaded area contains preliminary information.

**Notes:**

7. This specification is a measure of the delay from input signal applied to a dedicated input (68-pin PLCC input pin 1, 2, 32, 34, 35, 66, or 68) to combinatorial output on any output pin. This delay assumes no expander terms are used to form the logic function.
When this note is applied to any parameter specification it indicates that the signal (data, asynchronous clock, asynchronous clear, and/or asynchronous preset) is applied to a dedicated input only and no signal path (either clock or data) employs expander logic.
If an input signal is applied to an I/O pin an additional delay equal to $t_{PIA}$ should be added to the comparable delay for a dedicated input. If expanders are used, add the maximum expander delay $t_{EXP}$ to the overall delay for the comparable delay without expanders.
8. This specification is a measure of the delay from input signal applied to an I/O macrocell pin to any output. This delay assumes no expander terms are used to form the logic function.
9. This specification is a measure of the delay from an input signal applied to a dedicated input (68-pin PLCC input pin 1, 2, 32, 34, 35, 36, 66, or 68) to combinatorial output on any output pin. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic.
10. This specification is a measure of the delay from an input signal applied to an I/O macrocell pin to any output. This delay assumes expander terms are used to form the logic function and includes the worst-case expander logic delay for one pass through the expander logic. This parameter is tested periodically by sampling production material.
11. This specification is a measure of the delay from synchronous register clock to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used, register is synchronously clocked and all feedback is within the same LAB. This parameter is tested periodically by sampling production material.
12. If data is applied to an I/O input for capture by a macrocell register, the I/O pin input set-up time minimums should be observed. These parameters are $t_{S2}$ for synchronous operation and $t_{AS2}$ for asynchronous operation.
13. This specification is a measure of the delay associated with the internal register feedback path. This is the delay from synchronous clock to LAB logic array input. This delay plus the register set-up time, $t_{S1}$, is the minimum internal period for an internal synchronous state machine configuration. This delay is for feedback within the same LAB. This parameter is tested periodically by sampling production material.
14. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which a state machine configuration with external feedback can operate. It is assumed that all data inputs and external feedback signals are applied to dedicated inputs.
15. This specification indicates the guaranteed maximum frequency at which a state machine with internal-only feedback can operate. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{CO1}$. All feedback is assumed to be local originating within the same LAB.
16. This frequency indicates the maximum frequency at which the device may operate in data path mode (dedicated input pin to output pin). This assumes data input signals are applied to dedicated input pins and no expander logic is used. If any of the data inputs are I/O pins, $t_{S2}$ is the appropriate $t_S$ for calculation.
17. This specification indicates the guaranteed maximum frequency, in synchronous mode, at which an individual output or buried register can be cycled by a clock signal applied to the dedicated clock input pin.
18. This parameter indicates the minimum time after a synchronous register clock input that the previous register output data is maintained on the output pin.

## Commercial and Industrial External Asynchronous Switching Characteristics[6] Over Operating Range

| | | 7C346B–15 | | 7C346B–20 | | 7C346–25<br>7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | Description | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] | | 15 | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | | 25 | | 32 | | 39 | | 46 | | 55 | ns |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | 5 | | 5 | | 5 | | 6 | | 8 | | ns |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | 14.5 | | 17 | | 19 | | 22 | | 28 | | ns |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | 5 | | 6 | | 6 | | 8 | | 10 | | ns |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | 9 | | 10 | | 11 | | 14 | | 16 | | ns |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | 7 | | 8 | | 9 | | 11 | | 14 | | ns |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4,21] | | 11 | | 13 | | 15 | | 18 | | 22 | ns |
| $t_{AP}$ | External Asynchronous Clock Period $(1/(f_{MAXA4}))$[4] | 16 | | 18 | | 20 | | 25 | | 30 | | ns |
| $f_{MAXA1}$ | External Feedback Maximum Frequency in Asynchronous Mode $(1/(t_{ACO1} + t_{AS1}))$[4,22] | 50 | | 40 | | 33.3 | | 27.7 | | 23.2 | | MHz |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4,23] | 62.5 | | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode [4,24] | 66.6 | | 50 | | 40 | | 33.3 | | 28.5 | | MHz |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency $1/(t_{AWH} + t_{AWL})$[4,25] | 62.5 | | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4,26] | 12 | | 12 | | 15 | | 15 | | 15 | | ns |

Shaded area contains preliminary information.

**Notes:**

19. This specification is a measure of the delay from an asynchronous register clock input to internal feedback of the register output signal to the input of the LAB logic array and then to a combinatorial output. This delay assumes no expanders are used in the logic of combinatorial output or the asynchronous clock input. The clock signal is applied to the dedicated clock input pin and all feedback is within a single LAB. This parameter is tested periodically by sampling production material.
20. This parameter is measured with a positive-edge triggered clock at the register. For negative edge triggering, the $t_{AWH}$ and $t_{AWL}$ parameters must be swapped. If a given input is used to clock multiple registers with both positive and negative polarity, $t_{AWH}$ should be used for both $t_{AWH}$ and $t_{AWL}$.
21. This specification is a measure of the delay associated with the internal register feedback path for an asynchronous clock to LAB logic array input. This delay plus the asynchronous register set-up time, $t_{AS1}$, is the minimum internal period for an internal asynchronously clocked state machine configuration. This delay is for feedback within the same LAB, assumes no expander logic in the clock path, and assumes that the clock input signal is applied to a dedicated input pin. This parameter is tested periodically by sampling production material.
22. This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine configuration with external feedback can operate. It is assumed that all data inputs, clock inputs, and feedback signals are applied to dedicated inputs and that no expander logic is employed in the clock signal path or data path.
23. This specification indicates the guaranteed maximum frequency at which an asynchronously clocked state machine with internal-only feedback can operate. This parameter is determined by the lesser of $(1/(t_{ACF} + t_{AS1}))$ or $(1/(t_{AWH} + t_{AWL}))$. If register output states must also control external points, this frequency can still be observed as long as this frequency is less than $1/t_{ACO1}$.
    This specification assumes no expander logic is utilized, all data inputs and clock inputs are applied to dedicated inputs, and all state feedback is within a single LAB. This parameter is tested periodically by sampling production material.
24. This frequency is the maximum frequency at which the device may operate in the asynchronously clocked data path mode. This specification is determined by the lesser of $1/(t_{AWH} + t_{AWL})$, $1/(t_{AS1} + t_{AH})$ or $1/t_{ACO1}$. It assumes data and clock input signals are applied to dedicated input pins and no expander logic is used.
25. This specification indicates the guaranteed maximum frequency at which an individual output or buried register can be cycled in asynchronously clocked mode by a clock signal applied to an external dedicated input pin.
26. This parameter indicates the minimum time that the previous register output data is maintained on the output after an asynchronous register clock input applied to an external dedicated input pin.

## Commercial and Industrial Internal Switching Characteristics Over Operating Range

| | | 7C346B–15 | | 7C346B–20 | | 7C346–25<br>7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | Description | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | Unit |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | | 3 | | 4 | | 5 | | 7 | | 9 | ns |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | | 3 | | 4 | | 6 | | 6 | | 9 | ns |
| $t_{EXP}$ | Expander Array Delay | | 8 | | 10 | | 12 | | 14 | | 20 | ns |
| $t_{LAD}$ | Logic Array Data Delay | | 8 | | 10 | | 12 | | 14 | | 16 | ns |
| $t_{LAC}$ | Logic Array Control Delay | | 5 | | 7 | | 10 | | 12 | | 13 | ns |
| $t_{OD}$ | Output Buffer and Pad Delay | | 3 | | 3 | | 5 | | 5 | | 6 | ns |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | | 5 | | 5 | | 10 | | 11 | | 13 | ns |
| $t_{XZ}$ | Output Buffer Disable Delay | | 5 | | 5 | | 10 | | 11 | | 13 | ns |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | 4 | | 5 | | 6 | | 8 | | 10 | | ns |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | 4 | | 5 | | 6 | | 8 | | 10 | | ns |
| $t_{LATCH}$ | Flow Through Latch Delay | | 1 | | 2 | | 3 | | 4 | | 4 | ns |
| $t_{RD}$ | Register Delay | | 1 | | 1 | | 1 | | 2 | | 2 | ns |
| $t_{COMB}$ | Transparent Mode Delay[28] | | 1 | | 2 | | 3 | | 4 | | 4 | ns |
| $t_{CH}$ | Clock HIGH Time | 4 | | 6 | | 8 | | 10 | | 12.5 | | ns |
| $t_{CL}$ | Clock LOW Time | 4 | | 6 | | 8 | | 10 | | 12.5 | | ns |
| $t_{IC}$ | Asynchronous Clock Logic Delay | | 6 | | 8 | | 14 | | 16 | | 18 | ns |
| $t_{ICS}$ | Synchronous Clock Delay | | 0.5 | | 0.5 | | 1 | | 1 | | 1 | ns |
| $t_{FD}$ | Feedback Delay | | 1 | | 1 | | 1 | | 1 | | 2 | ns |
| $t_{PRE}$ | Asynchronous Register Preset Time | | 3 | | 3 | | 5 | | 6 | | 7 | ns |
| $t_{CLR}$ | Asynchronous Register Clear Time | | 3 | | 3 | | 5 | | 6 | | 7 | ns |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | 3 | | 4 | | 5 | | 6 | | 7 | | ns |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | 3 | | 4 | | 5 | | 6 | | 7 | | ns |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | | 10 | | 12 | | 14 | | 16 | | 20 | ns |

Shaded area contains preliminary information.

**Notes:**

[27] Sample tested only for an output change of 500 mV.

[28] This specification guarantees the maximum combinatorial delay associated with the macrocell register bypass when the macrocell is configured for combinatorial operation.

## Military External Synchronous Switching Characteristics[6] Over Operating Range

| Parameter | Description | 7C346B–20 | | 7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD1}$ | Dedicated Input to Combinatorial Output Delay[7] | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{PD2}$ | I/O Input to Combinatorial Output Delay[8] | | 32 | | 39 | | 45 | | 55 | ns |
| $t_{PD3}$ | Dedicated Input to Combinatorial Output Delay with Expander Delay[9] | | 30 | | 37 | | 44 | | 55 | ns |
| $t_{PD4}$ | I/O Input to Combinatorial Output Delay with Expander Delay[4,10] | | 42 | | 51 | | 59 | | 75 | ns |
| $t_{EA}$ | Input to Output Enable Delay[4,7] | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{ER}$ | Input to Output Disable Delay[4,7] | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{CO1}$ | Synchronous Clock Input to Output Delay | | 8 | | 14 | | 16 | | 20 | ns |
| $t_{CO2}$ | Synchronous Clock to Local Feedback to Combinatorial Output[4,11] | | 20 | | 30 | | 35 | | 42 | ns |
| $t_{S1}$ | Dedicated Input or Feedback Set-Up Time to Synchronous Clock Input[7,12] | 13 | | 15 | | 20 | | 25 | | ns |
| $t_{S2}$ | I/O Input Set-Up Time to Synchronous Clock Input[7] | 24 | | 29 | | 36 | | 45 | | ns |
| $t_H$ | Input Hold Time from Synchronous Clock Input[7] | 0 | | 0 | | 0 | | 0 | | ns |
| $t_{WH}$ | Synchronous Clock Input HIGH Time | 7 | | 8 | | 10 | | 12.5 | | ns |
| $t_{WL}$ | Synchronous Clock Input LOW Time | 7 | | 8 | | 10 | | 12.5 | | ns |
| $t_{RW}$ | Asynchronous Clear Width[4,7] | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{RR}$ | Asynchronous Clear Recovery Time[4,7] | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{RO}$ | Asynchronous Clear to Registered Output Delay[7] | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{PW}$ | Asynchronous Preset Width[4,7] | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{PR}$ | Asynchronous Preset Recovery Time[4,7] | 20 | | 25 | | 30 | | 35 | | ns |
| $t_{PO}$ | Asynchronous Preset to Registered Output Delay[7] | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{CF}$ | Synchronous Clock to Local Feedback Input[4,13] | | 3 | | 3 | | 3 | | 6 | ns |
| $t_P$ | External Synchronous Clock Period ($1/(f_{MAX3})$)[4] | 14 | | 16 | | 20 | | 25 | | ns |
| $f_{MAX1}$ | External Feedback Maximum Frequency ($1/(t_{CO1} + t_{S1})$)[4,14] | 47.6 | | 34.5 | | 27.7 | | 22.2 | | MHz |

Shaded area contains preliminary information.

## Military External Synchronous Switching Characteristics[6] Over Operating Range (continued)

| | | 7C346B–20 | | 7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $f_{MAX2}$ | Internal Local Feedback Maximum Frequency, lesser of (1/($t_{S1}$ + $t_{CF}$)) or (1/$t_{CO1}$)[4,15] | 62.5 | | 55.5 | | 43.4 | | 32.2 | | MHz |
| $f_{MAX3}$ | Data Path Maximum Frequency, lesser of (1/($t_{WL}$ + $t_{WH}$)), (1/($t_{S1}$ + $t_{H}$)) or (1/$t_{CO1}$)[4,16] | 71.4 | | 62.5 | | 50 | | 40 | | MHz |
| $f_{MAX4}$ | Maximum Register Toggle Frequency (1/($t_{WL}$ + $t_{WH}$))[4,17] | 71.4 | | 62.5 | | 50 | | 40 | | MHz |
| $t_{OH}$ | Output Data Stable Time from Synchronous Clock Input[4,18] | 3 | | 3 | | 3 | | 3 | | ns |

Shaded area contains preliminary information.

## Military External Asynchronous Switching Characteristics[6] Over Operating Range

| | | 7C346B–20 | | 7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Description** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** |
| $t_{ACO1}$ | Asynchronous Clock Input to Output Delay[7] | | 20 | | 25 | | 30 | | 35 | ns |
| $t_{ACO2}$ | Asynchronous Clock Input to Local Feedback to Combinatorial Output[19] | | 32 | | 39 | | 46 | | 55 | ns |
| $t_{AS1}$ | Dedicated Input or Feedback Set-Up Time to Asynchronous Clock Input[7] | 6 | | 5 | | 6 | | 8 | | ns |
| $t_{AS2}$ | I/O Input Set-Up Time to Asynchronous Clock Input[7] | 17 | | 19 | | 22 | | 28 | | ns |
| $t_{AH}$ | Input Hold Time from Asynchronous Clock Input[7] | 6 | | 6 | | 8 | | 10 | | ns |
| $t_{AWH}$ | Asynchronous Clock Input HIGH Time[7] | 10 | | 11 | | 14 | | 16 | | ns |
| $t_{AWL}$ | Asynchronous Clock Input LOW Time[7, 20] | 8 | | 9 | | 11 | | 14 | | ns |
| $t_{ACF}$ | Asynchronous Clock to Local Feedback Input[4,21] | | 13 | | 15 | | 18 | | 22 | ns |
| $t_{AP}$ | External Asynchronous Clock Period (1/($f_{MAXA4}$))[4] | 18 | | 20 | | 25 | | 30 | | ns |
| $f_{MAXA1}$ | External Feedback Maximum Frequency in Asynchronous Mode (1/($t_{ACO1}$ + $t_{AS1}$))[4,22] | 40 | | 33.3 | | 27.7 | | 23.2 | | MHz |
| $f_{MAXA2}$ | Maximum Internal Asynchronous Frequency[4,23] | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| $f_{MAXA3}$ | Data Path Maximum Frequency in Asynchronous Mode[4,24] | 50 | | 40 | | 33.3 | | 28.5 | | MHz |
| $f_{MAXA4}$ | Maximum Asynchronous Register Toggle Frequency 1/($t_{AWH}$ + $t_{AWL}$)[4,25] | 55.5 | | 50 | | 40 | | 33.3 | | MHz |
| $t_{AOH}$ | Output Data Stable Time from Asynchronous Clock Input[4,26] | 12 | | 15 | | 15 | | 15 | | ns |

Shaded area contains preliminary information.

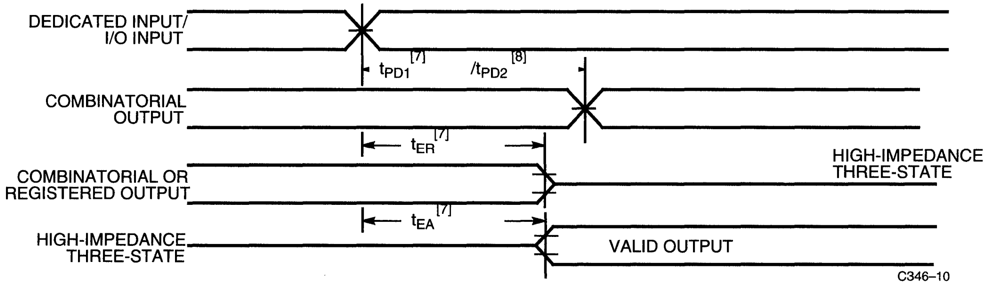
## Military Typical Internal Switching Characteristics Over Operating Range

| Parameter | Description | 7C346B–20 | | 7C346B–25 | | 7C346–30<br>7C346B–30 | | 7C346–35<br>7C346B–35 | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{IN}$ | Dedicated Input Pad and Buffer Delay | | 4 | | 5 | | 7 | | 9 | ns |
| $t_{IO}$ | I/O Input Pad and Buffer Delay | | 4 | | 6 | | 6 | | 9 | ns |
| $t_{EXP}$ | Expander Array Delay | | 10 | | 12 | | 14 | | 20 | ns |
| $t_{LAD}$ | Logic Array Data Delay | | 10 | | 12 | | 14 | | 16 | ns |
| $t_{LAC}$ | Logic Array Control Delay | | 7 | | 10 | | 12 | | 13 | ns |
| $t_{OD}$ | Output Buffer and Pad Delay | | 3 | | 5 | | 5 | | 6 | ns |
| $t_{ZX}$ | Output Buffer Enable Delay[27] | | 5 | | 10 | | 11 | | 13 | ns |
| $t_{XZ}$ | Output Buffer Disable Delay | | 5 | | 10 | | 11 | | 13 | ns |
| $t_{RSU}$ | Register Set-Up Time Relative to Clock Signal at Register | 5 | | 6 | | 8 | | 10 | | ns |
| $t_{RH}$ | Register Hold Time Relative to Clock Signal at Register | 5 | | 6 | | 8 | | 10 | | ns |
| $t_{LATCH}$ | Flow Through Latch Delay | | 2 | | 3 | | 4 | | 4 | ns |
| $t_{RD}$ | Register Delay | | 1 | | 1 | | 2 | | 2 | ns |
| $t_{COMB}$ | Transparent Mode Delay[28] | | 2 | | 3 | | 4 | | 4 | ns |
| $t_{CH}$ | Clock HIGH Time | 6 | | 8 | | 10 | | 12.5 | | ns |
| $t_{CL}$ | Clock LOW Time | 6 | | 8 | | 10 | | 12.5 | | ns |
| $t_{IC}$ | Asynchronous Clock Logic Delay | | 8 | | 14 | | 16 | | 18 | ns |
| $t_{ICS}$ | Synchronous Clock Delay | | 0.5 | | 2 | | 2 | | 3 | ns |
| $t_{FD}$ | Feedback Delay | | 1 | | 1 | | 1 | | 2 | ns |
| $t_{PRE}$ | Asynchronous Register Preset Time | | 3 | | 5 | | 6 | | 7 | ns |
| $t_{CLR}$ | Asynchronous Register Clear Time | | 3 | | 5 | | 6 | | 7 | ns |
| $t_{PCW}$ | Asynchronous Preset and Clear Pulse Width | 4 | | 5 | | 6 | | 7 | | ns |
| $t_{PCR}$ | Asynchronous Preset and Clear Recovery Time | 4 | | 5 | | 6 | | 7 | | ns |
| $t_{PIA}$ | Programmable Interconnect Array Delay Time | | 12 | | 14 | | 16 | | 20 | ns |

Shaded area contains preliminary information.

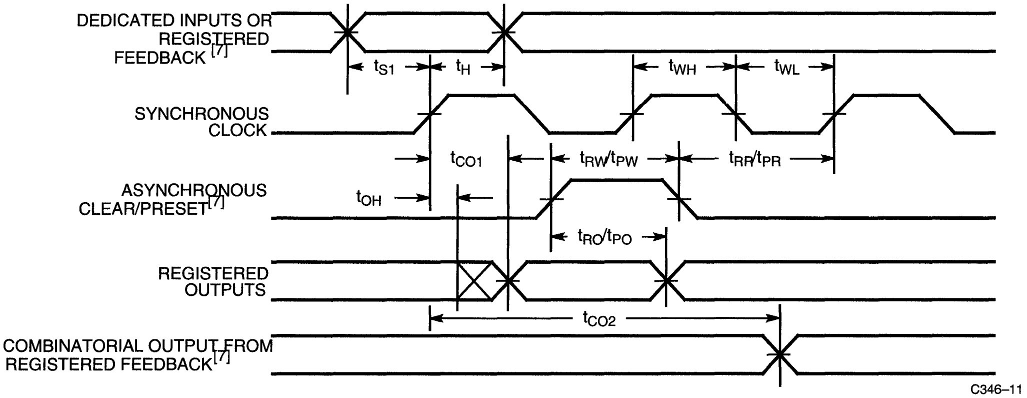
## Switching Waveforms

### External Combinatorial



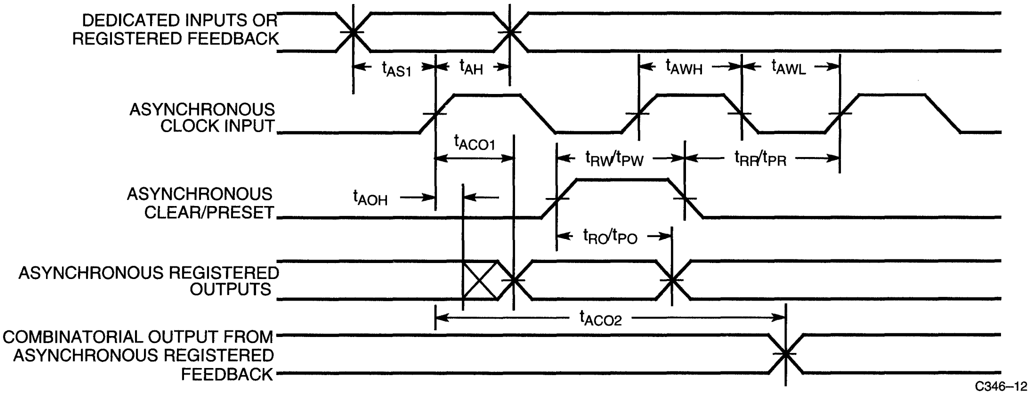
C346–10

### External Synchronous



C346–11

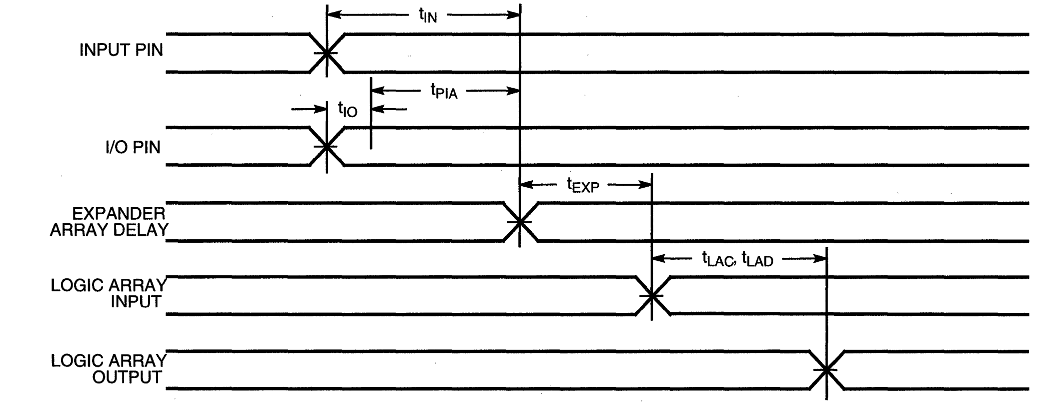
### External Asynchronous



C346–12

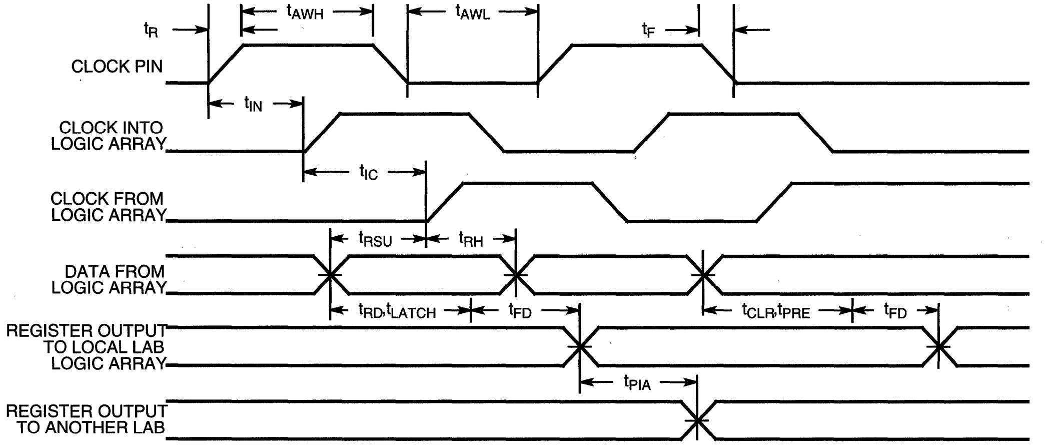
## Switching Waveforms (continued)

### Internal Combinatorial



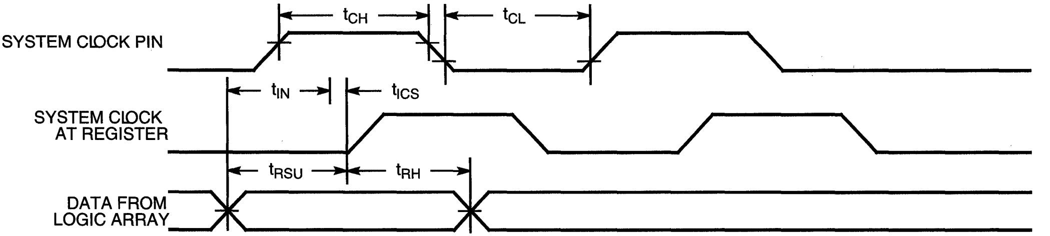
C346–13

### Internal Asynchronous



C346–14

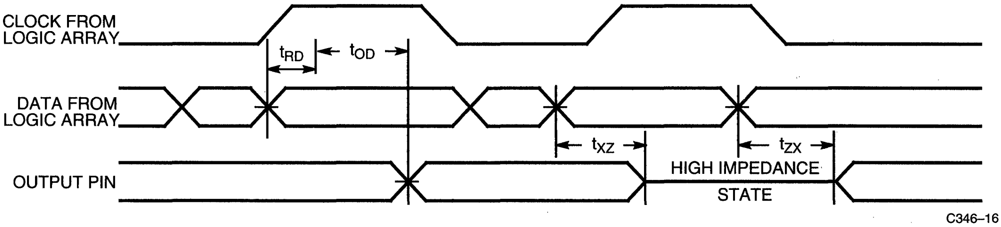
### Internal Synchronous



C346–15

## Switching Waveforms (continued)

### Internal Synchronous



## Ordering Information

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 15 | CY7C346B–15HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C346B–15JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346B–15NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346B–15RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| 20 | CY7C346B–20HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C346B–20JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346B–20NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346B–20RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346B–20HMB | H84 | 84-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C346B–20RMB | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| 25 | CY7C346–25HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C346–25JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346–25NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346–25RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346B–25HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | |
| | CY7C346B–25JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346B–25NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346B–25RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346B–25HMB | H84 | 84-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C346B–25RMB | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| 30 | CY7C346–30HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | Commercial/Industrial |
| | CY7C346–30JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346–30NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346B–30HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | |
| | CY7C346B–30JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346B–30NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346B–30RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346–30HMB | H84 | 84-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C346–30RMB | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346B–30HMB | H84 | 84-Pin Windowed Leaded Chip Carrier | |
| | CY7C346B–30RMB | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |

Shaded area contains preliminary information.

## Ordering Information (continued)

| Speed (ns) | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| 35 | CY7C346–35JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | Commercial/Industrial |
| | CY7C346–35NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346–35RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346B–35HC/HI | H84 | 84-Pin Windowed Leaded Chip Carrier | |
| | CY7C346B–35JC/JI | J83 | 84-Lead Plastic Leaded Chip Carrier | |
| | CY7C346B–35NC/NI | N100 | 100-Lead Plastic Quad Flatpack | |
| | CY7C346B–35RC/RI | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346–35HMB | H84 | 84-Pin Windowed Leaded Chip Carrier | Military |
| | CY7C346–35RMB | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |
| | CY7C346B–35HMB | H84 | 84-Pin Windowed Leaded Chip Carrier | |
| | CY7C346B–35RMB | R100 | 100-Pin Windowed Ceramic Pin Grid Array | |

Shaded area contains preliminary information.

## MILITARY SPECIFICATIONS
## Group A Subgroup Testing

### DC Characteristics

| Parameter | Subgroups |
|---|---|
| $V_{OH}$ | 1, 2, 3 |
| $V_{OL}$ | 1, 2, 3 |
| $V_{IH}$ | 1, 2, 3 |
| $V_{IL}$ | 1, 2, 3 |
| $I_{IX}$ | 1, 2, 3 |
| $I_{OZ}$ | 1, 2, 3 |
| $I_{CC1}$ | 1, 2, 3 |

### Switching Characteristics

| Parameter | Subgroups |
|---|---|
| $t_{PD1}$ | 7, 8, 9, 10, 11 |
| $t_{PD2}$ | 7, 8, 9, 10, 11 |
| $t_{PD3}$ | 7, 8, 9, 10, 11 |
| $t_{CO1}$ | 7, 8, 9, 10, 11 |
| $t_{S1}$ | 7, 8, 9, 10, 11 |
| $t_{S2}$ | 7, 8, 9, 10, 11 |
| $t_{H}$ | 7, 8, 9, 10, 11 |
| $t_{WH}$ | 7, 8, 9, 10, 11 |
| $t_{WL}$ | 7, 8, 9, 10, 11 |
| $t_{RO}$ | 7, 8, 9, 10, 11 |
| $t_{PO}$ | 7, 8, 9, 10, 11 |
| $t_{ACO1}$ | 7, 8, 9, 10, 11 |
| $t_{ACO2}$ | 7, 8, 9, 10, 11 |
| $t_{AS1}$ | 7, 8, 9, 10, 11 |
| $t_{AH}$ | 7, 8, 9, 10, 11 |
| $t_{AWH}$ | 7, 8, 9, 10, 11 |
| $t_{AWL}$ | 7, 8, 9, 10, 11 |

Document #: 38–00244–D

MAX is a registered trademark of Altera Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

CYPRESS

GENERAL INFORMATION 1

SMALL PLDS 2

CPLDS 3

**DEVELOPMENT TOOLS** 4

APPLICATION NOTES 5

QUALITY 6

PACKAGES 7

## Table of Contents

**Page Number**

## Development Tools

| Device | Description | Page Number |
|---|---|---|
| | PLD Development Tools Overview | 4-1 |
| CY3120 | *Warp2*® VHDL Compiler for PLDs | 4-2 |
| CY3122/CY3127 | *Warp2*Sim™ VHDL Development System for PLDs | 4-7 |
| CY3130/CY3135 | *Warp3*® VHDL Development System for PLDs | 4-12 |
| | Third-Party Tool Support | 4-18 |
| CY3140 | ABEL™/Synario™ Design Kit for FLASH370i™ | 4-26 |
| CY3144 | Cypress Mentor Graphics Bolt-in Kit | 4-27 |
| | PLD Programming Information | 4-28 |
| CY3500 | *Impulse3* ™ Device Programmer and Adapters | 4-30 |
| CY3600 | InSRkit™: ISR™ Programming Kit | 4-33 |
| CY3620 | *Warp2*ISR™ ISR Design Kit for CPLDs | 4-35 |

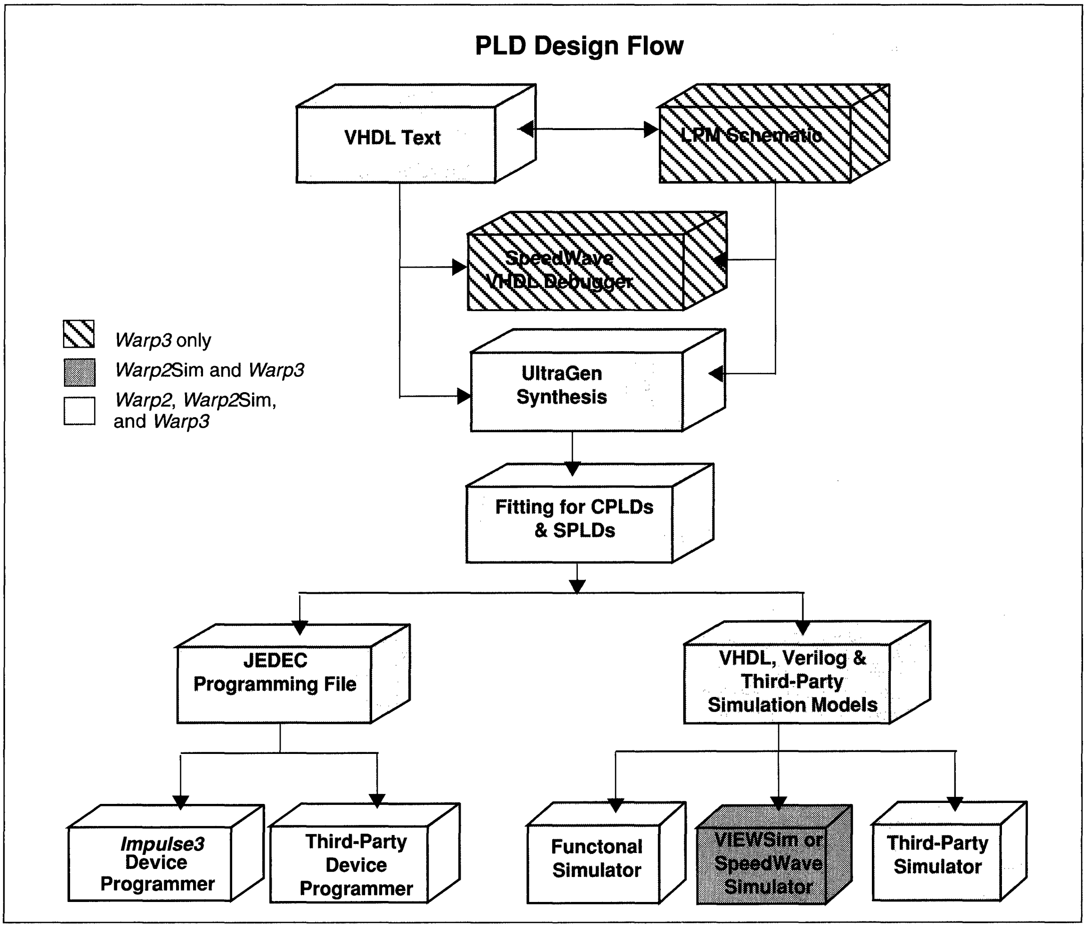
CYPRESS

# PLD Development Tools Overview

A large number of development tools are available for use when designing with Cypress Semiconductor's PLDs and CPLDs. Many of these tools are available from Cypress, while additional design flow options are available from numerous third-party tool vendors. (For a complete listing of third-party tool vendors, see the Third-Party Tools datasheet.)

Development software is available that provides design entry, synthesis, optimization, fitting, and simulation. As shown below, this software produces a programming file for use with a device programmer. Cypress offers the *Warp* software family of industry-leading VHDL synthesis tools. *Warp2*® provides VHDL design description and functional simulation. *Warp2*Sim™ has all of the features of *Warp2* and adds full timing simulation. *Warp3*® includes *Warp2*Sim functionality plus schematic entry and VHDL source code simulation. In addition, *Warp* seamlessly bolts-in to many third-party tools for various levels of support.

Device programmers use the programming file created by the development tool to program the PLD or CPLD. The *Impulse3*™ can program any Cypress Programmable Logic device and can be upgraded to program other manufacturers' devices. Many third-party programmers are available that can be used to program a wide array of devices including those from Cypress.


PLD Design Flow
VHDL Text
LPM Schematic
SpeedWave VHDL Debugger
▨ Warp3 only
▩ Warp2Sim and Warp3
□ Warp2, Warp2Sim, and Warp3
UltraGen Synthesis
Fitting for CPLDs & SPLDs
JEDEC Programming File
VHDL, Verilog & Third-Party Simulation Models
Impulse3 Device Programmer
Third-Party Device Programmer
Functonal Simulator
VIEWSim or SpeedWave Simulator
Third-Party Simulator

*Warp2*Sim and *Impulse3* are trademarks of Cypress Semiconductor Corporation.

*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

Document #: 38–00370–B

CYPRESS

# CY3120

# *Warp2*® VHDL Compiler for PLDs

## Features

- VHDL (IEEE 1076 and 1164) high-level language compiler
  - —Facilitates device independent design
  - —Designs are portable across multiple devices and/or CAE environments
  - —Facilitates the use of industry-standard simulation and synthesis tools for board and system-level design
  - —Supports functions and libraries facilitating modular design methodology
- *Warp2*® provides synthesis of IEEE standards 1076 and 1164 VHDL including:
  - —enumerated types
  - —operator overloading
  - —for ... generate statements
  - —integers
- Several design entry methods support high and low-level design descriptions:
  - —Behavioral VHDL (IF...THEN...ELSE; CASE...)
  - —Boolean
  - —Structural VHDL (RTL)
  - —Designs can include multiple VHDL entry methods in a single design
- State-of-the-art optimizations and reduction algorithms
  - —Automatic selection of optimal flip-flop type (D type/T type)
  - —Automatic pin assignment
- UltraGen™ Synthesis Technology
  - —Infers "modules" like adders, comparators, etc., from behavioral descriptions
  - —Replaces operator internally with an architecture specific circuit based on the target device
  - —User selectable speed and/or area optimization on a block-by-block basis
- Supports all Cypress Programmable Logic Devices
  - —Industry standard PLDs (16V8, 20V8, 22V10)
  - —MAX340 CPLDs
  - —FLASH370i CPLDs
- VHDL and Verilog timing model output for use with third-party simulators
- Functional simulation provided with Cypress NOVA simulator:
  - —Graphical waveform simulator
  - —Entry and modification of on-screen waveforms
  - —Ability to probe internal nodes
  - —Display of inputs, outputs, and High Z signals in different colors
  - —Automatic clock and pulse creation
  - —Waveform to JEDEC test vector conversion utility
  - —Support for buses
- PC, Sun (SunOS™, Sun Solaris™), and HP platforms
- Windows 3.1x, Windows 95, and Windows NT
- Motif on Unix workstations
- On-line documentation and help

## Functional Description

*Warp2* is a state-of-the-art VHDL compiler for designing with Cypress Programmable Logic Devices. *Warp2* utilizes a subset of IEEE 1076 and 1164 VHDL as its Hardware Description Language (HDL) for design entry. *Warp2* accepts VHDL input, synthesizes and optimizes the entered design, and outputs a JEDEC map for the desired PLD or CPLD (see *Figure 1*). For simulation, *Warp2* provides a graphical waveform simulator called NOVA, as well as VHDL and Verilog models for use with third party simulators.

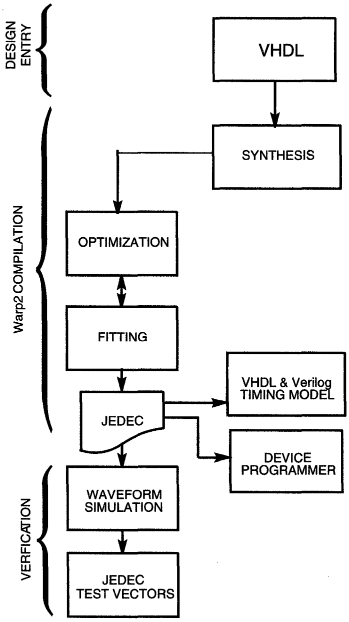


Figure 1. *Warp2* Design Flow

### VHDL Compiler

VHDL (VHSIC Hardware Description Language) is a powerful, non-proprietary language that is a standard for behavioral design entry and simulation, and is supported by every major vendor of CAE tools. VHDL allows designers to learn a single language that is useful for all facets of the design process.

VHDL offers designers the ability to describe designs at many different levels. At the highest level, designs can be entered as a description of their behavior. This behavioral description is not tied to any specific target device. As a result, simulation can be done very early in the design to verify correct functionality, which significantly speeds the design process.

*Warp2*'s VHDL syntax also includes support for intermediate level entry modes such as state tables and boolean entry. At the lowest level, designs can be described using gate-level RTL (Register Transfer Language) descriptions. *Warp2* gives the designer the flexibility to intermix all of these entry modes.

In addition, VHDL allows you to design hierarchically, building up entities in terms of other entities. This allows you to work either "top-down" (designing the highest levels of the system and its interfaces first, then progressing to greater and greater detail) or "bottom-up" (designing elementary building blocks of the system, then combining these to build larger and larger parts) with equal ease.

Because VHDL is an IEEE standard, multiple vendors offer tools for design entry and simulation at both high and low levels, and synthesis of designs to different silicon targets. The use of device independent behavioral design entry gives users the freedom to retarget designs to different devices. The wide availability of VHDL tools provides complete vendor independence as well. Designers can begin their project using *Warp2* for Cypress CPLDs and convert to high volume gate arrays using the same VHDL behavioral description with industry-standard synthesis tools.

The VHDL language allows users to define their own functions. User-defined functions allow users to extend the capabilities of the language and build reusable libraries of tested routines. As a result the user can produce complex designs faster than with ordinary "flat" languages. VHDL also provides control over the timing of events or processes. VHDL has constructs that identify processes as either sequential, concurrent, or a combination of both. This is essential when describing the interaction of complex state machines.

VHDL is a rich programming language. Its flexibility reflects the nature of modern digital systems and allows designers to create accurate models of digital designs. Because of its depth and completeness, it is easier to describe a complex hardware system accurately in VHDL than in any other hardware description language. In addition, models created in VHDL can readily be transported to other CAE Environments. *Warp2* supports IEEE 1076 and 1164 VHDL including loops, for ... generate statements, full hierarchical designs with packages, as well as synthesis for enumerated types and integers.

## Designing with *Warp2*

### Design Entry

*Warp2* descriptions specify

1. The behavior or structure of a design, and
2. The mapping of signals in a design to the pins of a PLD/CPLD (optional)

The part of a *Warp2* description that specifies the behavior or structure of the design is called an entity/architecture pair. Entity/architecture pairs, as their name implies, can be divided into two parts: an entity declaration, which declares the design's interface signals (i.e., defines what external signals the design has, and what their directions and types are), and a design architecture, which describes the design's behavior or structure.

The entity portion of a design file is a declaration of what a design presents to the outside world (the interface). For each external signal, the entity declaration specifies a signal name, a direction and a data type. In addition, the entity declaration specifies a name by which the entity can be referenced in a design architecture. In this section are code segments from four sample design files. The top portion of each example features the entity declaration.

*Behavioral Description*

The architecture portion of a design file specifies the function of the design. As shown in *Figure 1*, multiple design-entry methods are supported in *Warp2*. A behavioral description in VHDL often includes well known constructs such as If...Then...Else, and Case statements. Here is a code segment from a simple state machine design (soda vending machine) that uses behavioral VHDL to implement the design:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY drink IS
  PORT (nickel,dime,quarter,clock:in
std_logic;
    returnDime,returnNickel,giveDrink:out

std_logic);
END drink;


ARCHITECTURE fsm OF drink IS

TYPE drinkState IS (zero,five,ten,fifteen,
twenty,twentyfive,owedime);
SIGNAL drinkstatus:drinkState;


BEGIN

PROCESS BEGIN

 WAIT UNTIL clock = '1';

  giveDrink <= '0';
  returnDime <= '0';
  returnNickel <= '0';

  CASE drinkStatus IS

  WHEN zero =>
    IF (nickel = '1') THEN
      drinkStatus <= drinkStatus'SUCC
      (drinkStatus);
        -- goto Five
    ELSIF (dime = '1') THEN
      drinkStatus <= Ten;
    ELSIF (quarter = '1') THEN
```

```
      drinkStatus <= twentyfive;
    END IF;
  WHEN five =>
    IF (nickel = ’1’) THEN
      drinkStatus <= ten;
    ELSIF (dime = ’1’) THEN
      drinkStatus <= fifteen;
    ELSIF (quarter = ’1’) THEN
      giveDrink <= ’1’;
      drinkStatus <= drinkStatus’PRED
      (drinkStatus);
        -- goto Zero
    END IF;

  WHEN owedime =>
    returnDime <= ’1’;
    drinkStatus <= zero;

  when others =>
  -- This makes sure that the state
  -- machine resets itself if
  -- it somehow gets into an undefined state.
    drinkStatus <= zero;
  END CASE;
  END PROCESS;

END FSM;
```

VHDL is a strongly typed language. It comes with several predefined operators, such as + and /= (add, not-equal-to). VHDL offers the capability of defining multiple meanings for operators (such as +), which results in simplification of the code written. For example, the following code segment shows that “count <= count +1” can be written such that count is a std_logic_vector, and 1 is an integer.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.std_arith.all;

ENTITY sequence IS
  port (clk: in std_logic;
    s : inout std_logic);
end sequence;

ARCHITECTURE fsm OF sequence IS

SIGNAL count: std_logic_vector(3 downto 0);

BEGIN

PROCESS BEGIN

  WAIT UNTIL clk = ’1’;

    CASE count IS

      WHEN x″0″ | x″1″ | x″2″ | x″3″ =>
        s <= ’1’;
        count <= count + 1;
      WHEN x″4″ | x″5″ | x″6″ | x″7″ =>
        s <= ’0’;
        count <= count + 1;
      WHEN x″8″ | x″9″ =>
        s <= ’1’;
        count <= count + 1;
      WHEN others =>
        s <= ’0’;
        count <= (others => ’0’);
    END CASE;

END PROCESS;

END FSM;
```

In this example, the + operator is overloaded to accept both integer and std_logic arguments. *Warp2* supports overloading of operators.

### *Functions*

A major advantage of VHDL is the ability to implement functions. The support of functions allows designs to be reused by simply specifying a function and passing the appropriate parameters. *Warp2* features some built-in functions such as ttf (truth-table function). The ttf function is particularly useful for state machine or look-up table designs. The following code describes a seven-segment display decoder implemented with the ttf function:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.table_std.all;

ENTITY seg7 IS
  PORT(
    inputs: IN STD_LOGIC_VECTOR (0 to 3)
    outputs: OUT STD_LOGIC_VECTOR (0 to 6)
  );
END SEG7;

ARCHITECTURE mixed OF seg7 IS

CONSTANT truthTable:
  ttf_table (0 to 11, 0 to 10) := (
-- input&    output
-- ----------------------
  ″0000″&  ″0111111″,
  ″0001″&  ″0000110″,
  ″0010″&  ″1011011″,
  ″0011″&  ″1001111″,
  ″0100″&  ″1100110″,
  ″0101″&  ″1101101″,
  ″0110″&  ″1111101″,
  ″0111″&  ″0000111″,
  ″1000″&  ″1111111″,
  ″1001″&  ″1101111″,
  ″101-″&  ″1111100″, --creates E pattern
  ″111-″&  ″1111100″
  );

BEGIN

    outputs <= ttf(truthTable,inputs);

END mixed;
```

### *Boolean Equations*

A third design-entry method available to *Warp2* users is Boolean equations. *Figure 2* displays a schematic of a simple one-bit half adder. The following code describes how this one-bit half adder can be implemented in *Warp2* with Boolean equations:

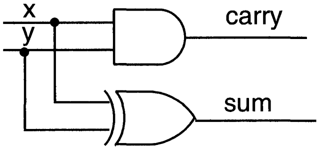


**Figure 2. One-Bit Half Adder**

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

--entity declaration
ENTITY half_adder IS
  PORT (x, y : IN std_logic;
    sum, carry : OUT std_logic);
END half_adder;
--architecture body
ARCHITECTURE behave OF half_adder IS
BEGIN
  sum <= x XOR y;
  carry <= x AND y;
END behave;
```

*Structural VHDL (RTL)*

While all of the design methodologies described thus far are high-level entry methods, structural VHDL provides a method for designing at a very low level. In structural descriptions (also called RTL), the designer simply lists the components that make up the design and specifies how the components are wired together. *Figure 3* displays the schematic of a simple 3-bit shift register and the following code shows how this design can be described in *Warp2* using structural VHDL:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.rtlpkg.all;

ENTITY shifter3 IS port (
    clk : IN STD_LOGIC;
    x : IN STD_LOGIC;
    q0  : OUT STD_LOGIC;
    q1  : OUT STD_LOGIC;
    q2  : OUT STD_LOGIC);
  END shifter3;

ARCHITECTURE struct OF shifter3 IS
  SIGNAL q0_temp, q1_temp, q2_temp : STD_LOGIC;
  BEGIN
    d1  : DFF PORT MAP(x,clk,q0_temp);
    d2  : DFF PORT MAP(q0_temp,clk,q1_temp);
    d3  : DFF PORT MAP(q1_temp,clk,q2_temp);
    q0 <= q0_temp;
    q1 <= q1_temp;
    q2 <= q2_temp;
  END struct;
```

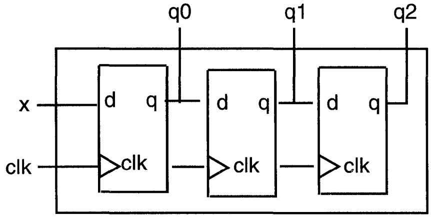


**Figure 3. Three-Bit Shift Register Circuit Design**

All of the design-entry methods described can be mixed as desired. The ability to combine both high- and low-level entry methods in a single file is unique to VHDL. The flexibility and power of VHDL allows users of *Warp2* to describe designs using whatever method is appropriate for their particular design.

## Compilation

Once the VHDL description of the design is complete, it is compiled using *Warp2*. Although implementation is with a single command, compilation is actually a multistep process as shown in *Figure 1*. The first part of the compilation process is the same for all devices. The input VHDL description is synthesized to a logical representation of the design. *Warp2* synthesis is unique in that the input language (VHDL) supports device-independent design descriptions. Competing programmable logic compilers require very specific and device-dependent information in the design description.

*Warp2* Synthesis is based on UltraGen Technology. This technology allows *Warp2* to infer "modules" like adders, counters, comparators, etc., from behavioral descriptions. *Warp2* then replaces that operator internally with an architecture specific circuit based on the target device. This circuit or "module" is also pre-optimized for either area or speed and *Warp2* uses the appropriate implementation based on user directives.

The second step of compilation is an interactive process of optimizing the design and fitting the logic into the targeted device. Logical optimization in *Warp2* is accomplished using Espresso algorithms. The optimized design is automatically fed to the *Warp2* fitter for targeting a PLD or CPLD. This fitter supports the automatic or manual placement of pin assignments as well as automatic selection of D or T flip-flops. After the optimization and fitting step is complete, *Warp2* creates a JEDEC file for the specified PLD or CPLD.

## Simulation

*Warp2* includes Cypress's NOVA Simulator. NOVA features a graphical waveform simulator that can be used to simulate PLD/CPLD designs generated in *Warp2*. The NOVA simulator provides functional simulation for PLDs/CPLDs and features interactive waveform editing and viewing. The simulator also provides the ability to probe internal nodes, automatically generate clocks and pulses, and generate JEDEC test vectors from simulator waveforms. (Higher level simulation support is available with *Warp2*Sim™ [CY3122] and *Warp3*® [CY3130].)

*Warp2* will also output standard VHDL and Verilog timing models. These models can be used with many third-party simulators to perform functional and timing verifications of the synthesized design.

## Programming

The result of *Warp2* compilation is a JEDEC file that implements the input design in the targeted device. Using this file, Cypress devices can be programmed on Cypress's *Impulse3*™ programmer or on any qualified third-party programmer.

## System Requirements

### For PCs

IBM PC or equivalent (486 or higher recommended)
16 Mbytes of RAM (32 Mbytes recommended)
100-Mbyte hard disk space
CD-ROM drive
Two- or three-button mouse
Windows 3.1x, Windows 95, or Windows NT

### For Sun Workstations

SPARC CPU
SunOS 4.1.3 or Solaris 2.5
16 Mbytes of RAM (32 Mbytes recommended)
CD-ROM drive

### For HP 9000 workstation (700 series)

HP-UX™ 10
16 Mbytes of RAM (32 Mbytes recommended)
CD-ROM drive

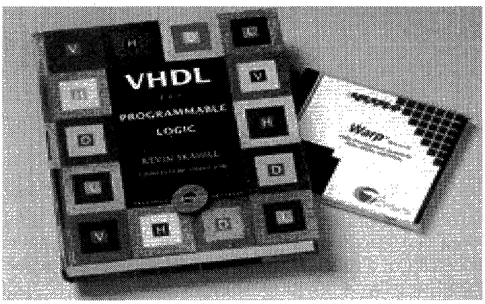
## Product Ordering Information

| Product Code | Description |
|---|---|
| CY3120R42 | *Warp2* development system |

*Warp2* includes:

CD-ROM with *Warp2* and on-line documentation
*VHDL for Programmable Logic* Textbook
Registration Card
Release Notes

Document #: 38-00218-F



*Warp2 and Warp3* are registered trademarks of Cypress Semiconductor Corporation.
*Warp2*Sim, UltraGen, MAX340, *Impulse3*, and FLASH370i are a trademarks of Cypress Semiconductor Corporation.
HP-UX is a trademark of Hewlett Packard Corporation.
SunOS and Solaris are trademarks of Sun Microsystems Corporation.

fax id: 6259

CYPRESS

# CY3122
# CY3127

# *Warp2*Sim™ VHDL Development System for PLDs

## Features

- **VHDL (IEEE 1076 and 1164) high-level language compiler**
  - **—Facilitates device independent design**
  - **—Designs are portable across multiple devices and/or CAE environments**
  - **—Facilitates the use of industry-standard simulation and synthesis tools for board and system-level design**
  - **—Supports functions and libraries for modular design methodology**
- ***Warp2*Sim™ provides synthesis of IEEE standards 1076 and 1164 VHDL including:**
  - **—enumerated types**
  - **—operator overloading**
  - **—for .... generate statements**
  - **—integers**
- **Several design entry methods support high and low-level design descriptions:**
  - **—Behavioral VHDL (IF ... THEN ... ELSE; CASE ...)**
  - **—Boolean**
  - **—Structural VHDL (RTL)**
  - **—Designs can include multiple VHDL entry methods in a single design**
- **State-of-the-art optimizations and reduction algorithms**
  - **—Automatic selection of optimal flip-flop type (D type/T type)**
  - **—Automatic pin assignment**
- **UltraGen™ Synthesis Technology**
  - **—Infers "modules" like adders, comparators, etc., from behavioral descriptions**
  - **—Replaces operator internally with an architecture specific circuit based on the target device**
  - **—User selectable speed and/or area optimization on a block-by-block basis**
- **Supports all Cypress Programmable Logic Devices**
  - **—Industry standard PLDs (16V8, 20V8, 22V10)**
  - **—MAX340 CPLDs**
  - **—FLASH370i CPLDs**
- **VHDL and Verilog timing model output for use with third-party simulators**
- **Timing Simulation provided with ViewLogic's ViewSim™ simulator**
  - **—Command line driven interface**
  - **—Graphical Waveform display**
  - **—Dynamic timing environment**
  - **—Flags setup and hold time violations**
  - **—Test bench support**
- **PC (Windows 3.1, 95, NT), Sun (SunOS™, Sun Solaris™), and HP platforms**
- **Motif on Unix workstations**
- **On-line documentation and help**

## Functional Description

*Warp2*Sim is a state-of-the-art VHDL compiler for designing with Cypress Programmable Logic Devices. *Warp2*Sim utilizes a subset of IEEE 1076 and 1164 VHDL as its Hardware Description Language (HDL) for design entry. *Warp2*Sim accepts VHDL input, synthesizes and optimizes the entered design, and outputs a JEDEC map for the desired PLD or CPLD. (see *Figure 1*). For simulation, *Warp2*Sim includes ViewLogic's ViewSim package as well as VHDL and Verilog models for use with third-party simulators.

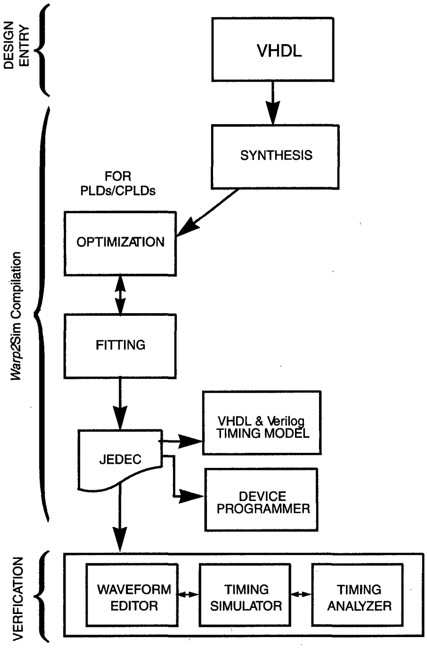


**Figure 1. Warp2Sim Design Flow**

### The Cockpit for WorkView PLUS and PowerView

The Viewlogic WorkView PLUS and PowerView graphical user interface (GUI) is built around a file/tool manager called "the cockpit". The cockpit is used to select the project and current tool set in use. The cockpit allows users to select from a variety of design environments called toolboxes.

### Toolbars for WorkView Office

The Viewlogic WorkView Office graphical user interface (GUI) for accessing various tools is the toolbar. There are two toolbars, one for WorkView Office, and one for *Warp*™.

### VHDL Compiler

VHDL (VHSIC Hardware Description Language) is a powerful, non-proprietary language that is a standard for behavioral design entry and simulation, and is supported by every major vendor of CAE tools. VHDL allows designers to learn a single language that is useful for all facets of the design process.

VHDL offers designers the ability to describe designs at many different levels. At the highest level, designs can be entered as a description of their behavior. This behavioral description is not tied to any specific target device. As a result, simulation can be done very early in the design to verify correct functionality, which significantly speeds the design process.

*Warp2*Sim's VHDL syntax also includes support for intermediate level entry modes such as state tables and boolean entry. At the lowest level, designs can be described using gate-level RTL (Register Transfer Language) descriptions. *Warp2*Sim gives the designer the flexibility to intermix all of these entry modes.

In addition, VHDL allows you to design hierarchically, building up entities in terms of other entities. This allows you to work either "top-down" (designing the highest levels of the system and its interfaces first, then progressing to greater and greater detail) or "bottom-up" (designing elementary building blocks of the system, then combining these to build larger and larger parts) with equal ease.

Because VHDL is an IEEE standard, multiple vendors offer tools for design entry and simulation at both high and low levels, and synthesis of designs to different silicon targets. The use of device independent behavioral design entry gives users the freedom to retarget designs to different devices. The wide availability of VHDL tools provides complete vendor independence as well. Designers can begin their project using *Warp2*Sim for Cypress CPLDs and convert to high volume gate arrays using the same VHDL behavioral description with industry-standard synthesis tools.

The VHDL language allows users to define their own functions. User-defined functions allow users to extend the capabilities of the language and build reusable libraries of tested routines. As a result the user can produce complex designs faster than with ordinary "flat" languages. VHDL also provides control over the timing of events or processes. VHDL has constructs that identify processes as either sequential, concurrent, or a combination of both. This is essential when describing the interaction of complex state machines.

VHDL is a rich programming language. Its flexibility reflects the nature of modern digital systems and allows designers to create accurate models of digital designs. Because of its depth and completeness, it is easier to describe a complex hardware system accurately in VHDL than in any other hardware description language. In addition, models created in VHDL can readily be transported to other CAE Environments. *Warp2*Sim supports IEEE 1076 and 1164 VHDL including loops, for ... generate statements, full hierarchical designs with packages, as well as synthesis for enumerated types and integers.

## Designing with *Warp2*Sim

### Design Entry

*Warp2*Sim descriptions specify

1. The behavior or structure of a design, and
2. The mapping of signals in a design to the pins of a PLD/CPLD (optional)

The part of a *Warp2*Sim description that specifies the behavior or structure of the design is called an entity/architecture pair. Entity/architecture pairs, as their name implies, can be divided into two parts: an entity declaration, which declares the design's interface signals (i.e., defines what external signals the design has, and what their directions and types are), and a design architecture, which describes the design's behavior or structure.

The entity portion of a design file is a declaration of what a design presents to the outside world (the interface). For each external signal, the entity declaration specifies a signal name, a direction and a data type. In addition, the entity declaration specifies a name by which the entity can be referenced in a design architecture. In this section are code segments from four sample design files. The top portion of each example features the entity declaration.

*Behavioral Description*

The architecture portion of a design file specifies the function of the design. Multiple design-entry methods are supported in *Warp2*Sim. A behavioral description in VHDL often includes well known constructs such as If ... Then ... Else, and Case statements. Here is a code segment from a simple state machine design (soda vending machine) that uses behavioral VHDL to implement the design:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY drink IS

PORT (nickel,dime,quarter,clock:in std_logi
c;
    returnDime,returnNickel,giveDrink:out

std_logic);
END drink;


ARCHITECTURE fsm OF drink IS

TYPE drinkState IS (zero,five,ten,fifteen,
twenty,twentyfive,owedime);
SIGNAL drinkstatus:drinkState;


BEGIN

PROCESS BEGIN

 WAIT UNTIL clock = '1';

  giveDrink <= '0';
  returnDime <= '0';
  returnNickel <= '0';

  CASE drinkStatus IS
  WHEN zero =>
    IF (nickel = '1') THEN
      drinkStatus <= drinkStatus'SUCC
      (drinkStatus);
        -- goto Five
    ELSIF (dime = '1') THEN
      drinkStatus <= Ten;
    ELSIF (quarter = '1') THEN
      drinkStatus <= twentyfive;
    END IF;
  WHEN five =>
    IF (nickel = '1') THEN
      drinkStatus <= ten;
    ELSIF (dime = '1') THEN
      drinkStatus <= fifteen;
    ELSIF (quarter = '1') THEN
      giveDrink <= '1';
      drinkStatus <= drinkStatus'PRED
      (drinkStatus);
        -- goto Zero
    END IF;

  WHEN owedime =>
    returnDime <= '1';
    drinkStatus <= zero;

  when others =>
  -- This makes sure that the state
  -- machine resets itself if

-- it somehow gets into an undefined state.
    drinkStatus <= zero;
  END CASE;
  END PROCESS;

END FSM;
```

VHDL is a strongly typed language. It comes with several predefined operators, such as + and /= (add, not-equal-to). VHDL offers the capability of defining multiple meanings for operators (such as +), which results in simplification of the code written. For example, the following code segment shows that "count <= count +1" can be written such that count is a std_logic_vector, and 1 is an integer.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.std_arith.all;

ENTITY sequence IS
  port (clk: in std_logic;
    s : inout std_logic);
end sequence;

ARCHITECTURE fsm OF sequence IS

SIGNAL count: std_logic_vector(3 downto 0);

BEGIN

PROCESS BEGIN

  WAIT UNTIL clk = '1';
    CASE count IS

      WHEN x"0" | x"1" | x"2" | x"3" =>
        s <= '1';
        count <= count + 1;
      WHEN x"4" | x"5" | x"6" | x"7" =>
        s <= '0';
        count <= count + 1;
      WHEN x"8" | x"9" =>
        s <= '1';
        count <= count + 1;
      WHEN others =>
        s <= '0';
        count <= (others => '0');
    END CASE;
END PROCESS;

END FSM;
```

In this example, the + operator is overloaded to accept both integer and std_logic arguments. *Warp2*Sim supports overloading of operators.

### *Functions*

A major advantage of VHDL is the ability to implement functions. The support of functions allows designs to be reused by simply specifying a function and passing the appropriate parameters. *Warp2*Sim features some built-in functions such as ttf (truth-table function). The ttf function is particularly useful for state machine or look-up table designs. The following code describes a seven-segment display decoder implemented with the ttf function:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.table_std.all;

ENTITY seg7 IS
  PORT(
    inputs: IN STD_LOGIC_VECTOR (0 to 3)
    outputs: OUT STD_LOGIC_VECTOR (0 to 6)
  );
END SEG7;

ARCHITECTURE mixed OF seg7 IS

CONSTANT truthTable:
  ttf_table (0 to 11, 0 to 10) := (
-- input&    output
-- ----------------------
  "0000"&  "0111111",
  "0001"&  "0000110",
  "0010"&  "1011011",
  "0011"&  "1001111",
  "0100"&  "1100110",
  "0101"&  "1101101",
  "0110"&  "1111101",
  "0111"&  "0000111",
  "1000"&  "1111111",
  "1001"&  "1101111",
  "101-"&  "1111100", --creates E pattern
  "111-"&  "1111100"
  );

BEGIN

    outputs <= ttf(truthTable,inputs);

END mixed;
```

4

*Boolean Equations*

A third design-entry method available to *Warp2*Sim users is Boolean equations. *Figure 1* displays a schematic of a simple one-bit half adder. The following code describes how this one-bit half adder can be implemented in *Warp2*Sim with Boolean equations:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

--entity declaration
ENTITY half_adder IS
  PORT (x, y : IN std_logic;
    sum, carry : OUT std_logic);
END half_adder;
--architecture body
ARCHITECTURE behave OF half_adder IS
BEGIN
  sum <= x XOR y;
  carry <= x AND y;
END behave;
```

*Structural VHDL (RTL)*

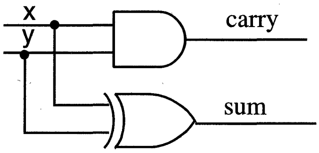


**Figure 2. One Bit Half Adder**

While all of the design methodologies described thus far are high-level entry methods, structural VHDL provides a method for designing at a very low level. In structural descriptions (also called RTL), the designer simply lists the components that make up the design and specifies how the components are wired together. *Figure 3* displays the schematic of a simple 3-bit shift register and the following code shows how this design can be described in *Warp2*Sim using structural VHDL:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.rtlpkg.all;

ENTITY shifter3 IS port (
    clk : IN STD_LOGIC;
    x : IN STD_LOGIC;
    q0 : OUT STD_LOGIC;
    q1 : OUT STD_LOGIC;
    q2 : OUT STD_LOGIC);
  END shifter3;

ARCHITECTURE struct OF shifter3 IS
  SIGNAL q0_temp, q1_temp, q2_temp : STD_LOGIC;
  BEGIN
    d1 : DFF PORT MAP(x,clk,q0_temp);
    d2 : DFF PORT MAP(q0_temp,clk,q1_temp);
    d3 : DFF PORT MAP(q1_temp,clk,q2_temp);
    q0 <= q0_temp;
    q1 <= q1_temp;
    q2 <= q2_temp;
  END struct;
```

All of the design-entry methods described can be mixed as desired. The ability to combine both high- and low-level entry methods in a single file is unique to VHDL. The flexibility and power of VHDL allows users of *Warp2*Sim to describe designs using whatever method is appropriate for their particular design.

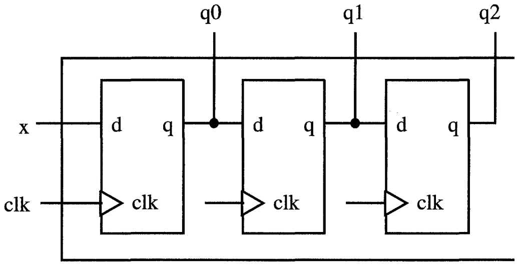


**Figure 3. Three-Bit Shift Register Circuit Design**

## Compilation

Once the VHDL description of the design is complete, it is compiled using *Warp2*Sim. Although implementation is with a single command, compilation is actually a multistep process. The first part of the compilation process is the same for all devices. The input VHDL description is synthesized to a logical representation of the design. *Warp2*Sim synthesis is unique in that the input language (VHDL) supports device-independent design descriptions. Competing programmable logic compilers require very specific and device-dependent information in the design description.

*Warp2*Sim Synthesis is based on UltraGen Technology. This technology allows *Warp2*Sim to infer "modules" like adders, counters, comparators, etc., from behavioral descriptions. *Warp2*Sim then replaces that operator internally with an architecture specific circuit based on the target device. This circuit or "module" is also pre-optimized for either area or speed and *Warp2*Sim uses the appropriate implementation based on user directives.

The second step of compilation is an interactive process of optimizing the design and fitting the logic into the targeted device. Logical optimization in *Warp2*Sim is accomplished using Espresso algorithms. The optimized design is automatically fed to the *Warp2*Sim fitter for targeting a PLD or CPLD. This fitter supports the automatic or manual placement of pin assignments as well as automatic selection of D or T flip-flops. After the optimization and fitting step is complete, *Warp2*Sim creates a JEDEC file for the specified PLD or CPLD.

## Simulation

The last step in the design process before programming is verifying the timing of your design. For this, *Warp2*Sim includes the ViewSim timing simulator. During compilation, delays that result from fitting the input design are "written" into an internal file for use by the *Warp2*Sim simulator. This information represents worst-case path delays for the design as fit in the selected device. Delays are based on the type of device and speed grade selected.

One of the ways to simulate is with the command-line interface to ViewSim. From the command line, the designer can specify the state of inputs (high, low, X, etc.) and watch how outputs behave over a specified time frame. In this way users can easily step through test cases and view the output results. Stimulus can be entered from the command line or from a file.

In addition to Viewsim, *Warp2*Sim will output VHDL and Verilog timing models for numerous other simulators.

- VHDL
  - —Mentor (Model Technology and QuickHDL)
  - —Cadence (Leapfrog)
  - —Synopsys ($V_{SS}$)
  - —Viewlogic (Vantage, SpeedWave)
- Verilog
  - —Cadence (Verilog - XL)
  - —Veribest Inc.

### Waveform Editor

A graphical method of simulation uses the Viewlogic waveform editor, ViewTrace™, in conjunction with ViewSim. With ViewTrace users can input stimulus from a file or graphically via digital waveforms. Outputs are viewed as digital waveforms that reflect the timing delays of the device as programmed. ViewTrace is interactive, allowing modifications of the stimulus and re-simulation of the results without re-running synthesis tools.

If user inputs violate device specifications the *Warp2*Sim simulator will detect the violation and warn the user. For example, if an input changes immediately before a CLK rise (violating the device set-up time) *Warp2*Sim will issue a warning and highlight the offending signal. The same occurs for all other timing violations.

## Programming

The result of *Warp2*Sim compilation is a JEDEC file that implements the input design in the targeted device. Using this file, Cypress devices can be programmed on Cypress's *Impulse3* programmer or on any qualified third-party programmer.

## System Requirements

### For PCs

IBM PC or equivalent (486 or higher recommended)
16 Mbytes of RAM (32 Mbytes recommended)
100-Mbyte hard disk space
CD-ROM drive
Two- or three-button mouse
Windows 3.1 with WorkView PLUS
Windows 95, NT with WorkView Office

### For Sun Workstations

SPARC CPU
SunOS 4.1.3 or Solaris 2.5
16 Mbytes of RAM (32 Mbytes recommended)
CD-ROM drive

### For HP 9000 workstation (700 series)

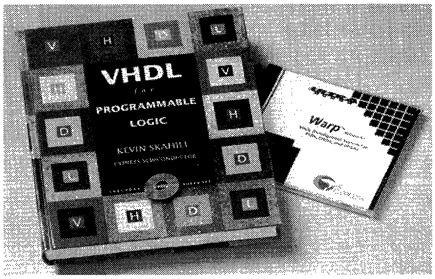
HP-UX™ 10.x
16 Mbytes of RAM (32 Mbytes recommended)
CD-ROM drive

## Product Ordering Information

| Product Code | Description |
|---|---|
| CY3122R42 | *Warp2*Sim development system for PCs |
| CY3127R42 | *Warp2*Sim development system for workstations |

*Warp2*Sim includes:

CD-ROM with *Warp2*Sim and on-line documentation
CD-ROM with ViewLogic WorkView Office release 7.3 and on-line documentation
*VHDL for Programmable Logic* Textbook
Registration Card
Release Notes
Hardware Key (PC )
*Warp* User's Guide and Reference Manual



Document #: 38-00523-B

*Warp2*Sim, *Warp*, UltraGen, MAX340, and FLASH370i are trademarks of Cypress Semiconductor Corporation.
HP-UX is a trademark of Hewlett Packard Corporation.
SunOS and Solaris are trademarks of Sun Microsystems Corporation.
ViewSim and ViewTrace are trademarks of ViewLogic Corporation.

# *Warp3* (CY3130/CY3135)

# *Warp3*® VHDL Development System for PLDs

## Features

- **Sophisticated PLD design and verification system based on VHDL**
- ***Warp3*® is based on Viewlogic's Powerview™ (Sun and HP), and Workview Plus™ and Workview Office™ (PC) design environments**
  - **—Advanced graphical user interface for Windows and Sun/HP Workstations**
  - **—Schematic capture (ViewDraw)**
  - **—Source-level Simulator (SpeedWave)**
  - **—Interactive timing simulator (ViewSim)**
  - **—Waveform stimulus and viewing (ViewTrace)**
  - **—Textual design entry using VHDL**
  - **—Mixed-mode design entry support**
- **The core of *Warp3* is an IEEE1076 and 1164 standard VHDL compiler**
  - **—VHDL is an open, powerful design language**
  - **—VHDL (IEEE standard 1076 and 1164) facilitates design portability across devices and/or CAD platforms**
  - **—VHDL facilitates the use of industry-standard simulation and synthesis tools for board and system-level design**
  - **—VHDL facilitates hierarchical design with support for functions and libraries**
- **Support for ALL Cypress PLDs and CPLDs including:**
  - **—Industry-standard 20- and 24-pin devices like the 22V10**
  - **—Cypress 7C33X family of 28-pin PLDs**
  - **—MAX340™ (MAX5000 Series) CPLDs**
  - **—FLASH370i™ CPLDs**

## Introduction

As the capacity and complexity of programmable logic increased dramatically over the last couple of years, users began to demand software tools that would allow them to manage this growing complexity. They also began to demand design-entry standards that would allow them to spend more time designing with PLDs rather than learning a vendor's proprietary software package. Thus, Hardware Description Languages (HDLs) in general, and VHDL (Very high speed integrated-circuit Hardware Description Language) in particular, have emerged as the standard methodology for integrated-circuit and system design.

While the design community debated whether VHDL could become the standard for PLDs, Cypress took an industry leading position by introducing the first native VHDL compiler for programmable logic—our *Warp*™ software tools.

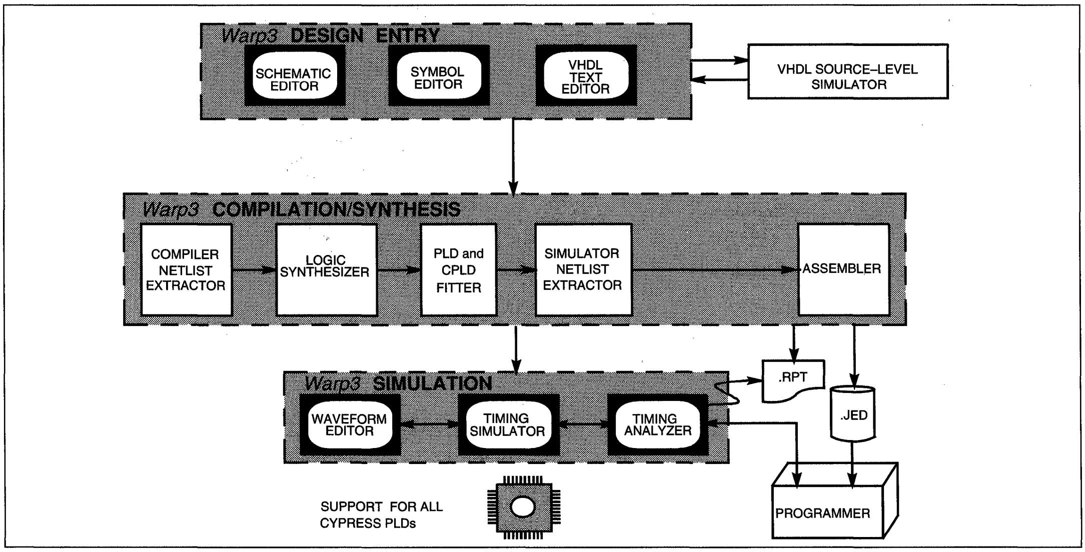


**Figure 1. *Warp3* Design Flow**

## Functional Description

*Warp3* is an integration of Cypress's advanced VHDL synthesis and fitting technology with Viewlogic's sophisticated CAE design environment. On the PC platform, *Warp3* includes Cypress's VHDL compiler and Viewlogic's WorkView PLUS software for Microsoft Windows 3.1, and WorkView Office software for Windows 95 and NT. On the Sun and HP platforms, *Warp3* includes Cypress's VHDL compiler and Viewlogic's PowerView software.

## Design Flow

*Figure 1* displays a block diagram of the typical design flow in *Warp3*. Designs can be entered in VHDL text or schematic capture. In fact, *Warp3* supports mixing these approaches on individual designs. Designs are then functionally verified using the *Warp3* functional simulator. The third step is to compile the design and target a PLD or CPLD. After synthesis, the waveform timing simulator is used to verify design timing as programmed in the chosen device. If the simulation results are satisfactory, the JEDEC or netlist file is used to program the targeted device. A detailed description of each step follows.

Specifically, the *Warp3* Design Flow includes the following:

Viewlogic GUI (Cockpit/Project Navigator)
IEEE1076 and 1164 VHDL Synthesis
Schematic Capture (ViewDraw)
Source-level Simulation (SpeedWave)
Hierarchy Navigator
Mixed-mode Design Entry
Waveform Editor (ViewTrace)
VHDL Timing Simulator (ViewSim)
Device fitters for all Cypress PLDs/CPLDs/PROMs

### The Cockpit for WorkView PLUS and PowerView

The Viewlogic WorkView PLUS and PowerView graphical user interface (GUI) is built around a file/tool manager called "the cockpit". The cockpit is used to select the project and current tool set in use. The cockpit allows users to select from a variety of design environments called toolboxes.

### Toolbars for WorkView Office

The Viewlogic WorkView Office graphical user interface (GUI) for accessing various tools is the toolbar. There are two toolbars, one for WorkView Office, and one for *Warp*. *Figure 2* shows these two toolbars.

## Design Entry

### Text Editor

Text entry is done with industry standard VHDL. *Warp3* can synthesize a rich set of the VHDL language in conformance with IEEE standard 1076 and 1164. This includes support for Behavioral, Boolean, State Table and Structural VHDL entry.

Text entry is ideal for describing complex logic functions such as state machines or truth tables. With VHDL, the behavior of a state machine can be described in concise, easily-readable code. In addition, the hierarchical nature of VHDL allows very complex functions to be described in a modular, top-down or bottom-up fashion. For more information on VHDL see the *Warp2*® (CY3120) or the *Warp2*Sim™ (CY3122) datasheet.

### Schematic Capture

*Warp3* users can also enter designs graphically with a sophisticated schematic capture system (ViewDraw). With schematic entry, designers can quickly describe a variety of common logic functions from simple logic gates to complex arithmetic functions. See *Figure 3*.

*Warp3* also supports the use of the LPM (Library of Parameterized Modules) Standard for Cypress.

Within *Warp3*, users have access to an extensive symbol library of standard components and macro functions. These include:

- adders/multipliers
- counters
- gates (AND, OR, XOR, INV, & BUF)
- io (singles, buses, three-states, clk-pads, hd-pad, gnd, & vcc)
- macrocells
- memory (assorted flip-flops and latches)
- mux (decoders and multiplexers)
- registers, shift registers and universal registers

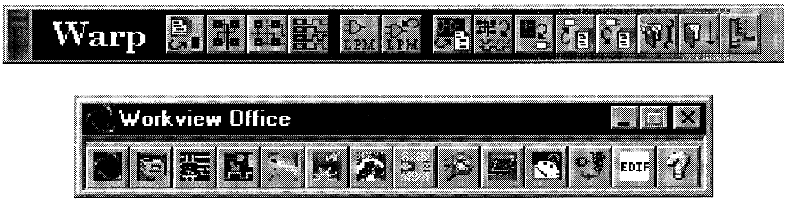
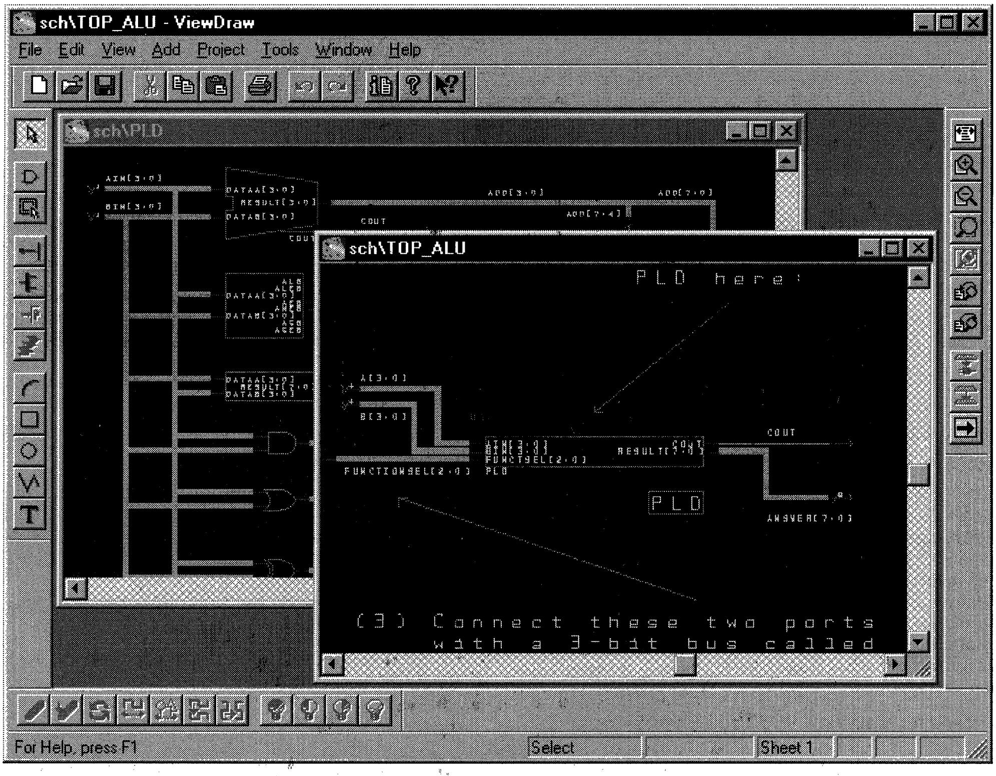


**Figure 2. *Warp* and WorkView Office toolbars for PC platforms**



(3) Connect these two ports with a 3-bit bus called

Figure 3. Schematics with ViewDraw

In addition, the designer may create custom functions that can be used in any *Warp3* design.

### Symbol Editor

The *Warp3* schematic capture tools also provide methods to create symbols for schematics and VHDL blocks. Using the Viewgen utility, symbols are automatically generated from lower-level schematic data. Using the VHDL2SYM utility, symbols are automatically generated from VHDL text files. Symbols are useful for creating a design hierarchy to easily describe complex designs.

### Mixed-mode Entry

Perhaps the most powerful design entry methodology in *Warp3* is the combination of the above methods. In most designs, some portions of the circuit are most easily described in schematic form while others are best described in text. Typically, standard logic components such as counters, adders and registers are best implemented by retrieving components from the *Warp3* schematic symbol library. Meanwhile, text entry is usually preferred for describing sections of the circuit design that implement control logic. In particular, state machines are often much easier to describe with behavioral VHDL than with schematic components. Combining these methods in a single design simplifies the input process and shortens the design cycle time.

As mentioned above, *Warp3* can automatically generate symbols for text and schematic designs. This capability facilitates hierarchical design entry by allowing users to represent complex functions with a symbol. The top level of the design may be represented by the connection of a small number of symbols representing the main functional blocks. To move to lower levels in the design the user can push into selected symbols. If the underlying design is described in VHDL, a text window will be launched with the design file. If the underlying design is a schematic, a ViewDraw window will be opened with the schematic design. There is no limit to the number of levels of hierarchy used or the number of symbols in a particular design.

## Design Verification

### Functional Simulation

Verifying functionality early in the design process can greatly reduce the number of design iterations necessary to complete a particular design. Using SpeedWave the functionality of the design can be verified with textual stimulus from the keyboard or from a file. ViewTrace can be used in conjunction with SpeedWave to simulate the design functionality graphically. The simulation process is described in detail below.

### VHDL Source-level Simulation

A unique and powerful feature of *Warp3* is the source-level VHDL simulator. The VHDL debugger works in concert with the *Warp3* simulator and waveform editor. The simulator allows users to graphically step through VHDL code and monitor the results textually or in waveforms. After each single step the simulator highlights the VHDL text representing the current state of the simulation. Simultaneously waveform and text windows can display the inputs and outputs of the design.

Note that a design does not have to be entered in VHDL text to use the VHDL simulator. Since *Warp3* converts design schematics, etc. to VHDL before compilation, this VHDL representation can be single stepped to verify design functionality.

### Hierarchy Navigator

Another powerful debugging tool within *Warp3* is the hierarchy navigator (Viewnav). The navigator allows users to select a net or node at one level of the design and automatically trace that net through all levels of the hierarchy. This is very useful for tracing signal paths when looking for design errors.

## Compilation

### VHDL Synthesis

- **For synthesis *Warp3* supports a rich subset of VHDL including**
  - **—Enumerated types**
  - **—Integers**
  - **—For . . . generate loops**
  - **—Operator overloading**

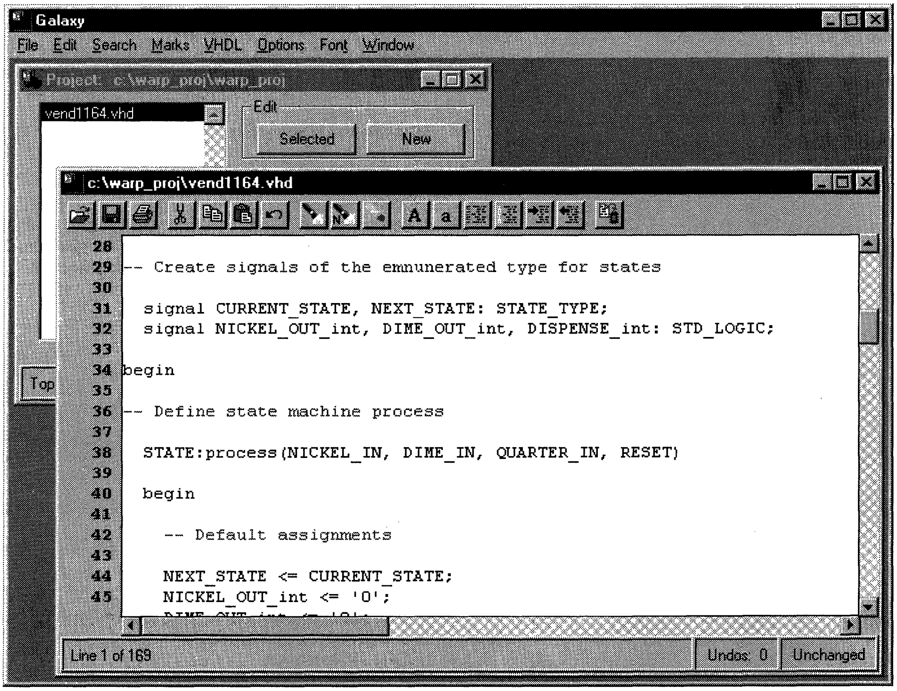
Once design entry is complete and functionality has been verified, the entire design is converted to VHDL using the “Export 1164” utility on schematic modules. At this point in the design there is a VHDL description of the entire design. This VHDL description is fed to the Cypress VHDL compiler for logic optimization, fitting, and translation to a device programming file. Although compilation is a multistep process, it appears as a single step to the user (as shown in *Figure 1*).

The first step in compilation is synthesizing the input VHDL into a logical representation of the design in terms of components found in the target device (AND gates, OR gates, flip-flops etc.). *Warp3* synthesis is unique in that the input language (VHDL) supports device-independent design descriptions. Competing programmable logic compilers require very specific and device-dependent information in the design file.

### Device Fitting

- **State-of-the-art optimization and reduction algorithms**
  - **—Optimization for flip-flop type (D type/T type)**
  - **—Automatic pin assignment**
  - **—User-specified state assignment (Gray code, binary, one-hot)**

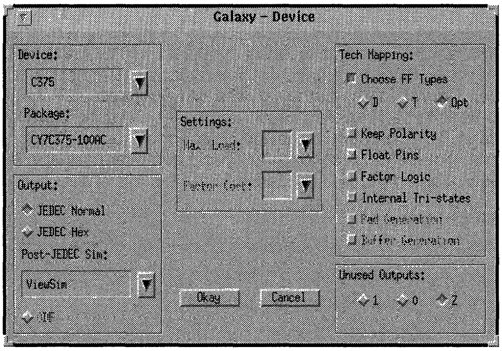
For PLDs and CPLDs, the second phase of the compilation is an iterative process of optimizing the design and fitting the logic into the targeted device (see *Figure 4*). Logical optimization in *Warp3* is accomplished with Espresso algorithms. Once optimized, the design is fed to the device-specific fitter which


Galaxy
File Edit Search Marks VHDL Options Font Window
Project: c:\warp_proj\warp_proj
vend1164.vhd
Edit
Selected
New
Top
c:\warp_proj\vend1164.vhd
A
a
Line 1 of 169
Undos: 0
Unchanged

```

-- Create signals of the emnunerated type for states

  signal CURRENT_STATE, NEXT_STATE: STATE_TYPE;
  signal NICKEL_OUT_int, DIME_OUT_int, DISPENSE_int: STD_LOGIC;

begin

-- Define state machine process

  STATE:process(NICKEL_IN, DIME_IN, QUARTER_IN, RESET)

  begin

    -- Default assignments

    NEXT_STATE <= CURRENT_STATE;
    NICKEL_OUT_int <= '0';
    DIME_OUT_int <= '0';
```

**Figure 4. Compile Dialog Box and Color-coded Editor**

applies the design to the selected device (see *Figure 5*). *Warp3* fitters support manual or automatic pin assignments as well as automatic selection of D-type or T-type flip-flops. After optimization and fitting are complete, *Warp3* will create a JEDEC file used to program the device.



Galaxy - Device

Device:
C375

Package:
CY7C375-100AC

Output:
● JEDEC Normal
○ JEDEC Hex

Post-JEDEC Sim:
ViewSim

○ 1164

Settings:
Max. Load:
Factor Cost:

Tech Mapping:
☑ Choose FF Types
○ D ○ T ● Opt
□ Keep Polarity
□ Float Pins
□ Factor Logic
□ Internal Tri-states
□ Pad Generation
□ Buffer Generation

Unused Outputs:
○ 1 ○ 0 ● Z

**Figure 5. Device Fitting Dialog Box**

### Automatic Error Tracking

Of course, the compilation process may not always go as planned. VHDL syntax errors should be identified and corrected in the pre-synthesis functional simulation stage. During the compilation phase, *Warp3* will detect errors that occur in the fitting process. *Warp3* features automatic error location that allows problems to be diagnosed and corrected in seconds. Errors from compilation are displayed immediately in a pop-up window (see *Figure 4*). If the user highlights a particular error, *Warp3* will automatically open the source code file and highlight the offending line in the entered design. If the device fitting process includes errors, a pop-up window will again describe them. In addition, a detailed report file is generated indicating the resources required to fit the input design and any problems that occurred in the process.

## Simulation

The last step in the design process before programming is verifying the timing of your design. For this, *Warp3* includes the ViewSim VHDL timing simulator. During compilation, delays that result from fitting the input design are "written" into an internal file for use by the *Warp3* simulator. This information represents worst-case path delays for the design as fit in the selected device. Delays are based on the type of device and speed grade selected.

One of the ways to simulate is with the command-line interface to ViewSim. From the command line, the designer can specify the state of inputs (high, low, X, etc.) and watch how outputs behave over a specified time frame. In this way users can easily step through test cases and view the output results. Stimulus can be entered from the command line or from a file.

In addition to ViewSim and SpeedWave, *Warp3* will output VHDL and Verilog timing models for numerous other simulators.

- VHDL
  —Mentor (Model Technology and Quicksim)
  —Cadence (Leapfrog)
  —Synopsys ($V_{SS}$)
  —Viewlogic (Vantage)
- Verilog
  —Cadence
  —Intergraph

### Waveform Editor

A graphical method of simulation uses the Viewlogic waveform editor, ViewTrace, in conjunction with ViewSim or SpeedWave. With ViewTrace users can input stimulus from a file or graphically via digital waveforms. Outputs are viewed as digital waveforms that reflect the timing delays of the device as programmed. ViewTrace is interactive, allowing modifications of the stimulus and re-simulation of the results without re-running synthesis tools.

If user inputs violate device specifications the *Warp3* simulator will detect the violation and warn the user. For example, if an input changes immediately before a CLK rise (violating the device set-up time) *Warp3* will issue a warning and highlight the offending signal. The same occurs for all other timing violations.

## Programming

After the design is compiled and verified, the targeted device is ready for programming. The program file generated in *Warp3* (a JEDEC file) is used as input to a device programmer. Cypress offers an inexpensive programmer, the Impulse3™ based on Data I/O's ChipLab™, that programs all Cypress PLDs. Alternatively, customers can use any one of several qualified third-party programmers from vendors such as Data I/O, BP Microelectronics, SMS and Logical Devices.

## System Requirements

### PC Platform

80486-based IBM PC
Microsoft Windows V3.1 if running WorkView PLUS
Microsoft Windows 95 or NT (4.0) if running WorkView Office
16 Mbytes of RAM (32 Mbytes recommended)
110 Mbytes Disk Space
CD-ROM drive

### Sun Platform

SPARC CPU
Sun OS 4.1.3 or Solaris 2.5
Motif GUI
16 Mbytes of RAM (32 Mbytes recommended)
120 Mbytes of Disk Space
CD-ROM drive

### HP Platform

HP700 Series CPU
OS 10.x
Motif GUI
16 Mbytes of RAM (32 Mbytes recommended)
130 Mbytes of Disk Space
CD-ROM drive

## Ordering Information

CY3130R42 *Warp3* PLD Development System on the PC includes:

CD-ROM with *Warp3* software and on-line documentation
CD-ROM with WorkView Office software and on-line documentation
*Warp3* Viewlogic hardware key
*Warp3* User's Guide
*Warp3* Reference Manual
*VHDL for Programmable Logic* textbook
License and registration form

CY3135R42 *Warp3* PLD Development System on a UNIX/SUN Workstation includes:

CD-ROM with *Warp3* software and on-line documentation
CD-ROM with PowerView software and on-line documentation for SunOS
CD-ROM with PowerView software and on-line documentation for Solaris
CD-ROM with PowerView software and on-line documentation for HP
PowerView documentation User's Guide
*Warp3* User's Guide
*Warp3* Reference Manual
*VHDL for Programmable Logic* textbook
License and registration form

Document #: 38-00242-E

FLASH370i, *Warp,* MAX340, *Warp2*Sim, and *Impulse3* are trademarks of Cypress Semiconductor Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.
PowerView and WorkView are registered trademarks of Viewlogic Systems, Inc.
ChipLab is a trademark of Data I/O Corporation.

# Third-Party Tool Support

Support for Cypress programmable logic devices is available in many software products from third-party vendors. Some companies include support for the entire design process in products that they sell. Others provide software for a portion of the design process (i.e., schematic capture, synthesis, or simulation) and interface with Cypress's *Warp*™ software tools to complete the design flow. This section will describe the design flow using these third-party software products and will describe the interface between these products and Cypress's *Warp* software.

In describing the design flow through these tools, it is useful to break the process into its major functional blocks. *Figure 1* shows these blocks. A similar figure is included for each of the third-party products, and the portions of this flow that are covered by that product are highlighted. Where applicable, the portion of the flow covered by Cypress's *Warp* software is also highlighted.

At the top of each figure are the "Text" and "Sch" blocks. These represent hardware description language (HDL) entry and schematic capture, respectively. Some tools offer design simulation at the design entry stage. This is known as "pre-synthesis" simulation, and is represented by "Simulation" block.

After design entry and simulation are complete, the design description is synthesized. The "Synthesis" block represents the translation of the design description into logic equations, and the optimization of these equations to a device architecture. The next step is device fitting, represented by the "Fitting" block. Here, the logic equations are mapped into the resources of the device. The result of "Fitting" is a file used to program the device.

The last block in the flow diagram is "Device Sim." This corresponds to simulation of the design according to its implementation in the device. Some simulators will take the timing parameters (i.e., propagation delay) of the device into account, and will provide simulation results consistent with this timing. Others will verify that the programming file is functionally consistent with the design description, but will not contain device timing information.

Finally, along the left of the diagram is the Design Flow Manager. The Design Flow Manager keeps track of the design process for the user. This module typically informs the designer which steps of the design flow have been completed and which have not. This flow manager can also be used to launch vendor tools such as *Warp* synthesis.

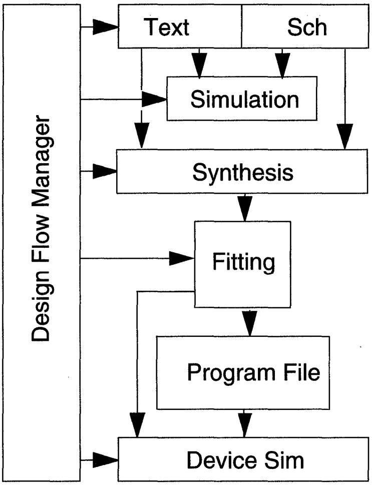


**Figure 1.**

| Contents | |
|---|---|
| **Company** | **Product** |
| Acugen | ATGEN |
| Cadence | Concept, Composer |
| Exemplar Logic | Galileo |
| Flynn Systems | FS-ATG |
| IsData | LOG/iC |
| Logical Devices | CUPL |
| Mentor Graphics | Design Architect |
| OrCAD | PLD386+ |
| Synario Design | ABEL4/ABEL5/ABEL6 |
| Synario Design | Synario |
| Synopsys | Design Compiler |
| ViewLogic | WorkView, PowerView |

# Acugen

## Product

ATGEN

## Device Support

Small PLDs,FLASH370i™

## Input Format

JEDEC file

## Required Cypress Product

*Warp2*®, *Warp2*Sim™, or *Warp3*®

## Design Flow Description

Acugen's ATGEN software can automatically generate test vectors to be used with device programmers or with automatic test equipment (ATE) for Cypress PLDs and CPLDs. The JEDEC file output by *Warp* is read into the ATGEN software, where test vectors are generated for the design. ATGEN can output a JEDEC file with test vectors to be used on a device programmer, or a test program to be used on a tester.

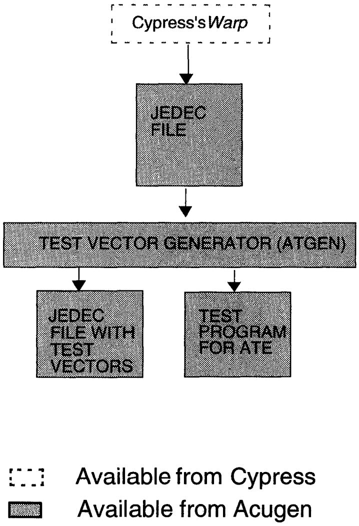


**Figure 2.**

# Cadence Design Systems

## Product

Cadence Concept or Cadence Composer

## Device Support

Small PLDs, MAX340™, FLASH370i, Ultra37000™

## Input Format

Schematic Entry and VHDL

## Required Cypress Product

Cypress Cadence Bolt-in Kit

## Design Flow Description

Both schematic and textual designs (VHDL) are supported using the Cadence Concept or Composer schematic capture tools. The *Warp* software interfaces to these tools and provides the synthesis and fitting steps of the design flow. Timing information and simulation models are then produced by *Warp* and fed back into the Cadence environment for device-level simulation. Programming files are also produced for device programming.

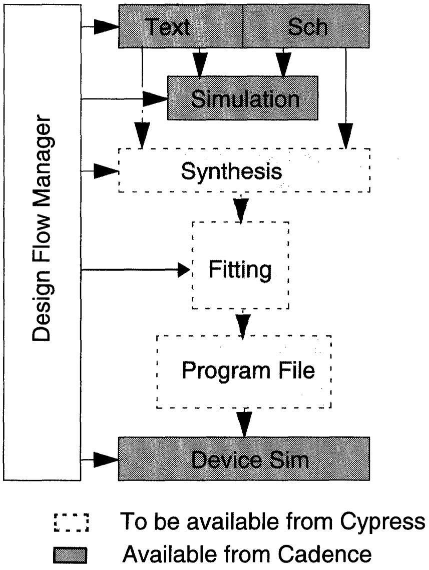


**Figure 3.**

# Exemplar Logic

## Product

Galileo

## Device Support

Small PLDs, MAX340, FLASH370i, (Ultra37000)

## Input Format

VHDL, Verilog, OpenABEL, Berkeley PLA, EIL, EDIF, ADL, XNF

## Required Product

*Warp2*, *Warp2*Sim, or *Warp3*

## Design Flow Description

Exemplar Logic's Galileo synthesis software accepts any of the above file formats and synthesizes the logic to any of the Cypress programmable logic devices. Galileo outputs a PLA file that can be read into any of the *Warp* tools, where device fitting and JEDEC (programming file) output occurs. Galileo can also provide timing simulation for the design once fitting has been performed.

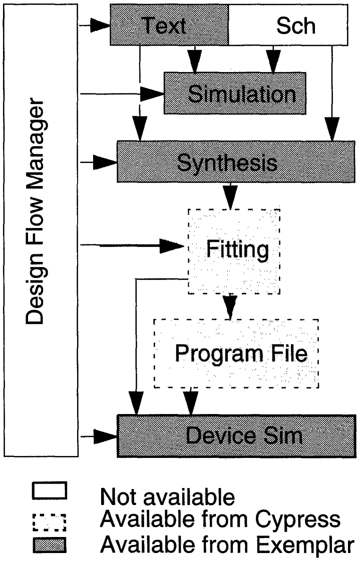


**Figure 4.**

# Flynn Systems

## Product

FS-ATG

## Device Support

Small PLDs, MAX340, FLASH370i

## Input Format

JEDEC file

## Required Cypress Product

*Warp2*, *Warp2*Sim, or *Warp3*

## Design Flow Description

Using FS-ATG from Flynn Systems, users can automatically generate test vectors to be used on device programmers or in-circuit testers.The JEDEC file output by *Warp* is read into the FS-ATG software, where test vectors are generated automatically. The user can enter constraints for this generator as desired. FS-ATG then outputs the JEDEC file with test vectors, to be used on a device programmer. It also outputs test vector files that can be translated for use with an in-circuit tester. This translator will take the test vectors and convert them to an automatic test equipment program.

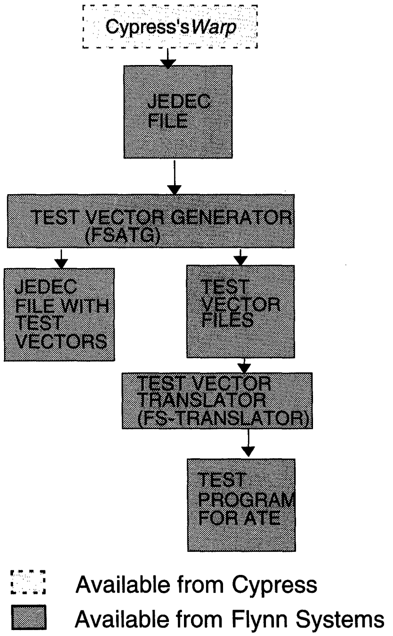

Cypress's *Warp*
JEDEC FILE
TEST VECTOR GENERATOR (FSATG)
JEDEC FILE WITH TEST VECTORS
TEST VECTOR FILES
TEST VECTOR TRANSLATOR (FS-TRANSLATOR)
TEST PROGRAM FOR ATE
Available from Cypress
Available from Flynn Systems

**Figure 5.**

# IsData

## Product

LOG/iC2 or LOG/iC Classic

## Device Support

Small PLDs, MAX340, FLASH370i, (Ultra37000)

## Input Format

LOG/iC-HDL

## Required Cypress Product

*Warp2*, *Warp2*Sim, or *Warp3*

## Design Flow Description

Design entry in LOG/iC is done using schematic capture or IsData's proprietary LOG/iC hardware description language. After design description and debugging of the source code, the design is synthesized and fit to a Cypress PLD or CPLD via the *Warp2*, *Warp2*Sim, or *Warp3* software. After synthesis and fitting, the software outputs a programming file. Simulation is also available to complete the design flow.

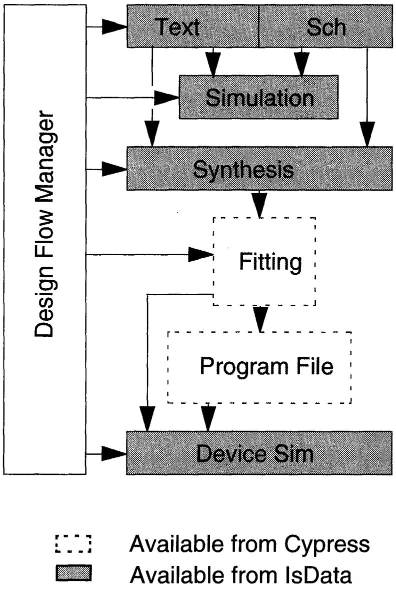


**Figure 6.**

# Logical Devices

## Product

CUPL

## Device Support

Small PLDs, MAX340, FLASH370i

## Input Format

CUPL-HDL

## Required Cypress Product

None

## Design Flow Description

Design entry is done using Logical Devices' proprietary CUPL hardware description language. Source-code simulation is used to catch design errors before the design is synthesized and fit to a Cypress device.

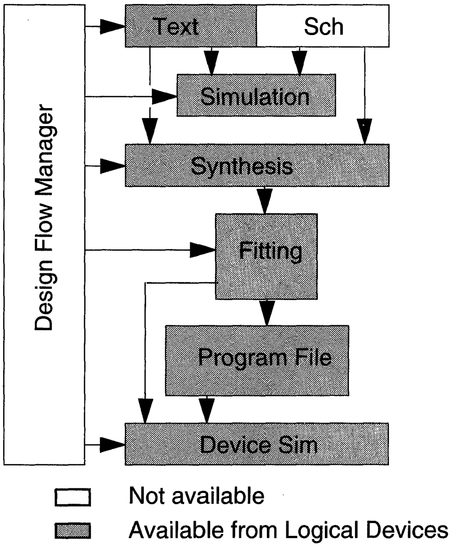


**Figure 7.**

# Mentor Graphics

### Product

Design Architect

### Device Support

Small PLDs, MAX340, FLASH370i, (Ultra37000)

### Input Format

Schematic Entry and VHDL

### Required Cypress Product

Cypress Mentor Graphics Bolt-in Kit (CY3144)

### Design Flow Description

Both schematic and textual designs (VHDL) are supported using the Mentor Design Architect schematic capture tool. The *Warp* software interfaces to this tool and provides the synthesis and fitting steps of the design flow. Timing information and simulation models are then produced by *Warp* and fed back into the Mentor environment for device-level simulation. Programming files are also produced for device programming.

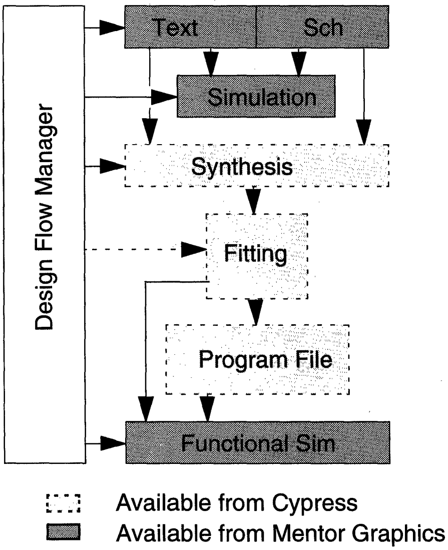


**Figure 8.**

# OrCAD

### Product

PLD386+

### Device Support

Small PLDs, MAX340, FLASH370i, (Ultra37000)

### Input Format

Orcad-HDL

### Required Cypress Product

*Warp2*, *Warp2*Sim, or *Warp3*

### Design Flow Description

PLD386+ interfaces to the other tools in OrCAD's design software suite to provide complete design entry, synthesis, and simulation. Designs are entered using OrCAD's proprietary design language, or by OrCAD's schematic capture software. After design description and debugging of the source code, the design is synthesized and fit to a PLD or CPLD via PLA file. PLD386+ outputs the programming file, which is also used for device timing simulation to complete the design flow.

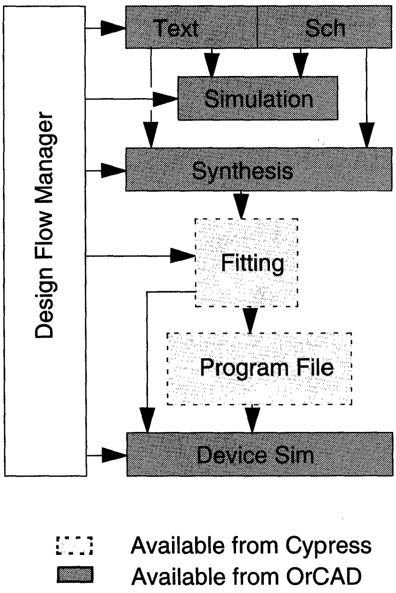


**Figure 9.**

# Synario Design Automation

## Product

ABEL4, ABEL5, and ABEL6

## Device Support

Small PLDs, MAX340, FLASH370i

## Input Format

ABEL-HDL

## Required Product

None for Small PLDs or MAX340

Cypress ABEL Fitter Kit for FLASH370i (CY3140)

## Design Flow Description

Designs entered in ABEL-HDL can first be functionally simulated using PLASIM and then go through logic optimization and minimization. The output files from ABEL will then go through fitting. The resulting output files can then be used for device-level simulation and device programming.

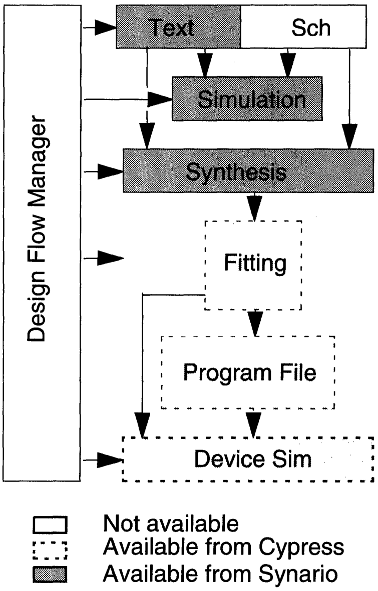


**Figure 10.**

# Synario Design Automation

## Product

Synario

## Device Support

Small PLDs, MAX340, FLASH370i, (Ultra37000)

## Input Format

Schematic Entry, VHDL, and ABEL-HDL

## Required Product

*Warp2*

## Design Flow Description

The user is guided through the design process by the Project Navigator. Designs can be entered in schematic entry, VHDL, or ABEL-HDL. Designs entered can be functionally simulated and then optimized by Synario. The designs will then go through fitting. The resulting output files can be used for device-level simulation and device programming.

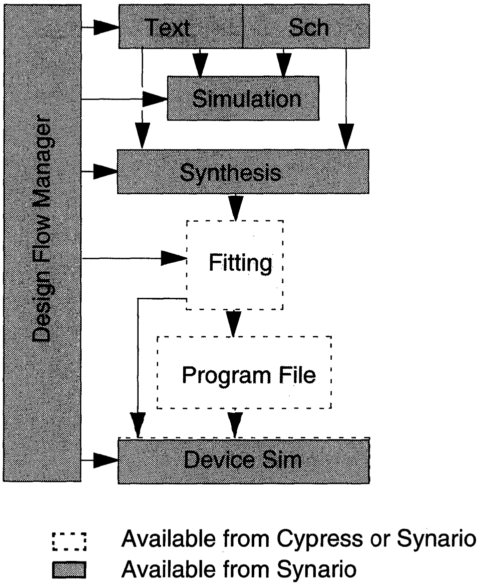


**Figure 11.**

# Synopsys

### Product

Design Compiler

### Device Support

FLASH370i, Ultra37000

### Input Format

VHDL and Verilog

### Required Product

Cypress Synopsys Bolt-in Kit

### Design Flow Description

Designs are entered in either HDL, or netlist format. It can then be functionally simulated using the VHDL System Simulator (VSS). The next step in the process is logic optimization and technology mapping. The output then goes into the *Warp* fitter, and programming and simulation files are generated for device programming and device-level simulation with VSS.

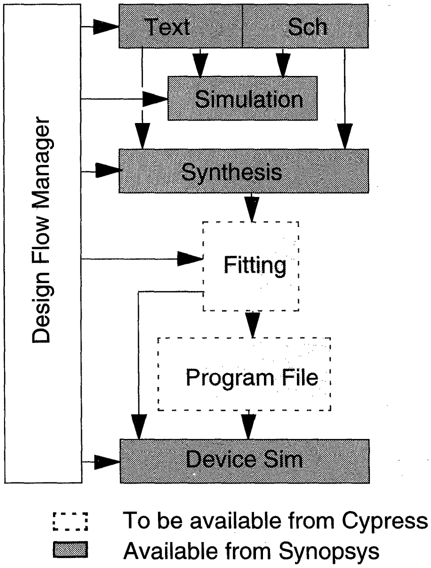


**Figure 12.**

# ViewLogic

### Product

WorkView PLUS, WorkView Office, PowerView

### Device Support

Small PLDs, MAX340, FLASH370i, Ultra37000

### Input Format

LPM Schematic entry, VHDL, or both

### Required Cypress Product

*Warp2*

### Design Flow Description

LPM schematics and/or VHDL files can be used for design entry. Pre-synthesis simulation is performed via SpeedWave. The design will then be exported into a VHDL file, which will go into *Warp2* for optimization and synthesis. The design will then go through fitting, resulting in a JEDEC file for device programming and timing models for device-level simulation.

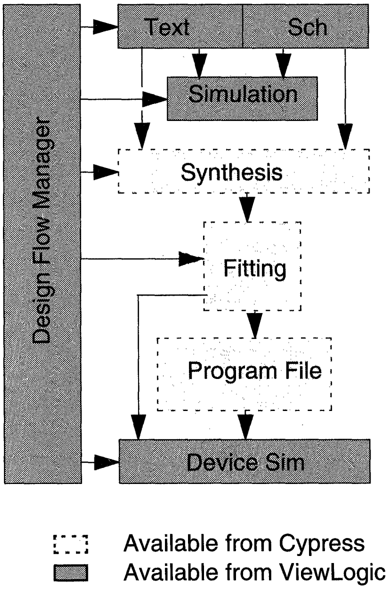


**Figure 13.**

## Company Addresses

ACUGEN Software Inc.
427-3 Amherst Street
Suite 391
Nashua, NH 03063
(603) 881-8821

Cadence Design Systems, Inc.
555 River Oaks Parkway
San Jose, CA 95134
(408) 943-1234

Exemplar Logic
815 Atlantic Avenue
Suite 105
Alameda, CA 94501-2274
(510) 337-3700

Flynn Systems Corporation
Exit 4 Office Building
7 1/2 Harris Road
Nashua, NH 03062
(603) 891-1111

ISDATA GmbH
Haid-und-Neu-Strasse 7
D-7500 Karlsruhe 1
Germany
(0721) 69 30 92

Logical Devices Inc.
692 S. Military Trail
Deerfield Beach, FL 33442
(305) 428-6868

Mentor Graphics
8005 SW Boeckman Road
Wilsonville, OR 97070-7777
(503) 685-8000

OrCAD
3175 NW Aloclek Dr.
Hillsboro, OR 97124
(503) 690-9881

Synario Design Automation
10525 Willows Rd. N.E.
P.O. Box 97046
Redmond, WA 98073
(206) 881-6444

Synopsys
700 E. Middlefield Rd.
Mountain View, CA 94043-4033
(415) 962-5000

ViewLogic Systems, Inc.
293 Boston Post Road West
Marlborough, MA 01752

*Warp, Warp2*Sim, MAX340, FLASH370i, and Ultra37000 are trademarks of Cypress Semiconductor Corporation.
MAX is a trademark of Altera Corporation.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.

Document #: 38–00371–B

fax id: 6256

CYPRESS **PRELIMINARY** **CY3140**

# ABEL™/Synario™ Design Kit for FLASH370i™

## Features

- **Device independent design entry formats:**
  - **—ABEL-HDL for ABEL-4, ABEL-5, and ABEL-6**
  - **—Schematic entry, VHDL, and ABEL-HDL for Synario™**
- **Full integration supporting all ABEL™ and Synario™ design features**
- **Supports the full family of FLASH370i™ devices**
- **Graphical device simulator included (CYPSIM)**
- **Automatic installation into existing ABEL and Synario environment**
- **Available on PC and Sun workstation design platforms**

## Introduction

The seamless integration of Data I/O's ABEL or Synario software design environment and the Cypress FLASH370i ABEL fitter offers a powerful solution for fitting ABEL and Synario designs into the Cypress CPLD device family.

## Functional Description

The design process in the ABEL environment begins with entering ABEL-HDL (and optional test vectors) using any text editor. The process in Synario is guided by the Project Navigator, and begins with design entry in either schematic, VHDL, or ABEL-HDL. The design can then be functionally simulated at the source-level. It then goes through logic optimization and minimization. The output file then goes into the FLASH370 fitter. Test vectors specified in the ABEL-HDL files are also automatically processed for use in post-fitting device simulation.

The FLASH370i fitter generates a JEDEC file for device programming and post-fitting simulation in CYPSIM. The test vectors will also be read in for functional verification.

The post-fitting simulator, CYPSIM, operates under the Windows environment. It takes JEDEC files as input and can read in and write out stimulus files (e.g., test vectors from ABEL-HDL) for functional verification of the design. Users can edit input waveforms graphically and specify simulation length and resolution interactively. Signals can also be grouped, manipulated, and viewed in various formats.

## System Requirements

### PC Platform

486-based IBM PC
Microsoft Windows 3.1, 95, NT
16 Mbytes of RAM
8 Mbytes of disk space
1.44-Mbyte 3.5-inch floppy disk drive

### Sun Platform

SPARC CPU
Sun OS 4.1 or later
16 Mbytes of RAM
8 Mbytes of disk space

## Ordering Information

CY3140 ABEL/Synario Design Kit for FLASH370i includes:
ABEL Fitter Software on two 3.5-inch floppy disks for PCs
ABEL Fitter Software on 3 3.5-inch floppy disks for Sun
ABEL Fitter User's Guide
Registration Card

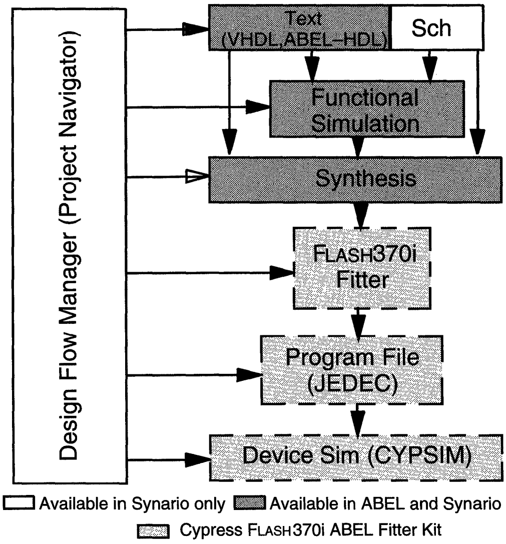

Design Flow Manager (Project Navigator)
Text (VHDL,ABEL-HDL)
Sch
Functional Simulation
Synthesis
FLASH370i Fitter
Program File (JEDEC)
Device Sim (CYPSIM)
□ Available in Synario only
▨ Available in ABEL and Synario
▨ Cypress FLASH370i ABEL Fitter Kit

**Figure 1. ABEL/Synario Design Flow**

Document #: 38-00431–B

FLASH370i is a trademark of Cypress Semiconductor Corporation.
ABEL and Synario are trademarks of Data I/O Corporation.

CYPRESS **_PRELIMINARY_** **CY3144**

# Cypress Mentor Graphics Bolt-in Kit

## Features

- **Seamless integration with your Mentor Graphics tools**
- **Powerful schematic symbol library**
- **IEEE-compliant VHDL**
- **Supports the UltraLogic™ family of SPLDs and CPLDs**
- **Industry-leading synthesis for programmable logic**
- **100% automatic fitting**
- **VHDL and Verilog post-layout timing models**
- **Complete solution from design entry to programming**

## Introduction

Cypress offers powerful integrated solutions for programmable logic. The Cypress Mentor Graphics Bolt-in Kit gives you everything to design with Cypress's UltraLogic PLDs in one seamless device-independent design environment. It allows you to take advantage of Mentor Graphics' powerful Design Architect™ schematic entry tool, Cypress's industry-leading *Warp*™ VHDL synthesis tool, and a wide variety of simulators.

## Functional Description

### Design Entry

Design with ease using schematic symbols, VHDL design descriptions, or a combination of both, supported with the Mentor Design Architect tool available with your Mentor flow.

### Synthesis

Your entire design is automatically converted into VHDL and efficiently synthesized into a SPLD or CPLD device using *Warp*. UltraGen™ synthesis technology will ensure that you achieve the best results for every Cypress device. For a description of UltraGen and synthesis, see the *Warp* datasheets.

### Fitting

Easily retarget your design to different devices. The 100% automatic fitting tools produce optimal results in minutes.

### Post-synthesis Simulation

*Warp* outputs VHDL and Verilog timing simulation models. Verify your design with timing using your choice of Mentor Graphics' QuickHDL™ or any other VHDL or Verilog simulator.

### Programming

*Warp* generates JEDEC programming files for all Cypress devices which can be used for in-system reprogramming (ISR™) or with various device programmers.

## System Requirements

**For Sun Workstations**

SPARC CPU
SunOS 4.1.3 or Solaris 2.5
CD-ROM drive

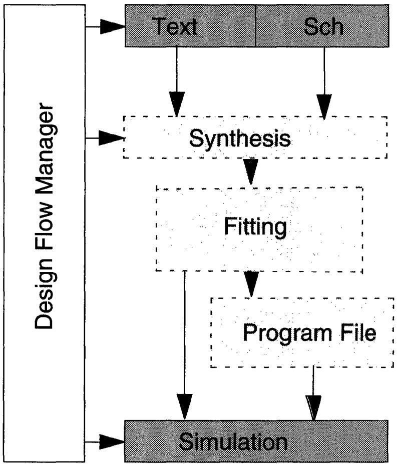
**For HP 9000 workstation (700 series)**

HP-UX™ 9.x
CD-ROM drive

## Ordering Information

CY3144 Cypress Mentor Graphics Bolt-in Kit includes:

CD-ROM with Bolt-in software and on-line documentation
CD-ROM with *Warp* software and on-line documentation
*Warp* User's Guide and Reference Manual
Release Notes
*VHDL for Programmable Logic* Textbook



⬚ Available from Cypress
▨ Available from Mentor Graphics

**Figure 1. Cypress Mentor Graphics Design Flow**

Document #: 38-00593

ISR, UltraGen, UltraLogic, and *Warp* are trademarks of Cypress Semiconductor Corporation.
HP-UX is a trademark of Hewlett Packard Corporation.
SunOS and Solaris are trademarks of Sun Microsystems Corporation.
Design Architect, and QuickHDL are trademarks of Mentor Graphics Corporation.

# PLD Programming Information

## Introduction

PLDs, or programmable logic devices, provide an attractive alternative to logic implemented with discrete devices. Cypress Semiconductor is in the enviable position of being able to offer PLDs in several different process technologies, thus assuring our customers of a wide range of options for leading-edge speed as well as very low power consumption. Cypress optimizes the mix of technology and device architecture to insure that the programmable logic requirements of today's highest-performance electronics systems can be fully supported by a single PLD vendor.

Cypress offers a wide variety of PLDs based on our leading-edge CMOS EPROM process technology. This technology facilitates the lowest power consumption and the highest logic density of any nonvolatile PLD technology on the market today, at speeds that are as fast as state-of-the-art bipolar technology would provide. Furthermore, these devices offer the user the option of device erasure and reprogrammability in windowed packages. Cypress offers PLDs based on CMOS Flash technology which features electrical erasure and reprogrammability. Thus Cypress offers solutions for state-of-the-art systems regardless of what the optimal balance is between speed, power, and density for any particular system.

## Programmable Technology

### EPROM Process Technology

EPROM technology employs a floating or isolated gate between the normal control gate and the source/drain region of a transistor. This gate may be charged with electrons during the programming operation, permanently turning off the transistor. The state of the floating gate, charged or uncharged, is permanent because the gate is isolated in an extremely pure oxide. The charge may be removed if the device is irradiated with ultraviolet energy in the form of light. This ultraviolet light allows the electrons on the gate to recombine and discharge the gate. This process is repeatable and therefore can be used during the processing of the device, repeatedly if necessary, to assure programming function and performance.

### Two Transistor Cells

Cypress uses a two-transistor EPROM cell. One transistor is optimized for reliable programming, and one transistor is optimized for high speed. The floating gates are connected such that charge injected on the floating gate of the programming transistor is conducted to the read transistor biasing it off.

### Flash Process Technology

The Flash cell is programmed in the same manner as the EPROM cell, and is electrically erased via Fowler-Nordheim tunneling. This next-generation PLD technology will combine a number of key advantages for future Cypress PLDs. The principal advantages will be leading-edge speed, low CMOS power consumption, and electrical alterability for simplified inventory management.

## Programming Algorithm—EPROM and Flash Technology

### Byte Addressing and Programming

Most Cypress programmable logic devices are addressed and programmed on a byte or extended byte basis where an extended byte is a field that is as wide as the output path of the device. Each device or family of devices has a unique address map that is available in the product datasheet. Each byte or extended byte is written into the addressed location from the pins that serve as the output pins in normal operation. To program a cell, a 1 or HIGH is placed on the input pin and a 0 or LOW is placed on pins corresponding to cells that are not to be programmed. Data is also read from these pins in parallel for verification after programming. A 1 or HIGH during program verify operation indicates an unprogrammed cell, while a 0 or LOW indicates that the cell accessed has been programmed.

### Blank Check

Before programming, all programmable logic devices may be checked in a conventional manner to determine that they have not been previously programmed. This is accomplished in a program verify mode of operation by reading the contents of the array. During this operation, a 1 or HIGH output indicates that the addressed cell is unprogrammed, while a 0 or LOW indicates a programmed cell.

### Programming the Data Array

Programming is accomplished by applying a supervoltage to one pin of the device causing it to enter the programming mode of operation. This also provides the programming voltage for the cells to be programmed. In this mode of operation, the address lines of the device are used to address each location to be programmed, and the data is presented on the pins normally used for reading the contents of the device. Each device has a read/write pin in the programming mode. This signal causes a write operation when switched to a supervoltage and a read operation when switched to a logic 0 or LOW. In the logic HIGH or 1 state, the device is in a program inhibit condition and the output pins are in a high-impedance state. During a write operation, the data on the output pins is written into the addressed array location. In a read operation, the contents of the addressed location are present on the output pins and may be verified. Programming therefore is accomplished by placing data on the output pins and writing it into the addressed location. Verification of data is accomplished by examining the information on the output pins during a read operation.

The timing for actual programming is supplied in the unique programming specification for each device.

### Phantom Operating Modes

All Cypress programmable logic devices on the EPROM and BiCMOS technology contain a Phantom array for post assembly testing. This array is accessed, programmed, and operated in a special Phantom mode of operation. In this mode, the normal array is disconnected from control of the logic, and in its place the Phantom array is connected. In normal operation the Phantom array is disconnected and control is only via the normal array. This special feature allows every device to be tested for both functionality and performance after packaging and, if desired, by the user before programming and use. The Phantom modes are entered through the use of supervoltages and are unique for each device or family of devices. See specific datasheets for details.

### Special Features

Cypress programmable logic devices, depending on the device, have several special features. For example, the security mechanism defeats the verify operation and therefore secures the contents of the device against unauthorized tampering or access. In advanced devices such as the PALCE22V10, PLDC20G10, and CY7C335, the macrocells are programmable through the use of the architecture bits. This allows users to more effectively tailor the device architecture to their unique system requirements. Specific programming is detailed in the device datasheet.

## Programmers

All of Cypress's programmable logic devices can be programmed on the Cypress *Impulse3*™ programmer. Many third-party programmers also support these products. In addition, all CY7C370i In-System Reprogrammable™ devices can be programmed with the InSRkit™ on-board programmer for the PC.

*Impulse3*, In-System Reprogrammable, and InSRkit are trademarks of Cypress Semiconductor Corporation.

CYPRESS

# CY3500

# *Impulse3* ™ Device Programmer and Adapters

## Features

- **OEM version of Data I/O ChipLab™**
- **Programs all Cypress PROMs, EPROMs, PLDs, and CPLDs**
- **Modular for easy device-specific support**
- **Easy to use Windows-based, PC interface**
- **New device support available with software changes on floppy disk, Cypress bulletin board, and Cypress web site**
- **DIP adapter included with base unit**
- **Mouse-driven user interface**
- **On-line documentation and device support list**
- **One-year warranty**
- **Dimensions of CY3500 are 25 x 25 x 7.6 cm or 9.75 x 9.75 x 3 in and the weight is 1.02 kg or 2.25 lbs.**

## Functional Description

*Impulse3*™ is Cypress's OEM version of the Data I/O ChipLab. It provides programming support for all of Cypress's programmable devices. The programmer uses a PC interface to provide an easily accessible programming environment. The PC's parallel port is used to communicate with the programmer, and device-specific adapters and drivers to ensure that you get the specific device support you need for your programming application.

CY3500 uses industry standard JEDEC and HEX (for PROMs) data format for programming and can be upgraded by Data I/O to support products from other vendors.

## System Requirements

The CY3500 works with your IBM compatible PC computer. The minimum system requirements are:

- **One free parallel port**
- **Minimum 2-MB extended memory**
- **Intel 386, 486, or Pentium processor**
- **DOS version 3.3 or higher, or Windows 3.1/95**
- **5 MB of free hard disk space for the programmer drivers and programs**
- **High Density floppy disk drive (3.5-inch)**
- **Microsoft-compatible mouse**

## Device Support

*Impulse3* is sold modularly and supports all Cypress programmable products. The base unit (CY3500) supports DIP devices up to 44 pins. For other device/package combinations, an adapter is required. In addition, devices over 44 pins require a high pin-count adapter (CY3501A).

## Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

| | |
|---|---|
| Input Voltage | 90 to 264 Vac, 48 to 63 Hz |
| Programmer Voltage | 24V (AC or DC) ±10% |
| Programmer Current | AC = 1.67Å, DC = 1.25Å |
| Operating Temperature | 0°C to 40°C |
| Storage Temperature | –40°C to 55°C |
| Relative Humidity (Operating) | 20% to 80% |
| Relative Humidity (Storage) | 10% to 90% |
| Operating Altitude | to 5,000 Meters |
| Storage Altitude | to 15,000 Meters |

## Ordering Information

| Part Number | Description |
|---|---|
| CY3500 | *Impulse3* base unit and DIP adapter for all DIP packaged devices |
| CY3501A | Adapter for high pin count devices including FLASH370i™ |
| CY3509 | 28-pin PLCC for 344 |
| CY3511 | 44-pin PLCC PPI for 343 |
| CY3512 | 44-pin PLCC PPI for 371, 372 |
| CY3514 | 68-pin PLCC PPI for 342 |
| CY3516 | 84-pin PLCC PPI for 341 |
| CY3517 | 84-pin PLCC PPI for 373, 374 |
| CY3519 | 160-pin TQFP PPI for 375 |
| CY3522 | 100-pin TQFP PPI for 373, 374 |
| CY3524 | 100-pin PGA PPI for 346 |
| CY3526 | 85-pin PGA PPI for 341 |
| CY3527 | 85-pin PGA PPI for 373, 374 |
| CY3529 | 69-pin PGA PPI for 342 |
| CY3535 | 100-pin TQFP PPI for 346 |
| CY3536 | 84-pin PLCC for 346 |
| CY3537 | 160-pin CPGA PPI for 375 |
| CY3541 | 160-pin CQFP T&F PPI for 375 |
| CY3542 | 160-pin CQFP MCR PPI for 375 |
| CY3544 | 20-pin QSOP PPI for 16V8 |
| CY3545 | 44-pin PLCC PPI for 276 |
| CY3546 | 44-pin TQFP PPI for 371 |
| CY3004A | 28-pin LCC adapter for 22V10 |
| CY3004B | 28-pin LCC adapter for PROM |
| CY3005 | 20-pin LCC adapter for PAL20, PALC20 |
| CY3006A | 28-pin PLCC adapter for 22V10 |
| CY3006B | 28-pin PLCC adapter for PROM |
| CY3007 | 20-pin PLCC adapter for PAL20, PALC20 |
| CY3008 | 28-pin LCC adapter for 251, 265, 269, 271, 331, 335 |
| CY3009 | 28-pin PLCC adapter for 251, 265, 269, 271, 331, 335 |
| CY3010 | 28-pin LCC adapter for 20G10, 20RA10 |
| CY3011 | 28-pin PLCC adapter for 20G10, 20RA10 |
| CY3014 | 24-pin SOIC adapter for all PROMs except 268 |
| CY3017 | 32-pin PLCC adapter for 251, 254 |
| CY3019 | 24-pin CerPack adapter for 245, 261, 263, 291 |
| CY3020 | 28-pin CerPack adapter for 251, 331, 271, 265 |
| CY3021 | 20-pin CerPack adapter for PAL20, PALC20 |
| CY3022 | 28-pin SOJ adapter for PROM |
| CY3024 | 32-pin LCC adapter for 256, 266, 271, 274, 277, 279, 286 |
| CY3027 | 32-pin LCC adapter for 285, 287 |
| CY3043 | 32-pin PLCC adapter for 27H010 |
| CY3044 | 32-pin PLCC adapter for 256, 266, 271, 274, 277, 279, 286 |
| CY3045 | 32-pin PLCC adapter for 285, 287 |

## Ordering Information (continued)

| Part Number | Description |
|---|---|
| CY3047 | 28-pin TSOP adapter for 256 |
| CY3048 | 32-pin TSOP adapter for 256 |
| CY3049 | 32-pin TSOP adapter for 27H010 |

Document #: 38–00374–B

CYPRESS

# CY3600

# InSRkit™: ISR™ Programming Kit

## Features

- **Supports all FLASH370i™ devices**
- **Standard JTAG programming Interface**
- **Multi-device programming**
- **Supports cascading of devices**
- **Easy to use PC based interface**
- **Eliminates programming insertion to improve manufacturing efficiency**
- **For programming in the lab, on manufacturing floor, or at remote sites**

## Kit Contents

- **ISR™ (In-System Reprogrammable) Programming Cable**
- **ISR Programming Software**
- **ISR User's Guide**
  - **PC Requirements**
  - **Configuration File Syntax**
  - **Description of all error codes**
- **ISR Application Notes**

## Functional Description

The ISR programming kit enables users to program FLASH370i CPLDs on board with our ISR Programming kit and a personal computer. The ISR programming cable connects to the parallel port of a PC into a standard 10-pin male connector mounted on the user's board.

The ISR software reads in standard JEDEC format programming files and converts them to a serial bit stream for ISR programming. The ISR software reads in a simple user defined configuration file that dictates how many devices are in the daisy chain and what operation is to be done on each FLASH370i device. The same chain can be used with other JTAG compliant devices.

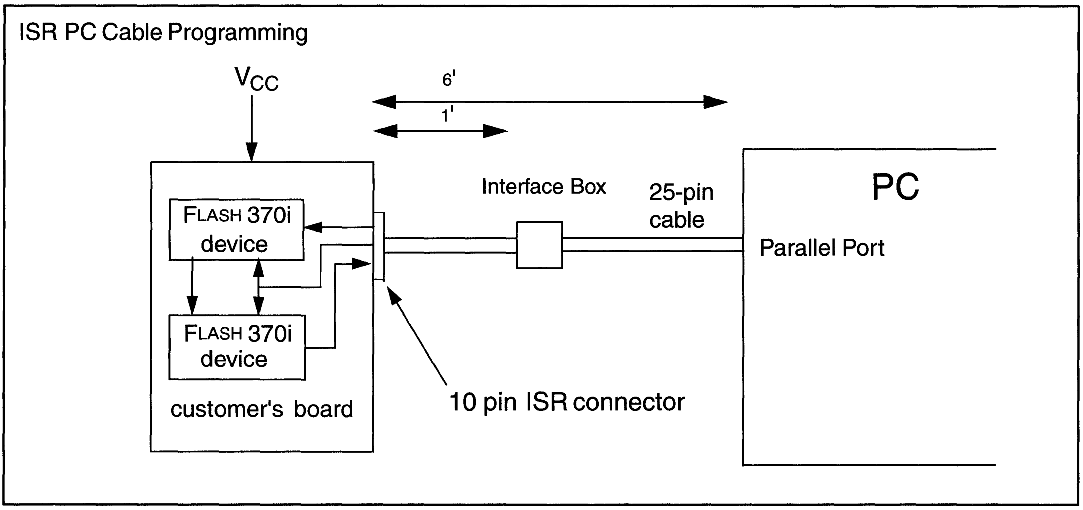
The ISR User's Guide describes the operation of the ISR software and how to set up the configuration file. The application notes included with the kit describe the FLASH370i architecture and all system design considerations for programming with the InSRkit.

### 10-pin ISR Connector

The diagram below shows the pinout of the 10-pin male connector to be mounted onto the board.

The view looking down onto the board is shown below.

| | | | | |
|---|---|---|---|---|
| SMODE | SCLK | SDI | NC | GND |
| GND | ISRVPP | ISR[1] | VCC | SDO |



**Note:**

[1] Refer to the ISR User's Guide for detailed information on ISR programming and electrical specification requirements.

*Table 1* describes the function of each of these pins on the ISR programming cable. An OUTPUT is provided by the PC and an INPUT is provided by the target system.

**Table 1. Pin Description**

| Pin | Type | Description |
|---|---|---|
| ISRVPP | OUTPUT | In System Programming high voltage programming pin. |
| ISR[1] | OUTPUT | In System Reprogramming enable indicator. |
| VCC | INPUT | +5V supply voltage provided from the target system to the cable. |
| SDO | INPUT | Serial Data Output Receiver. The SDO output pin of the last device in the ISR chain of the system is connected to this input pin. |
| SMODE | OUTPUT | Serial Mode Control. This is the mode select control input for the TAP controller state machine contained in the ISR interface. |
| SCLK | OUTPUT | Serial Clock. ISR interface clock input. |
| SDI | OUTPUT | Serial Data Input Driver. This output pin is connected to the SDI input of the first device in the ISR chain. |
| NC | NC | No Connect |
| GND | — | Zero volt common ground for PC and target system. |

The dimensions of the male connector required in the system are given below.

Male Connector:
2 x 5 = 2 rows, 5 pins per row
Measuring from center of the pins, each pin is 0.1” from the others.
Pin Length is 0.23”
Pin cross-section is 0.025” x 0.025”

Part Number: Amp 103309-1

## PC System Requirements

- One free parallel port
- 500K conventional memory or higher
- 486 PC or better
- DOS version 6.0 or higher
- 1MB or more disk space

## Ordering Information

| Product Code | Description |
|---|---|
| CY3600 | ISR Programming Kit |
| CY3601[2] | Detachable Ribbon Cable |

**Note:**
2. For replacement purposes only.

Document #: 38-00518

InSRkit, ISR, and FLASH370i are trademarks of Cypress Semiconductor Corporation.

fax id: 6260

CYPRESS

# CY3620

# *Warp2*ISR™ ISR Design Kit for CPLDs

## Features

- **Complete design and programming kit for In-System Reprogrammable (ISR™) CPLDs**
- **Industry-leading *Warp2*® design software**
- **Easy-to-use InSRkit™ PC programmer for on-board programming**
- **Supports all FLASH370i™ CPLDs**
- **Standard PC-based interface**

## Functional Description

*Warp2*ISR is a complete ISR CPLD design and programming solution. Included is *Warp2*, a state-of-the-art VHDL compiler for designing with Cypress FLASH370i devices, and an ISR PC programmer. (For a complete description of *Warp2* please see the CY3120 datasheet. For a complete description of the ISR PC programmer, please see the CY3600 datasheet.)

*Figure 1* shows the *Warp2*ISR kit in action with *Warp2* being used for design and InSRkit for programming. The ISR programmer connects from the PC parallel port directly to your board for quick design changes. Multiple devices can be programmed in series, and ISR devices are cascadable with other JTAG-compliant devices for convenient programming.

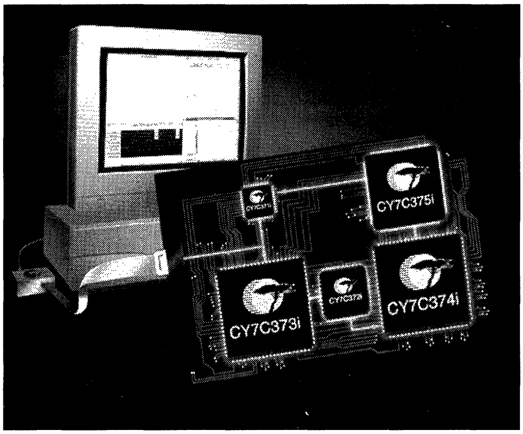


**Figure 1. Design and Program with *Warp2*ISR**

## PC System Requirements

**For *Warp2***

IBM PC or equivalent (486 or higher recommended)
16 Mbytes of RAM (32 Mbytes recommended)
100-Mbyte hard disk space
CD-ROM drive
Two- or three-button mouse
Windows 3.1x, Windows 95, or Windows NT

**For InSRkit**

IBM PC or equivalent (486 or higher recommended)
500K or more conventional memory
1-Mbyte or more hard disk space
One free parallel port
DOS version 6.0 or higher

## Ordering Information

| Product Code | Description |
|---|---|
| CY3620R42 | *Warp2*ISR ISR Design Kit for CPLDs |

*Warp2*ISR includes:

CD-ROM with *Warp2* and on-line documentation
ISR programmer for PC and software
ISR programmer User's Guide
FLASH370i™ samples
*VHDL for Programmable Logic* Textbook
Registration Card
Release Notes

Document #: 38-00592-A

*Warp2* is a registered trademark of Cypress Semiconductor Corporation.
ISR, InSRkit, *Warp2*ISR, and FLASH370i are all trademarks of Cypress Semiconductor Corporation.

CYPRESS

GENERAL INFORMATION 1

SMALL PLDS 2

CPLDS 3

DEVELOPMENT TOOLS 4

**APPLICATION NOTES** 5

QUALITY 6

PACKAGES 7

**Table of Contents** **Page Number**

## Application Notes

Are Your PLDs Metastable? ........ 5-1
Designing with the CY7C335 and *Warp2*® VHDL Compiler ........ 5-20
Getting Started Converting .ABL Files to VHDL ........ 5-48
Abel™-HDL vs. IEEE-1076 VHDL ........ 5-71
State Machine Design Considerations and Methodologies ........ 5-82
Efficient Arithmetic Designs Targeting FLASH370i™ CPLDs ........ 5-112
The FLASH370i™ Family Of CPLDs and Designing with *Warp2*® ........ 5-136
Implementing a Reframe Controller for the CY7B933 HOTLink™ Receiver in a CY7C371 CPLD ........ 5-154
Implementing a 128Kx32 Dual-Port RAM Using the FLASH370™ ........ 5-166
An Introduction to In System Reprogramming with FLASH370i™ ........ 5-177
Designing With FLASH370i™ for PC Cable Programming ........ 5-183
Using Hierarchy in VHDL Design ........ 5-195

# Are Your PLDs Metastable?

This application note provides a detailed description of the metastable behavior in PLDs from both circuit and statistical viewpoints. Additionally, the information on the metastable characteristics of Cypress PLDs presented here can help you achieve any desired degree of reliability.

Metastable is a Greek word meaning "in between." Metastability is an undesirable output condition of digital logic storage elements caused by marginal triggering. This marginal triggering is usually caused by violating the storage elements' minimum set-up and hold times.

In most logic families, metastability is seen as a voltage level in the area between a logic HIGH and a logic LOW. Although systems have been designed that did not account for metastability, its effects have taken their toll on many of those systems.

In most digital systems, marginal triggering of storage elements does not occur. These systems are designed as synchronous systems that meet or exceed their components' worst-case specifications. Totally synchronous design is not possible for systems that impose no fixed relationship between input signals and the local system clock. This includes systems with asynchronous bus arbitration, telecommunications equipment, and most I/O interfaces. For these systems to function properly, it is necessary to synchronize the incoming asynchronous signals with the local system clock before using them.

*Figure 1* shows a simple synchronizer, whose asynchronous input comes from outside the local system. The synchronizer operates with a system clock that is synchronous to the local system's operation. On each rising edge of this system clock, the synchronizer attempts to capture the state of the asynchronous input. *Figure 2* shows the expected result. Most of the time, this synchronizer performs as desired.

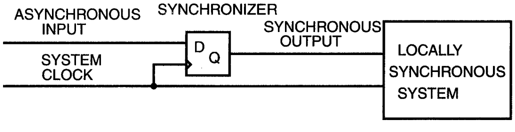


**Figure 1. Simple Synchronizer**

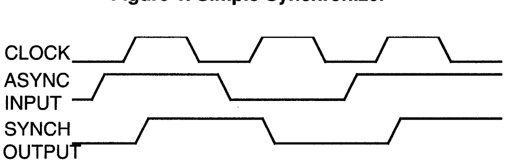


**Figure 2. Expected Synchronizer Output**

Digital systems are supposed to function properly all the time, however. But because there is no direct relationship between the asynchronous input and the system clock, at some point the two signals will both be in transition at very nearly the same instant. *Figure 3* shows some of the synchronizer's possible metastable outputs when this input condition occurs. These types of outputs would not occur if the synchronizer made a decision one way or the other in its specified clock-to-output time. A flip-flop, when not properly triggered, might not make a decision in this time. When improperly triggered into a metastable state, the output might later transition to a HIGH or a LOW or might oscillate.

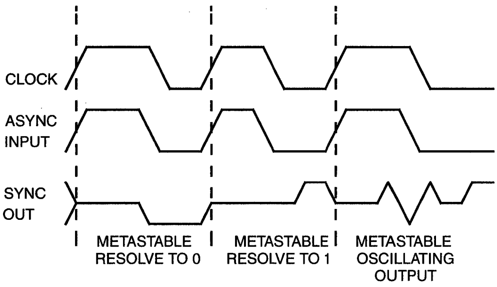


**Figure 3. Possible Metastable States of Synchronizer**

When other components in the local system sample the synchronizer's metastable output, they might also become metastable. A potentially worse problem can occur if two or more components sample the metastable signal and yield different results. This situation can easily corrupt data or cause a system failure.

Such system failures are not a new problem. In 1952, Lubkin (Reference 1) stated that system designers, including the designers of the ENIAC, knew about metastability. The accepted solution at that time was to concatenate an additional flip-flop after the original synchronizer stage (*Figure 4*). This added flip-flop does not totally remove the problem but does improve reliability. This same solution is still in wide use today.

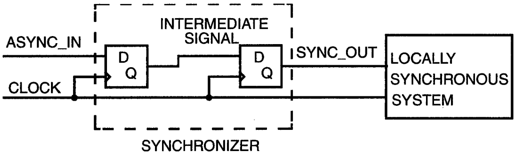


**Figure 4. Two-Stage Synchronizer**

Recovery from metastability is probabilistic. In the improved synchronizer, the first flip-flop's output might still be in a metastable state at the end of the sample clock period. Because the flip-flops are sequential, the probability of propagating a metastable condition from the second flip-flop stage is the square of the probability of the first flip-flop remaining metastable for its sample clock period. This type of synchronizer does have the drawback of adding one clock cycle of latency, which might be unacceptable in some systems.

As system speeds increase and as more systems utilize inputs from asynchronous external sources, metastability-induced failures become an increasingly significant portion of the total possible system failures. So far, no known method totally eliminates the possibility of metastability. However, while you cannot eliminate metastability, you can employ design techniques that make its probability relatively small compared with other failure modes.

## Explanation of Metastability

In a flip-flop, a metastable output is undefined or oscillates between HIGH and LOW for an indefinite time due to marginal triggering of the circuit. This anomalous flip-flop behavior results when data inputs violate the specified set-up and hold times with respect to the clock.

In the case of a D-type flip-flop, the data must be stable at the device's D input before the clock edge by a time known as the set-up time, $t_S$. This data must remain stable after the clock edge by a time known as the hold time, $t_h$ (*Figure 5*). The data signal must satisfy both the set-up and hold times to ensure that the storage device (register, flip-flop, latch) stores valid data and to ensure that the outputs present valid data after a maximum specified clock-to-output delay $t_{co\_max}$. As used in this application note, $t_{co\_max}$ refers to the interval from the clock's rising edge to the time the data is valid on the outputs. In most cases, $t_{co\_max}$ refers to the maximum $t_{co}$ specified by a data sheet, as opposed to the average or typical $t_{co}$ value.

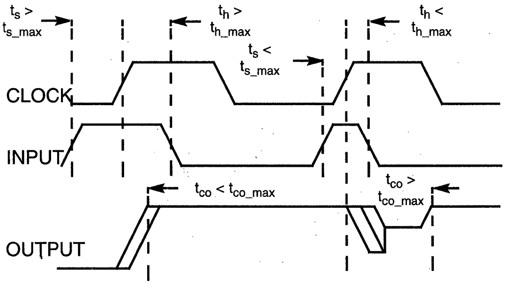


**Figure 5. Triggering Modes of a Simple Flip-Flop**

If the data violates either the set-up or hold specifications, the flip-flop output might go to an anomalous state for a time greater than $t_{co\_max}$ (*Figure 5*). The additional time it takes the outputs to reach a valid level can range from a few hundred picoseconds to tens of microseconds. The amount of additional time beyond $t_{co\_max}$ required for the outputs to reach a valid logic level is known as the metastable walk-out time. This walk-out time, while statistically predictable, is not deterministic.

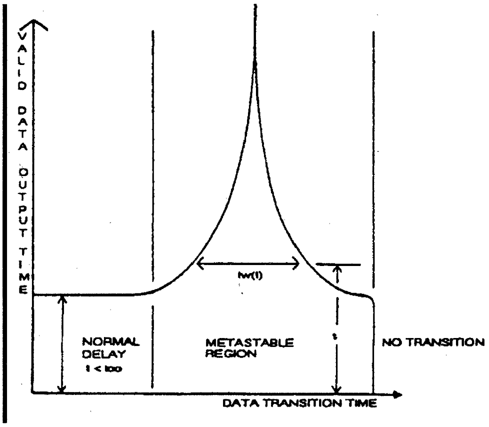


**Figure 6. Output Propagation vs. Data Transition**

*Figure 6,* from Reference 2, shows the variation in output delay with data input time. The left portion of the graph shows that when the data meets the required set-up time, the device has valid output after a predictable delay, which equals $t_{co}$. The middle portion of the graph indicates the metastable region. If the data transitions in this region, valid output is delayed beyond $t_{co\_max}$. The closer the input transitions to the center of the metastable region, violating the device's triggering requirements, the longer the propagation delay. If the data transitions after the metastable region, the device does not recognize the input at that clock edge, and no transition occurs at the output. As given in Reference, you can predict the region $t_w$, where data transitions cause a propagation delay longer than t, from the formula:

$$t_w = t_{co} e^{\frac{-(t - t_{co})}{\tau}} \qquad \text{Eq. 1}$$

where t depends on device-specific characteristics such as transistor dimensions and the flip-flop's gain-bandwidth product.

*Figure 7* shows another way of looking at metastability. A flip-flop, like any other bistable device, has two minimum-potential energy levels, separated by a maximum-energy potential. A bistable system has stability at either of the two minimum-energy points. The system can also have temporary stability—metastability—at the energy maximum. If nothing pushes the system from the maximum-energy point, the system remains at this point indefinitely.

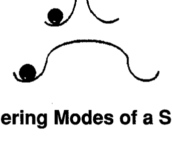


**Figure 7. Triggering Modes of a Simple Flip-Flop**

A hill with valleys on either side is another bistable system. A ball placed on top of the hill tends to roll toward one of the minimum-energy levels. If left undisturbed at the top, the ball can remain there for an indeterminate amount of time. As this figure indicates, the characteristics of the top of the hill as well as natural factors affect how long the ball stays there. The steepness of the hill is analogous to the gain-bandwidth product of the flip-flop's input stage.

## Causes of Metastability

Systems with separate entities, each running at different clock rates, are called globally asynchronous systems (Reference 4). The entities might include keyboards, communication devices, disk drives, and processors. A system containing such entities is asynchronous because signals between two or more entities do not share a fixed relationship.

Metastability can occur between two concurrently operating digital systems that lack a common time reference. For example, in a multiprocessing system, it is possible that a request for data from one system can occur at nearly the exact moment that this signal is sampled by another part of the system. In this case, the request might be undefined if it does not obey the set-up and hold time of the requested system.

When globally asynchronous systems communicate with each other, their signals must be synchronized. Arbitration must occur when two or more requests for a shared resource are received from asynchronous systems. An arbiter decides which of two events should be serviced first. A synchronizer, which is a type of arbiter with a clock as one of the arbited signals, must make its decision within a fixed amount of time. A device can synchronize an input signal from an external, asynchronous device in cases such as a keyboard input, an external interrupt, or a communication request.

Care must be taken when two locally synchronous systems communicate in a globally asynchronous environment. A synchronization failure occurs when one system samples a flip-flop in the other system that has an undefined or oscillating output. This event can distribute non-binary signals through a binary system (Reference 5).

In synchronizers, the circuit must decide the state of the data input at the clock input's rising edge. If these two signals arrive at the same time, the circuit can produce an output based on either decision, but must decide one way or the other within a fixed amount of time.

## Attacking Metastability

The design of synchronous systems is much different than the design of globally asynchronous systems. The design of a synchronous digital system is based on known maximum propagation delays of flip-flops and logical gates. Asynchronous systems by definition have no fixed relationship with each other, and therefore, any propagation delay from one locally synchronous system to the next has no physical meaning.

Two different methods are available to produce locally synchronous systems from globally asynchronous systems. The first method involves creating self-timed systems. In a self-timed system, the entity that performs a task also emits a signal that indicates the task's completion. This handshaking signal allows the use of the results when they are ready instead of waiting for the worst-case delay. Such handshaking signals allow communications between locally synchronous systems.

The advantage of the self-timed method is that it permits machines to run at the average speed instead of the worst-case speed. The disadvantages are that a self-timed system must have extra circuitry to compute its own completion signals and extra circuitry to check for the completion of any tasks assigned to external entities.

Petri Nets, data flow machines, and self-timed modules all use the self-timed method of communication among locally synchronous systems. Self-timed structures do not completely eliminate metastability, however, because they can include arbiters that can be metastable. Most systems do not include self-timed interfaces due to the additional circuitry and complexity.

The second method of producing locally synchronous systems from globally asynchronous systems is the simple synchronizer. This is the most common way of communicating between asynchronous objects. The metastability errors that might arise from these systems must be made to play an insignificant role when compared with other causes of system failure.

Many metastability solutions involve special circuits (References 6 and 7). Some of these solutions do not reduce metastability at all (References 13 and 8). Others, however, do reduce metastability errors by pushing the occurrence of metastability to a place where sufficient time is available for resolving the error. Most of these circuits are system dependent and do not offer a universal solution to metastability errors.

The easiest and the most widely used solution is to give the synchronizing circuit enough time to both synchronize the signal and resolve any possible metastable event before other parts of the system sample the synchronized output. This solution requires knowledge of the metastable characteristics of the device performing the synchronization.

Many semiconductor companies have developed circuits such as arbiters, flip-flops, and latches that are specifically designed to reduce the occurrence of metastability. Although these parts might have good metastability characteristics, they have very limited application. The circuits can only function as flip-flops or arbiters and do not have the flexibility of PLDs. Cypress Semiconductor has designed the flip-flops in the company's PLDs to be metastability hardened. This allows you to use Cypress PLDs in a wide range of systems requiring synchronization.

## Circuit Analysis of Metastability

Many authors have written papers detailing the analysis of metastability from a circuit standpoint (References 5, 7, 8, 9, 10, 11, and 12). In Reference 11, for example, Kacprzak presents a detailed analysis of an RS flip-flop's metastable operation. He states that a flip-flop has two stages of metastable operation (*Figure 8*).

During the initialization phase, the Q and $\overline{Q}$ outputs move simultaneously from their existing levels to the metastable voltage $V_m$, which is the voltage at which $V_q = \overline{V}_q$.

The second or resolving phase occurs when the outputs once again drift toward stable voltages. Once a flip-flop has entered a metastable state, the device can stay there for an indeterminate length of time. The probability that the flip-flop will stay

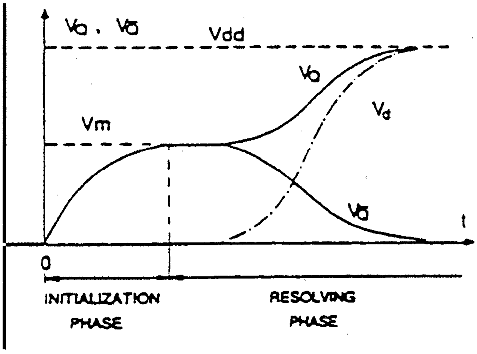


**Figure 8. Two Phases of Metastability**

metastable for an unusually long period of time is zero, however, due to factors such as noise, temperature imbalance within the chip, transistor differences, and variance in input timing. During the second phase of metastability, for very small deviations around the metastable voltage, $V_m$, the flip-flop behaves like two cross-coupled linear amplifier stages that gain $V_d = V_q - \overline{V_q}$. When the gain of the cross-coupled loop exceeds unity, the differential voltage increases exponentially with time.

The length of time the flip-flop takes to resolve cannot be exactly determined. The probability that the flip-flop will resolve within a specific length of time, however, can be predicted. This probability depends on the electrical parameters of the flip-flop acting as a linear amplifier around the metastability voltage. The solution (Reference 11) to the differential voltage $V_d(t)$ driving the resolving phase is given by

$$V_d(t) = V_d(t_0)e^{\frac{(t-t_0)}{\tau}} \qquad \text{Eq. 2}$$

where t depends directly on the amplifier gain and capacitance, and where $V_d(t_0)$ represents the differential voltage at some time $t_0$. You can use this equation to determine the length of time that the output voltage will take to drift from the metastable voltage $V_m$ to a specified voltage difference $V_d$.

Horstmann (Reference 5) states that a flip-flop, like any other system with two stable states, can be described by an energy function with two local energy minima where P(x) = 0 (*Figure 9*). Any bistable system has at least one metastable state, which is an unstable energy level within the system and represents the local maximum of the energy function. The system's gradient can be represented by a force, F(x), that is zero at stable and metastable states (inflection points of the energy function).

*Figure 10* shows a simplified first-order model of an RS flip-flop used to predict and visualize metastability. A flip-flop energy transfer curve (*Figure 11*) shows the relationship between the two outputs. The two stable states are local energy minima of the system. The metastable state, M, is a local energy maximum and represents an unstable state with loop gain near M that is greater than one.

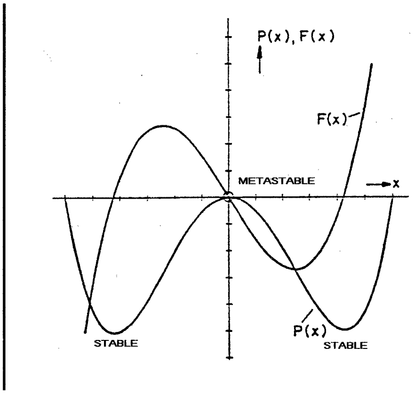


**Figure 9. Energy/Force Function of a Bistable System**

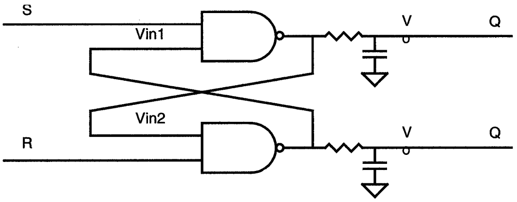


**Figure 10. First-Order Flip-Flop**

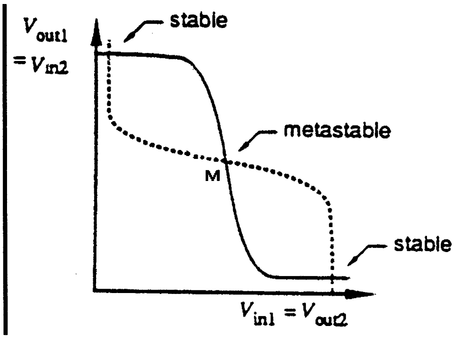


**Figure 11. Energy Transfer Diagram of Simple RS Flip-Flop**

*Figure 12* shows the trigger line for the first-order approximation of the flip-flop. The dashed line RS represents the device's normal trigger line, which does not follow the transfer curve because, during triggering, the feedback loop has not been established. If at varying points along the trigger line the feedback loop is re-established, the nodes of the device follow the curves that lead to the line $S_0 - S_1$. Once on this line, the circuit exponentially drifts toward stability at either $S_0$ or $S_1$, depending on which side of the line $Q = \overline{Q}$ the feedback loop

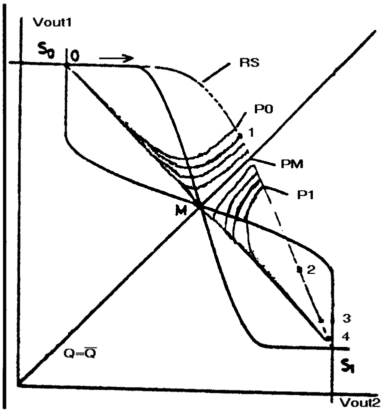


**Figure 12. Energy Transfer Curves showing Trigger Paths**

was re-established. The curves are solutions to the first-order model circuit equations for the device shown in *Figure 10.*

When the feedback loop is restored near the line Q = Q̄, the system moves toward the unstable state M and can take an indefinite amount of time to exit from this metastable state. You can see this from the graph by noticing that $S_0$ and $S_1$ are equally likely solutions for system stability from M. Once the feedback loop is re-established, the system exponentially decays toward M and then exponentially grows toward $S_0$ or $S_1$.

*Figure 13* shows the system's possible trigger events using the implied time scale of the state-space curves. The solution of these simplified first-order equations indicates that the fastest metastable resolution time occurs when the circuit's gain-bandwidth product is maximized.

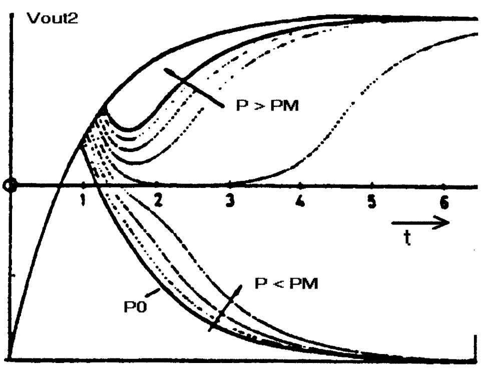


**Figure 13. Time Scale Showing Trigger Paths**

Flannagan (Reference 12), in an attempt to maximize the gain-bandwidth product, solves simplified flip-flop equations to determine the phase trajectory near the metastable point. His results, which are supported by other authors, indicate that p and n devices with equal geometries produce the optimal gain-bandwidth product for metastable event resolution.

## Statistical Analysis of Metastability

To begin the analysis of metastability, assume that the flip-flop's probability of resolving its metastable state does not depend on its previous metastable state. In other words, the metastable device has no memory of how long it has been in a metastable region. The analysis of metastability also assumes that the flip-flop's probability of resolving its metastable state in a given time interval does not depend on the metastable resolution in another disjoint time interval. The probability that a metastable event will resolve in a given interval (0,t) is only proportional to the length of the interval.

These assumptions yield an exponential distribution that describes the probability that the flip-flop resolves its metastability at a time t. The exponential distribution has the form

$$f(x) = \mu e^{-\mu t} \qquad \text{Eq. 3}$$

where m is the expected value of metastability resolution per unit time (settling rate).

Using this equation and given that the flip-flop was metastable at time t = 0, the probability of a metastable event lasting a time t or longer is

$$P(met_t | met_{t=0}) = \int_t^{\infty} \mu e^{-\mu t} dT = e^{-\mu t} \qquad \text{Eq. 4}$$

The next part of the analysis involves the probability that the flip-flop is metastable at time t = 0. This part of the analysis assumes that the probability that the data transitions in a given time interval depends only on the length of the interval. A Poisson process with rate $f_d$ describes the probability of the data transitioning at a time t:

$$p(x) = \frac{e^{-f_d t}(f_d t)^x}{x!} \qquad \text{Eq. 5}$$

where *x* is the number of transitions.

If a data transition within a bounded time interval, W, of the clock edge causes a metastable condition, the expected number of transitions of this Poisson process with rate $f_d$ in time interval W is

$$E(X) = \sum_{x=0}^{\infty} \frac{x e^{-f_d W}(f_d W)^x}{x!} = f_d W \qquad \text{Eq. 6}$$

Because this expected number of transitions is the same as the probability that the flip-flop is metastable at t = 0, the equation for the probability at t = 0 is

$$P(met_{t=0}) = f_d W \qquad \text{Eq. 7}$$

Using *Equations 5* and 7, the probability that a given clock cycle results in metastability that lasts at most a time t is

$$\begin{aligned} P(met_t) &= P(met_t | met_{t=0}) P(met_{t=0}) \\ &= f_d W e^{-\mu t} \end{aligned} \qquad \text{Eq. 8}$$

Substituting $\frac{1}{t_{sw}}$ for μ allows this variable to be expressed as a settling time constant of the flip-flop. Further, a synchronization failure for a given clock cycle exists whenever a metastable event lasts a specified time ($t_r$) or longer. Using these two substitutions, the probability that the flip-flop is metastable in a given clock cycle is

$$P(fail_{1\ clock}) = f_d W e^{\frac{-t_r}{t_{sw}}} \qquad \text{Eq. 9}$$

Because the data transitions are independent, the number of failures in n clock cycles has a binomial distribution with an expected number of failures:

$$E(fail_{n\ cycles}) = nP(fail_{1\ cycle}) \qquad \text{Eq. 10}$$

Assuming a sample clock frequency, $f_c$, that represents the number of clock cycles, n, per unit time, the expected number of failures per unit time is

$$E(fail_{unit\ time}) = f_c f_d W e^{\frac{-t_r}{t_{sw}}} \qquad \text{Eq. 11}$$

Assuming that all data transitions are independent and that the clock has a fixed period, the mean time between failures (MTBF) is

$$MTBF = \frac{1}{E(fail_{unit\ time})} = \frac{e^{\frac{t_r}{t_{sw}}}}{f_c f_d W} \qquad \text{Eq. 12}$$

where MTBF is a measure of how often, on the average, a metastable event lasts a time $t_r$ or longer.

## Metastability Data

The strong resemblance between *Equation 12* and *Equation* 2 is based on the predictions of the first-order circuit analysis of an RS flip-flop. In fact, the metastability resolving time constant, $t_{sw}$, is directly related to the variable t, which is based on the flip-flop's gain-bandwidth product.

The device-dependent variable W depends mostly on the window of time within which the combination of the input and clock generate a metastable condition. This parameter also depends on process, temperature, and voltage levels. The MTBF equation is usually plotted with $t_r$ (the resolving time allowed for metastable events) on the X axis and the natural log of the MTBF plotted on the Y axis (see the appendix in this note). Because the metastability equation is plotted on a semi-log scale, the graph of $t_r$ vs ln(MTBF) is a line described by the equation

$$\ln(MTBF) = \frac{t_r}{t_{sw}} - \ln(f_c f_d W) \qquad \text{Eq. 13}$$

Graphically, the parameter $t_{sw}$ is 1/slope of the line on this graph. The equation for $t_{sw}$ from the graph is

$$t_{sw} = \frac{t_{r1} - t_{r2}}{\ln(MTBF_1) - \ln(\mathrm{MTBF}_2)} \qquad \text{Eq. 14}$$

To determine how often, on the average, a given synchronizer in a system will go metastable (MTBF), you must know the two device-specific parameters W and $t_{sw}$, which should be available from the manufacturer. *Table 1*, discussed later in this note, lists these values for Cypress PLDs. Additional values you need are the average frequency of both the system data and the synchronizer clock and the amount of time after the synchronizer's maximum clock-to-Q time that is allowed to resolve metastable events.

For example, consider the method for determining the MTBF for a Cypress PALC22V10 registered PLD used as a synchronizer in a system with the following characteristics:

| | |
|---|---|
| W | = 0.125 ps |
| $t_{sw}$ | = 190 ps |
| $f_c$ | = system clock frequency = 25 MHz |
| $f_d$ | = average asynchronous data frequency<br>= 10 MHz |

In addition to these values, the PLD's maximum operating frequency, $f_{max}$, is taken directly from the data sheet. The frequency is specified as the internal feedback maximum operating frequency. It is calculated as

$$f_{max} = \frac{1}{t_{cf} + t_s} = 41.6MHz$$

where $t_{cf}$ is the clock-to-feedback time. If the data sheet does not specify $t_{cf}$, you can use $t_{co}$ as $t_{cf}$'s upper bound.

Using $f_{max}$, you calculate the amount of time that a metastable event is allowed to resolve, $t_r$, with

$$t_r = \frac{1}{f_c} - \frac{1}{f_{max}} = \frac{1}{25MHz} - \frac{1}{41.6MHz} = 16ns$$

Now you enter these values into the MTBF equation, making sure to keep all units in seconds:

$$MTBF = \frac{e^{\frac{t_r}{t_{sw}}}}{f_c f_d W}$$

$$= \frac{e^{\frac{16 \times 10^{-9} s}{190 \times 10^{-12} s}}}{25 \times 10^6 s^{-1} \times 20 \times 10^6 s^{-1} \times 0.125 \times 10^{-12} s}$$

$$= 59.7 \times 10^{33} s$$

$$= 1.89 \times 10^{27}\ years = Almost\ forever$$

If the operating frequency of the system, $f_c$, is simply changed to 33.3 MHz,

$$MTBF = \frac{e^{\frac{6 \times 10^{-9} s}{190 \times 10^{-12} s}}}{33.3 \times 10^6 s^{-1} \times 20 \times 10^6 s^{-1} \times 0.125 \times 10^{-12} s}$$

$$= 623 \times 10^{27}\ s$$

the system fails, on the average, about every 19,700 years—still beyond the system's normal lifetime.

And if $f_c$ is changed to $f_{max}$ (41.6 MHz),

$$MTBF = \frac{e^{\frac{0 \times 10^{-9} s}{190 \times 10^{-12} s}}}{41.6 \times 10^{6} s^{-1} \times 20 \times 10^{6} s^{-1} \times 0.125 \times 10^{-12} s}$$

the system fails, on the average, every 9.62 ms.

A 16-ns difference in resolve time, $t_r$, results in almost 36 orders of magnitude difference in MTBF. Obviously, accurate data is needed to design a system with a high degree of reliability without being overly cautious.

## Characterization of Metastability

Many authors (References 6, 8, 9, 10, 11, and 12) have performed numerous experiments on circuits to predict the likelihood of device metastability. These researchers have used several testing theories and apparatus that can be classified into three basic types (Reference 14).

Intermediate voltage sensors constitute the first type. Two voltage comparators determine whether the output voltage, Q, lies between two given voltages. The fixture produces an error output if Q has a level that is neither HIGH nor LOW, hence metastable. *Figure 14* shows an intermediate voltage sensor.

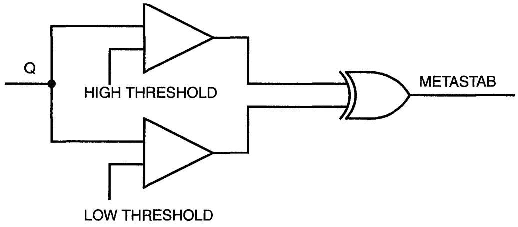


**Figure 14. Intermediate Voltage Sensor**

The second type of apparatus uses an output proximity sensor to determine if the Q and $\overline{Q}$ outputs have approximately the same voltages, which would indicate that the device is metastable. *Figure 15* shows an output proximity sensor.

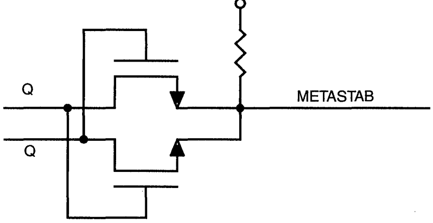


**Figure 15. Output Proximity**

The last type of apparatus uses a late-transition sensor to test for metastability. Note that if one or more gates separate the sensor from the metastable signal, the metastability might not be detected. The test circuitry must infer the occurrence of metastability by some other means. *Figure 16* shows an example of a late-transition sensor. The sample input is detected at time $t_1$, then at a later time $t_2$. If these two signals disagree, the device under test was metastable at $t_1$.

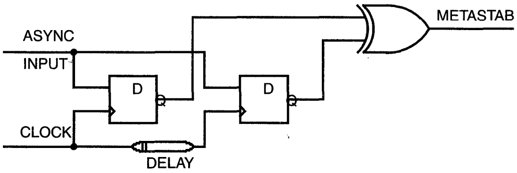


**Figure 16. Late-Transition Sensor**

## Information from Manufacturers

Many semiconductor companies provide metastability data on their parts. However, most companies do not present the data in a format the engineer can use. They either present inconclusive and incomplete data or they assume the engineer can use the data without further explanation. Few companies compare their devices with similar devices.

PLD manufacturers provide little data largely because of a fear that telling the design community that devices can fail in synchronizing applications will cause designers to use a competitor's parts. The truth is that no company can provide a device that is guaranteed never to become metastable when used as a synchronizer. At a given operating frequency, with a given asynchronous input, and given enough time, the device becomes metastable.

Cypress provides you with data you can use to build a system to any given level of reliability when using Cypress PLDs. Cypress has performed numerous tests and collected extensive data on Cypress PLDs, as well as PLDs from other companies. This data gives you a perspective of the parts that are best suited for a specific application. Specific data on the metastability characteristics of Cypress PLDs is found in this application note in the Test Results section.

## The Test Circuit

Cypress uses a test that falls into the category of the late-transition detection. Directly measuring the outputs of the flip-flop in a PLD are impossible due to the additional circuitry that lies between the flip-flop and the outside world. The metastability detection circuitry must, instead, infer the flip-flop's state.

*Figure 17* shows the metastability test circuit implemented in each test PLD. This circuit allows the PLD under test to effectively test itself. The device under test will both produce and record metastable conditions.

*Figure 18* is a state diagram showing the operation of the device. During normal operation, the two flip-flops' outputs ($F_1$, $F_2$) transition between states $S_1$ and $S_2$, depending on the synchronizer's state. During normal operation, the Exclusive-OR on these outputs produces a HIGH. This indicates either that metastability has not occurred within the device or that metastability that has occurred has resolved before the next clock cycle.

If a metastable event cannot resolve before the next clock cycle, the state machine move to states $S_3$ or $S_4$. In this case,

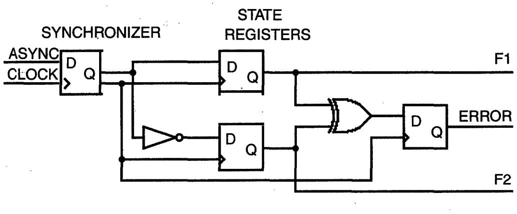


**Figure 17. Metastability Test Circuit**

the state flip-flops have interpreted the signal from the synchronization register differently; exclusive-ORing this signal produces a LOW at the device's output, indicating that unresolved metastability has occurred.

This test circuit does not catch all metastable events. Specifically, it does not record metastable events that resolve before the next clock cycle. But metastability causes an error only when it has not resolved by the time the signal is needed. The Cypress tests thus reveal the information designers need to know: how often metastability creates an error in the system.

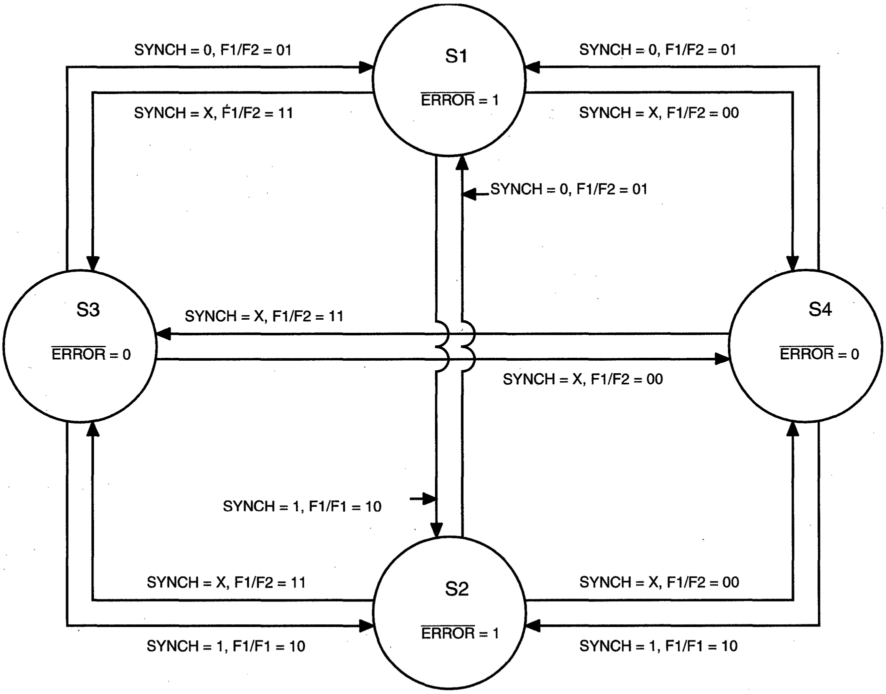


**Figure 18. Metastability Testing State Diagram**

The test circuit also includes the ability to check the maximum operating frequency of the device under test (*Figure 19*). At each clock edge, the first register's output toggles. When the device reaches its maximum operating frequency, the PLD array cannot resolve the changing signal fast enough to produce a valid output. At this speed, one register might resolve the signal correctly and one might not, or both might produce invalid signal resolutions. In any case, when Exclusive-ORing the state $T_1/T_2$ of the two maximum-frequency testing registers results in anything other than a HIGH, the part's maximum operating frequency is exceeded.

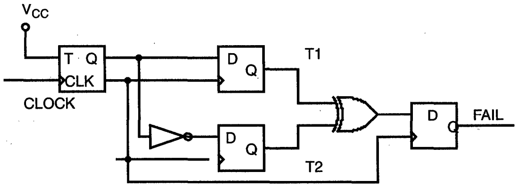


**Figure 19. Maximum Operating Frequency Test**

### The Test Board

A four-layer printed circuit board with two signal planes, a ground plane, and a power plane is used to perform the metastability measurements. Using this four-layer board gives a quiet testing environment with reliable, repeatable results. *Figure 20* shows a block diagram of the test board, with the complete schematic shown in *Figure 21*. The device under test (DUT) is decoupled with 0.01-mF and 100-pF capacitors. The test circuit is designed to fit all industry-standard and Cypress-proprietary PLDs. The socket allows DUT pins 1, 2, and 4 to serve as clock pins. Pin 3 is the device's asynchronous input. The $\overline{\text{ERROR}}$ condition is located on pin 27 of a 28-pin device, and the $\overline{\text{FAIL}}$ condition is on pin 20. Two additional outputs, $F_1$ and $F_2$, monitor the state of the metastability test circuit flip-flops.

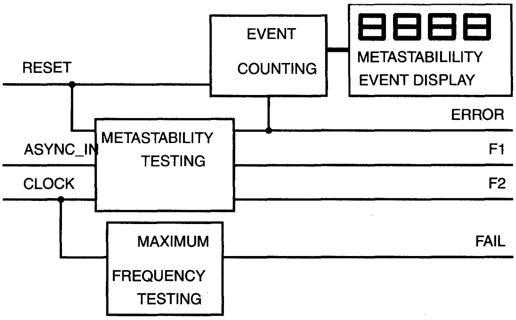


**Figure 20. Metastability Test Board Block Diagram**

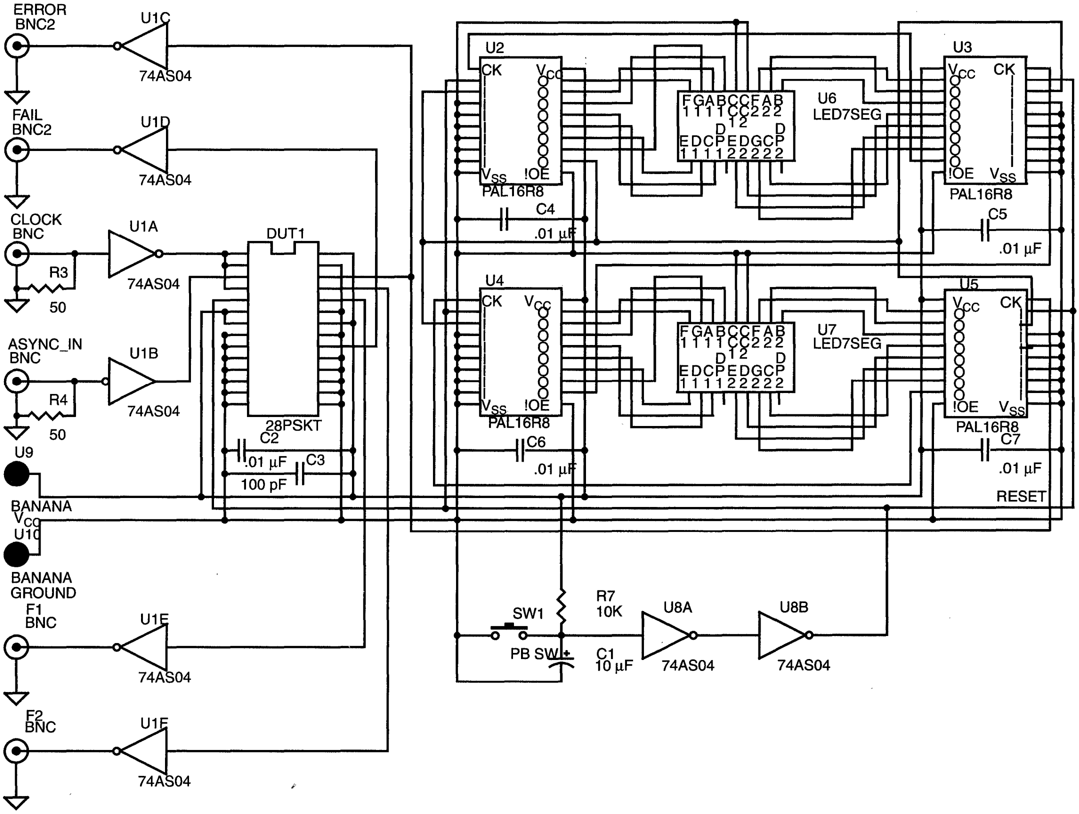


**Figure 21. Metastability Test Board Schematic**

All inputs and outputs connect with BNC connectors located around the board. The clock line, which is terminated with a 50Ω resistor to match the coax input impedance, is buffered with a 74AS04 and isolated from other signals by a ground trace. The input line is also terminated with a 50Ω resistor and buffered with a 74AS04. Four PLDs drive a four-digit LED display that counts metastability occurrences.

After going LOW in response to a metastable event, the ERROR signal automatically transitions HIGH again at the next system clock. This LOW-to-HIGH pulse produces a clock to the input of the first PLD, which in turn increments the display of metastable events. When a digit reaches 9, the next occurrence of metastability generates a cascade signal to the next higher digit.

In this way, the test board can record a maximum of 9,999 metastable events. If a metastable event is received at 9,999, all LEDs switch to E, indicating that an overflow condition occurred. A reset button resets all counters and initializes the DUT.

### Test Set-Up

*Figure 22* shows a block diagram of the test set-up used for metastability testing. Two independent pulse generators (Hewlett-Packard 8082As) produce the CLOCK and the ASYNC_IN signal to the test board. A Tektronix DAS9200 logic analyzer records metastable events. A 2465 CTS digital oscilloscope with frequency counter accurately determines the DUT's maximum operating frequency and the ASYNC_IN and CLOCK frequencies.

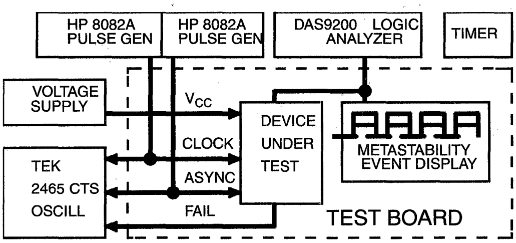


**Figure 22. Metastability Test Set-Up**

### Test Procedure

Cypress has tested all its 20-, 24-, and 28-pin PLDs. The fastest speed grades of each device type were tested because these devices have the best metastable resolution time and thus make the best synchronizers. Several parts from each device type were tested to ensure an average metastability characteristic for that product. Where possible, parts from different date codes were selected to eliminate variations among different wafer lots.

Testing for a specific device starts by creating the high-level description written in VHDL to be used with the *Warp2* VHDL Compiler. *Figure 23* lists the behavioral description used for generating a JEDEC file. All devices were programmed using JEDEC files generated by *Warp2*.

```
package test is
  component metastability port (
    clock, async_in, reset : in bit;
      fail, perror, f1, f2 : out bit);
  end component;
end test;

entity metastability is port (
  clock, async_in, reset : in bit;
    fail, perror, f1, f2 : out bit);
end metastability;

use work.bv_math.all

architecture fsm of metastability is

signal  sync  : bit;
signal  tsync : bit;
signal  t1,t2 : bit;
signal  f1_tmp, f2_tmp : bit;
signal  error_tmp : bit;
signal  fail_tmp : bit;

begin
  proc1: process begin

    wait until clock = '1';
    sync <= async_in;

  end process;
```

**Figure 23. *Warp2* VDHL Behavorial Description for Metastability Testing**

```
   proc2: process begin

      wait until clock = '1';
      f1_tmp <= sync;
      f2_tmp <= inv(sync);

   end process;

   proc3: process begin

      wait until clock = '1';
      error_tmp <= ((((inv(reset) and f1_tmp) and inv(f2_tmp))
         or ((inv(reset) and inv(f1_tmp)) and f2_tmp))
         or (reset and inv(error_tmp)));

   end process;

   proc4: process begin

      wait until clock = '1';
      if (tsync = '1') then
         tsync <= '0';
      else
         tsync <= '1';
      end if;

   end process;

   proc5: process begin

      wait until clock = '1';
      t1 <= tsync;
      t2 <= inv(tsync);

   end process;

   proc6: process begin

      wait until clock = '1' ;
      fail_tmp <= (t1 xor t2);

   end process;

fail <= inv(fail_tmp);
perror <= inv(error_tmp);
f1 <= inv(f1_tmp);
f2 <= inv(f2_tmp);

end fsm;
```

**Figure 23. *Warp2* VDHL Behavorial Description for Metastability Testing** (continued)

Each part is programmed, then tested for its maximum operating frequency, $f_{max}$. By attaching the $\overline{\text{FAIL}}$ output to the oscilloscope and observing the clock frequency at which the device started to malfunction ($\overline{\text{FAIL}}$ going LOW periodically), the maximum operating frequency for that part is determined. $f_{max}$ indicates the maximum rate at which metastability measurements can be taken with accurate results. Above this frequency, metastable events are indistinguishable from errors caused by exceeding $f_{max}$.

To determine each device's metastability characteristics, measurements are taken of the number of metastable events that occurred in a given time interval for several different clock and data frequencies.

*Equation 13* can be used to describe the graph of the metastability characteristics of the device:

$$\ln(MTBF) = \frac{t_r}{t_{sw}} - \ln(f_c f_c W)$$

The slope of the line, $t_{sw}$, can be determined only by forcing the Y intercept of the graph ($\ln(f_c f_d W)$) to a constant value when using *Equation 14*:

$$t_{sw} = \frac{t_{r1} - t_{r2}}{\ln(MTBF_1) - \ln(MTBF_2)}$$

Note that $t_{sw}$ is a constant, device-specific parameter.

Because W is also a constant, device-specific parameter, it is only necessary to hold the product $f_cf_d$ constant to make $\ln(f_cf_dW)$ constant. The independent variable $t_r$ is varied by changing $f_c$ to produce changes in the dependent variable ln(MTBF). Decreasing the frequency $f_c$ from its $f_{max}$ value increases the metastable resolution time, $t_r$, and decreases the probability that a metastable event will last longer than $t_r$.

As $f_c$ is decreased below a certain limit, the MTBF becomes too large to measure accurately. A metastable event occurring every minute is chosen as the upper limit for MTBF measurements. The range of clock rates for metastability testing is then between $f_{max}$ and the metastable-event-per-minute clock rate. Between these two rates, a selected frequency constant ($f_cf_d$) ensures that no point in this range has a clock frequency less than twice the data frequency. This is because a data signal that transitions more than once per clock period cannot be effectively sampled.

After determining this constant, data is taken from several test points within the test range by varying $f_c$ and $f_d$. The data at each test point is averaged among all test devices, and the equation for the line through these points is determined using a linear regression analysis. The correlation between the line and the data points verifies that the metastability equation accurately describes the test data. From the calculated results, the constants W and $t_{sw}$ are extracted.

### Test Results

*Table 1* and the Appendix list the results of the metastability analysis of Cypress PLDs. *Table 1* also lists the maximum data book operating frequency, $f_{max}$; the metastability equation constants, W and $t_{sw}$; the metastability resolve time, $t_r$, required for a 10-year MTBF; and the process for that part.

You can use this data to determine the maximum metastability resolve time ($t_r$) that you must use in a system to yield a given degree of reliability. The graphs and constants (W and $t_{sw}$) can be used with any speed grade of the device, but it is suggested that the fastest speed grade of the specific PLD be used for optimum synchronizer performance. These graphs indicate the time ($t_r$) and the device's minimum clock period that must be used to produce a desired degree of reliability.

For example, to determine the operating parameters of the Cypress PALC22V10-20 from *Table 1* when using the device as a synchronizer, determine the desired MTBF. With a 10-yr ($315 \times 10^6$s) MTBF, for instance, a synchronization failure will occur once every 10 years on the average. The maximum operating frequency ($f_{max}$) from the PALC22V10's data sheet is 41.6 MHz. From this information, you can determine the minimum time ($t_r$) beyond the device's minimum operating period that must be added for metastability resolution:

$$MTBF = \frac{e^{\frac{t_r}{t_{sw}}}}{f_cf_dW}$$

$$t_r = t_{sw}(\ln(MTBF) + \ln(f_cf_dW))$$

$$t_r = (0.190\times10^{-9}s)[\ln(315\times10^6 s)$$
$$+ \ln(41.6\times10^6\times41.6\times10^6\times0.125\times10^{-12})]$$
$$= 4.73\text{ ns}$$

This analysis assumes that the clock, $f_c$, operates at $f_{max}$ (41.6 MHz) and that the average asynchronous data frequency is no more than half the clock frequency. The latter condition ensures effective data sampling by the synchronizer. $f_d$, as explained in the Statistical Analysis of Metastability section represents the rate at which the data changes state. $f_d$ is twice the average frequency of the asynchronous data input because, during any given asynchronous data period, the asynchronous data changes state twice: once from LOW to HIGH and again from HIGH to LOW. Because either of these state changes can cause a metastable event, $f_d$ must be set to twice the average asynchronous data frequency when determining the worst-case MTBF.

Due to the real-world uncertainty in factors such as trace delays and the skew in clock generators, 5 ns is used instead of 4.73 ns for $t_r$. The synchronizer's maximum operating frequency, $f_c$, in this system is then

$$f_c = \frac{1}{t_s + t_{cf} + t_r} = \frac{1}{10ns + 12ns + 5ns} = 37.0MHz$$

The effective MTBF using these new values for $t_r$ and $f_c$ is

$$MTBF = \frac{e^{\frac{5\times10^{-9}s}{0.190\times10^{-9}s}}}{37.0\times10^6s^{-1}\times37.0\times10^6s^{-1}\times0.125\times10^{-12}s}$$
$$= 1.57\times10^9 = 49.7\text{ years}$$

**Table 1.Metastability Characteristics of Cypress PLDs.**

| Device | $f_{max}$ (MHz) | W (s) | $t_{sw}$ (s) | $t_r$ for 10-yr MTBF (ns) |
|---|---|---|---|---|
| PALC16R8-25 | 28.5 | 9.503E–12 | 0.515E–9 | 14.68 |
| PLDC20G10-20 | 41.6 | 3.730E–12 | 0.173E–9 | 4.91 |
| PALC20RA10-15 | 33.3 | 2.860E–12 | 0.216E–9 | 5.87 |
| PALCE22V10-7 | 100 | 32.35E–12 | 0.347E–9 | 10.56 |
| PALC22V10B-15 | 50.0 | 55.76E–12 | 0.261E–9 | 8.19 |
| PALC22V10-20 | 41.6 | 0.125E–12 | 0.190E–9 | 4.73 |
| CY7C331-20 | 31.2 | 0.298E–9 | 0.184E–9 | 5.91 |
| CY7C335-100 | 58.8 | 0.288E–12 | 0.189E–9 | 4.95 |
| CY7C344-20 | 41.6 | 0.966E–9 | 0.223E–9 | 7.55 |

Another example focuses on the CY7C330-50 used as a synchronizer in a system whose output registers are clocked at an $f_c$ of 35.7 MHz, and the data has an average frequency of 10 MHz. The MTBF for this device used as a synchronizer is calculated by first determining the metastable resolution time, $t_r$, allowed for synchronization. The maximum operating frequency of the part is specified in Cypress's *Data Book* as

$$f_{max} = \frac{1}{t_{co} + t_{is}}$$

where $t_{co}$ in this case specifies the clock-to-feedback delay, and $t_s$ specifies the set-up time of the output registers. $t_r$ is calculated with the equation:

$$t_r = \frac{1}{f_c} - \frac{1}{f_{max}} = \frac{1}{35.7MHz} - \frac{1}{50.0MHz} = 8 \text{ ns}$$

With this result, the MTBF is

$$MTBF = \frac{\dfrac{8 \times 10^{-9} s}{e^{0.290 \times 10^{-9} s}}}{35.7 \times 10^{6} s^{-1} \times 20.0 \times 10^{6} s^{-1} \times 1.02 \times 10^{-12} s}$$

$$= 1.31 \times 10^{9} = 41.6 \; years$$

This equation uses the same values for W and $t_{sw}$ with this 50-MHz device as with the 66-MHz device listed in *Table 1*. As stated previously, the constants listed in *Table 1* are valid for all speed grades of a specific device. Also note that the 10-MHz average data frequency is doubled to produce the frequency of data transitions, $f_d$.

The last example illustrates how to use a Cypress PALC22V10C-10 as a synchronizer. For a 10-year MTBF, assuming the maximum $f_c$ from Cypress's *Data Book* and $f_d$, the required $t_r$ is

$$t_r = (0.547 \times 10^{-9} s)[\ln(315 \times 10^{6} s)$$
$$+ \ln(90.9 \times 10^{6} \times 90.9 \times 10^{6} \times 8.08 \times 10^{-15})]$$
$$= 13.0 \; ns$$

Using this result, the synchronizer's maximum operating frequency is reduced from 90.9 MHz to

$$f_c = \frac{1}{\frac{1}{f_{max}} + t_r} = \frac{1}{\frac{1}{90.9MHz} + 13.0ns} = 41.6MHz$$

## Two-Stage Synchronization

As explained earlier, you can use a second register in series to perform two-stage synchronization (*Figure 4*). This is accomplished by feeding the output of the first synchronization register to the input of the second synchronization register. In PLDs, this method is common because the first synchronization stage can synchronize the asynchronous input signal, and the second synchronization stage can perform a Boolean function on a combination of the input and output signals. Boolean functions can be performed at either stage; the metastability characteristics listed in *Table 1* apply to PLD registers' asynchronous inputs that are used directly as well as asynchronous inputs used as a Boolean combination of existing inputs and outputs.

When implementing a two-stage synchronizer in a PLD, the probability that a synchronizer is metastable after the second stage of synchronization is the square of the probability that a synchronizer is metastable after the first stage of synchronization. The MTBF equation is

$$MTBF = \left( \frac{e^{\frac{t_r}{t_{sw}}}}{f_c f_d W} \right)^2$$

From this result, the equation for $t_r$ becomes

$$t_r = \frac{t_{sw}(\ln(MTBF) + 2 \times \ln(f_c f_d W))}{2}$$

Using this result for a two-stage synchronizer in a Cypress PALC22V10C, the $t_r$ for a 10-year MTBF is reduced from 13.0 ns to

$$t_r = (0.5)(0.547 \times 10^{-9} s)[\ln(315 \times 10^{6} s)$$
$$+ \ln(90.9 \times 10^{6} \times 90.9 \times 10^{6} \times 8.08 \times 10^{-15})]$$
$$= 7.65 \; ns$$

The maximum $f_c$ increases from 41.6 MHz to

$$f_c = \frac{1}{\frac{1}{f_{max}} + t_r} = \frac{1}{\frac{1}{90.9MHz} + 7.65ns} = 53.6MHz$$

This example shows that if the cycle of latency caused by the additional synchronization stage is acceptable, you can dramatically increase the synchronizer's maximum operating frequency.

## References

1. Lubkin, S., (Electronic Computer Corp.), “Asynchronous Signals in Digital Computers,” *Mathematical Tables and Other Aids to Computation*, Vol. 6, No. 40, October 1952, pp. 238-241.

2. Nootbaar, Keith, (Applied Microcircuits Corp.), “Design, Testing, and Application of a Metastable-Hardened Flip-Flop,” WESCON 87 (San Francisco, CA, Nov. 17-19, 1987), Electronic Conventions Management, Los Angeles, CA 90045.

3. Stoll, Peter A., “How to Avoid Synchronization Problems,” *VLSI Design*, November/December 1982, pp. 56-59.

4. Chapiro, Daniel M., *Globally-Asynchronous Locally-Synchronous Systems*, Stanford University, Department of Computer Science Report No. STAN-CS-84-1026, October 1984.

5. Horstmann, Jens U., Eichel, Hans W., Coates, Robert L., “Metastability Behavior of CMOS ASCI Flip-Flops in Theory and Test,” *IEEE Journal of Solid-State Circuits*, Vol. 24, No. 1, February 1989, pp. 146-157.

6. Wormald, E.G., “A Note on Synchronizer or Interlock Maloperation,” *Professional Program Session Record 16*, WESCON 87, November 17-19, 1987, Electronic Conventions Management, Los Angeles, CA 90045.

7. Pechouchek, Miroslav, “Anomalous Response Times of Input Synchronizers,” *IEEE Trans. Computers*, Vol. C-25, No. 2, February 1976, pp. 133-139.

8. Chaney, T. J., “Comments on ‘A Note on Synchronizer or Interlock Maloperation,” *IEEE Trans. Computing*, Vol. C-28, No. 10, Oct. 1979, pp. 802-804.

9. Couranz, George R., Wann, Donald F., “Theoretical and Experimental Behavior of Synchronizers Operating in the Metastable Region,” *IEEE Trans. Computers*, Vol. C-24, No. 6, June 1975, pp. 604-616.

10. Veendrick, Harry J.M., “The Behavior of Flip-Flop Used as Synchronizers and Prediction of Their Failure Rate,” *IEEE Journal of Solid-State Circuits*, Vol. SC-15, No. 2., April 1980, pp. 169-176.

11. Kacprzak, Tomasz, Albicki, Alexander, “Analysis of Metastable Operation in RS CMOS Flip-Flops,” *IEEE Journal of Solid-State Circuits*, Vol. SC-22, No. 1, February 1987, pp. 57-64.

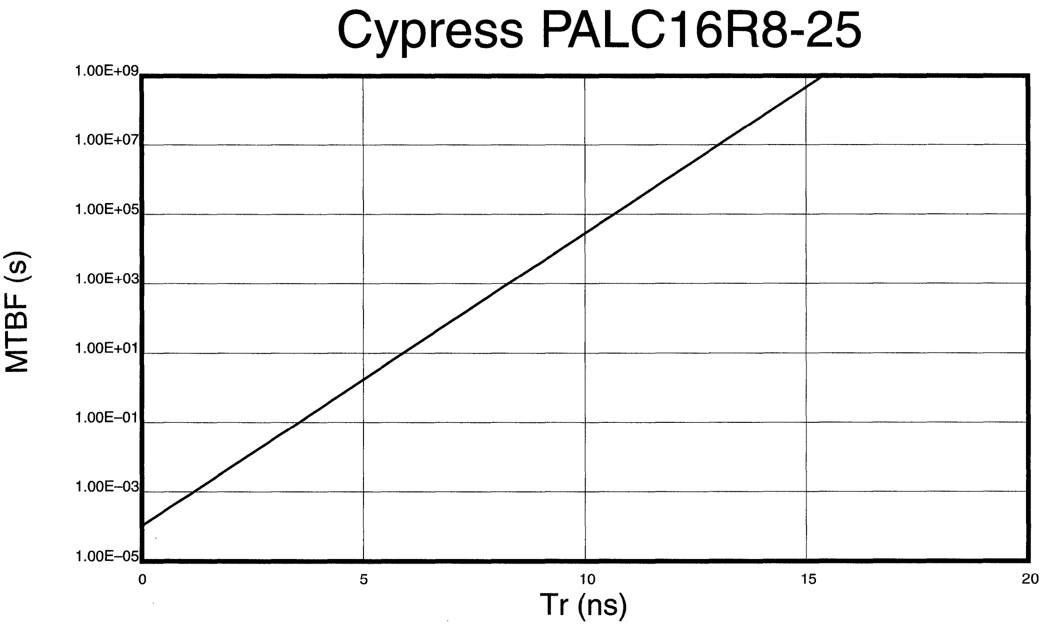
12. Flannagan, Stephen T., “Synchronization Reliability in CMOS Technology,” *IEEE Journal of Solid-State Circuits*, Vol. SC-20, No. 4, Aug 1985, pp. 880-882.

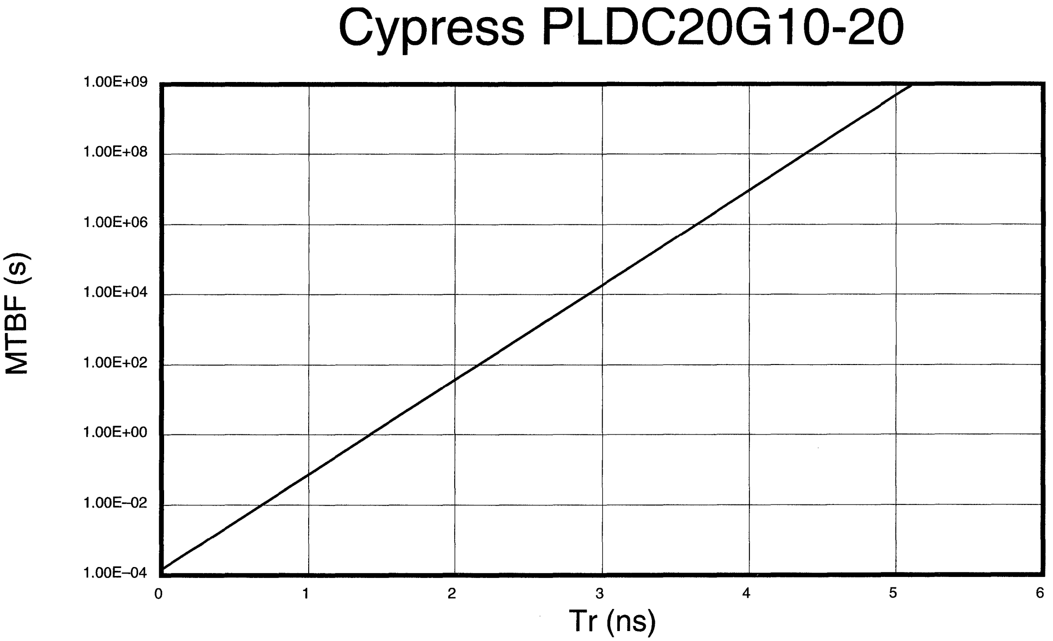
13. Wakerly, John F., *A Designers Guide to Synchronizers and Metastability*, Center for Reliable Computing Technical Report, CSL TN #88-341, February, 1988 Computer Systems Laboratory, Departments of Electrical Engineering and Computer Science, Stanford University, Stanford, CA.

14. Freeman, Gregory G., Liu, Dick L., Wooley, Bruce, and McClusky, Edward J., *Two CMOS Metastability Sensors*, CSL TN# 86-293, June 1986, Computer Systems Laboratory, Electrical Engineering and Computer Science Departments, Stanford University, Stanford, CA.

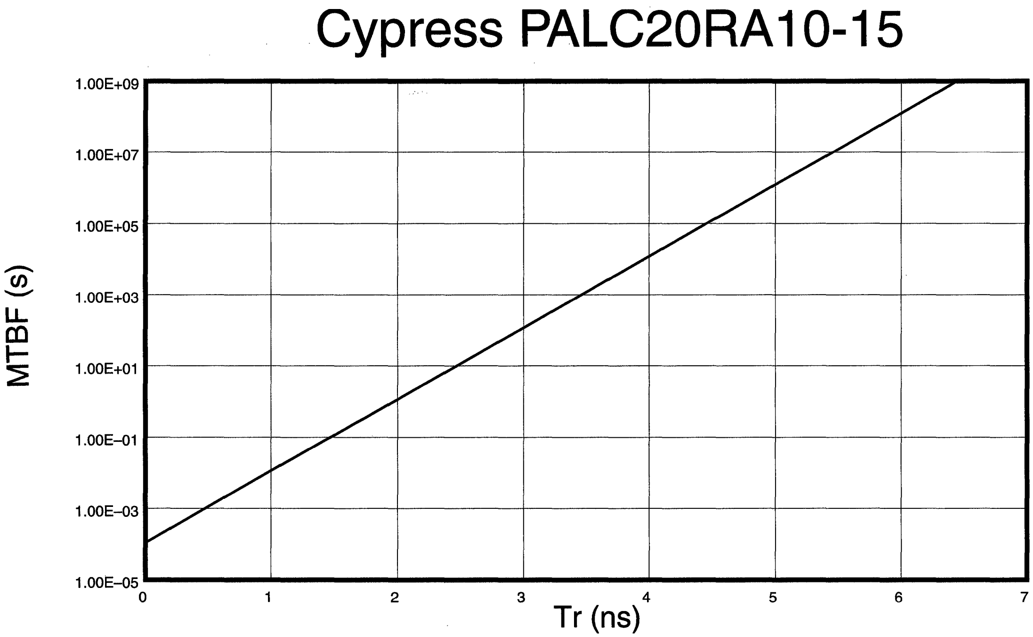
15. Rubin, Kim, “Metastability Testing in PALs,” WESCON 87 (San Francisco, CA, Nov. 17-19, 1987), Electronic Conventions Management, Los Angeles, CA 90045.

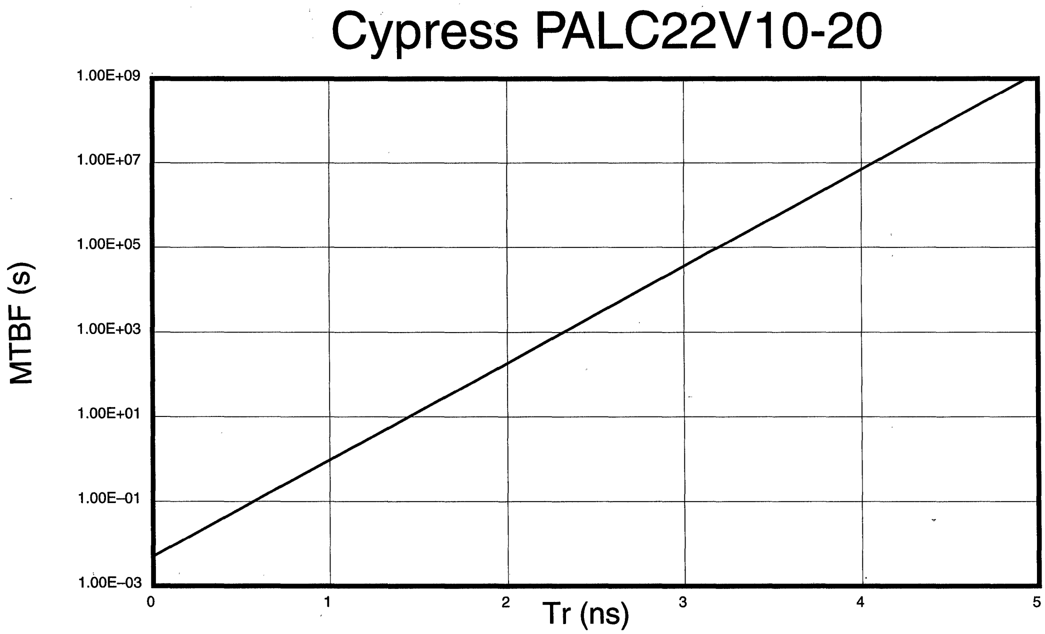
## Appendix A. Metastability Graphs of Cypress Devices



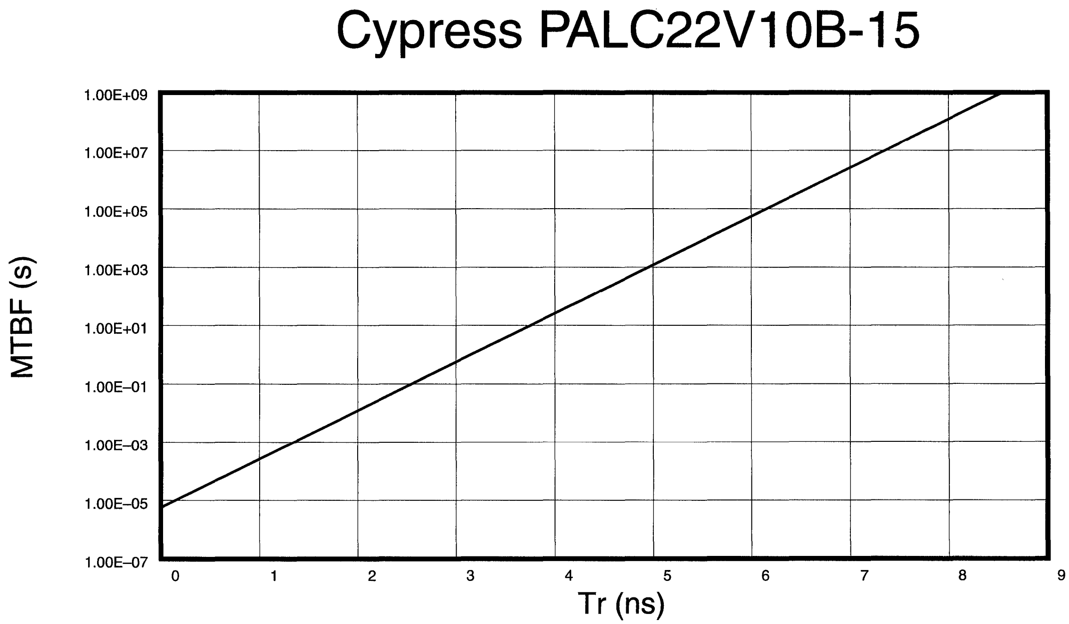


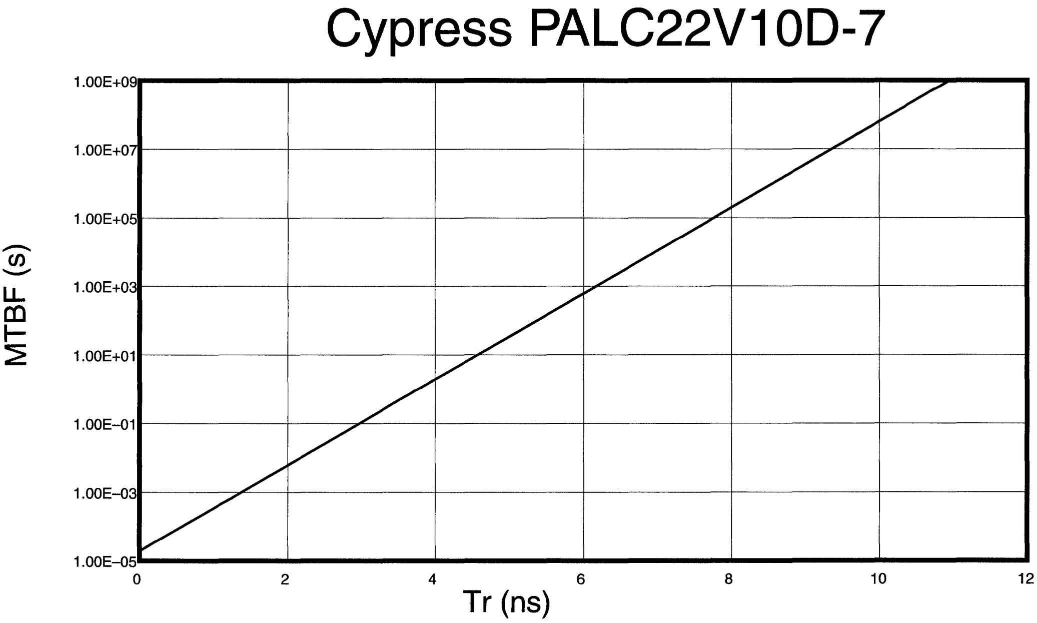
### Appendix A. Metastability Graphs of Cypress Devices (continued)





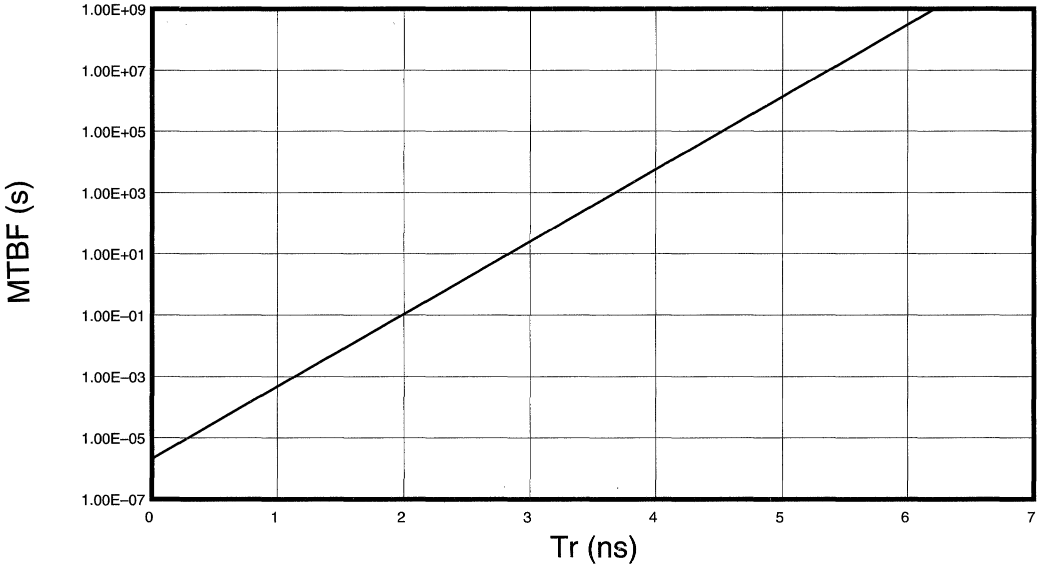
**Appendix A. Metastability Graphs of Cypress Devices** (continued)



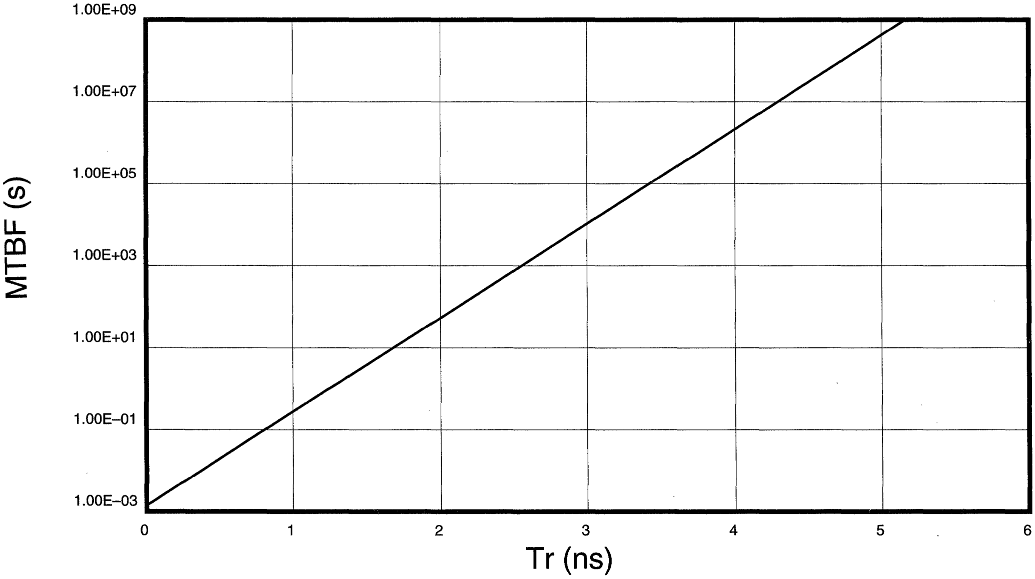


## Appendix A. Metastability Graphs of Cypress Devices (continued)

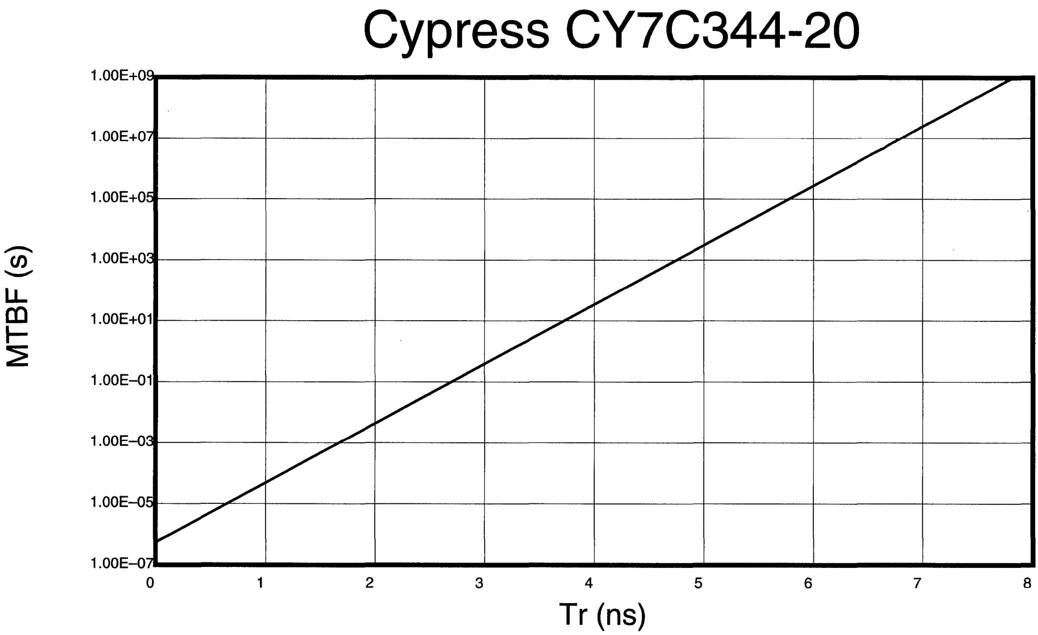
### Cypress CY7C331-20



### Cypress CY7C335-100



**Appendix A. Metastability Graphs of Cypress Devices** (continued)



# Designing with the CY7C335 and *Warp2*® VHDL Compiler

This application note provides an overview of the CY7C335 Universal Synchronous EPLD architecture and *Warp2*® VHDL Compiler for PLDs. Example designs demonstrate how the *Warp2* VHDL compiler takes advantage of the rich architectural features of the CY7C335.

The CY7C335 is a synchronous EPLD optimized for high-performance state machines and other clocked systems that operate at speeds of up to 100 MHz. The CY7C335 uses Cypress's low-power, 0.8-micron CMOS UV erasable technology and is packaged in 28-pin, 300-mil dual in-line and LCC/PLCC packages.

The CY7C335 builds on the popularity of the high-speed CMOS PALCE22V10 and exceeds the capability of the 26V12 and 26CV12. The CY7C335 offers significantly higher density solutions and can replace as many as four 22V10s. It has 258 variable product terms for 16 state registers (ranging from 9 to 19 product terms per macrocell), macrocells that can be configured as JK-, RS-, T-, or D-type, bidirectional pins, bypassable input registers, three clocks, and a product term output enable for each macrocell.

In addition to supporting the features of the CY7C335, the *Warp2* VHDL compiler enables the designer to create designs, using any combination of high-level behavioral descriptions, Boolean equations, state tables, or RTL structures, that can easily be retargeted to any Cypress PLD.

*Warp2* is a state-of-the-art VHDL compiler that facilitates device-independent designs by synthesizing for a powerful subset of IEEE1076. Optimization and reduction algorithms automatically select T- or D-type flip-flops and perform automatic state and pin assignment. *Warp2* includes a graphical user interface (which runs under Windows™ for the PC, and OpenLook™ or Motif™ for the Sun) and comes complete with a functional simulator for graphical waveform simulation.

## Overview of the CY7C335

*Figure 1* is the block diagram of the CY7C335. Three separate clock signals—two input and two output clocks (one shared)—can be used on pins 1, 2, and 3. Alternatively, pins 2 and 3 can be used as two of twelve inputs that may be registered or fed directly to the programmable AND array. Pin 14 can be used as an input or as a common output enable for each I/O pin. Outputs can also be enabled by product terms. The device features center ground and supply pins that reduce ground bounce due to parasitic effects, particularly lead inductance.

*Figure 2* illustrates the input macrocell. Each D-type input register can use either ICLK1 or ICLK2. Alternatively, the input register bypass multiplexer can be programmed to allow the signal to feed directly to the array as combinatorial input.

Twelve configurable I/O macrocells enable JK-, RS-, T-, or D-type state registers to optimize for minimal product terms. *Figure 3* illustrates the I/O macrocell, which includes the following features: (1) registered or combinatorial output; (2) global (by pin 14) or product term output enable; (3) global, synchronous, product-term set and reset; (4) three clocks–two can be used as input clocks and two can be used as output clocks (with one shared); (5) input/output flexibility (the cell can be configured as input only, output only, or a dedicated input with a buried register by using the shared input multiplexer and thereby maximizing cell resource utilization).

In addition to the input and I/O macrocells, the CY7C335 features four hidden macrocells, one of which is shown in *Figure 4.* Buried registers are highly useful for state machines, internal counters, or other applications that need registers that are not also used as outputs.

The clocking scheme is shown in *Figure 5.* The CY7C335 can utilize three separate clocks. Two clocks are inputs to each of the input clock multiplexers and state clock multiplexers. If two clocks are used on both the input and the state registers, then one of the clocks is shared, because a total of three clocks are supported. Pin 1 is a dedicated state clock pin, designated SCLK1 (state clock). Pins 2 and 3 may be used as either inputs or clocks, as shown in *Figure 5.*

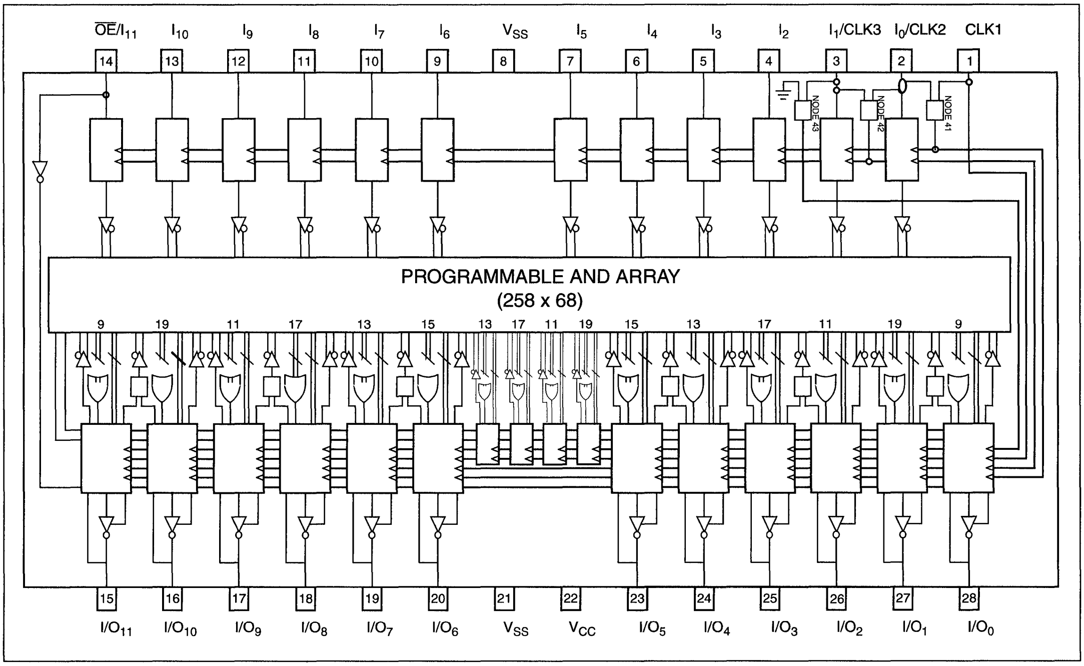


**Figure 1. CY7C335 Block Diagram**

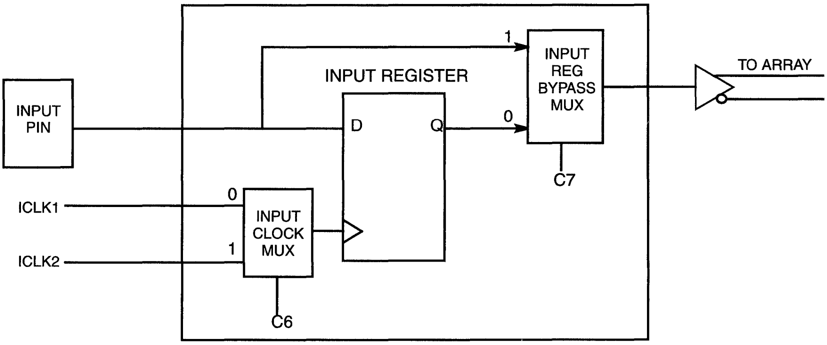


**Figure 2. CY7C335 Input Macrocell**

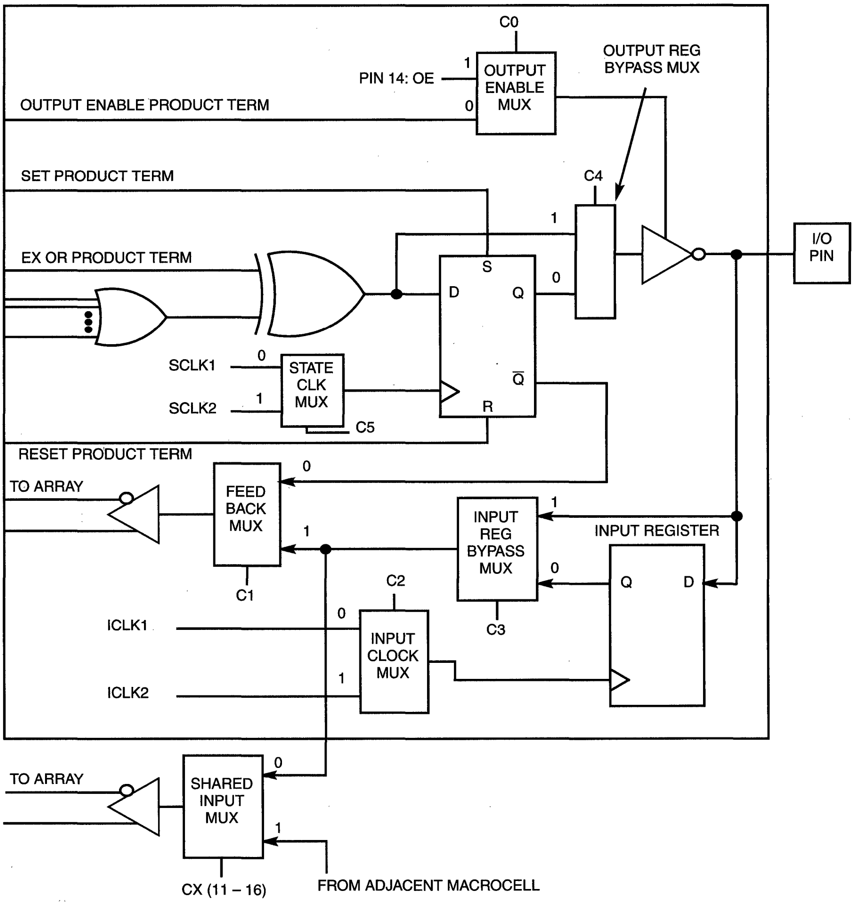


**Figure 3. CY7C335 Input/Output Macrocell**

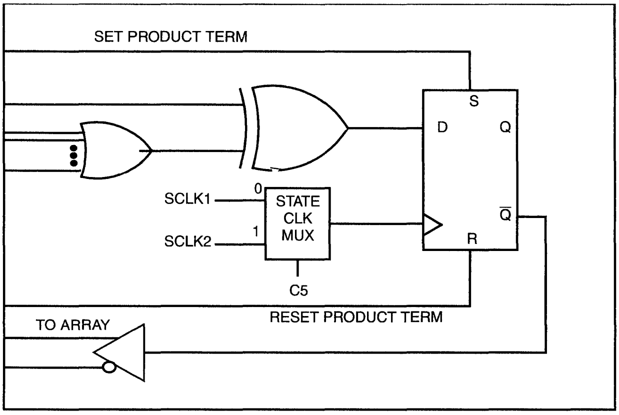


**Figure 4. CY7C335 Hidden Macrocell**

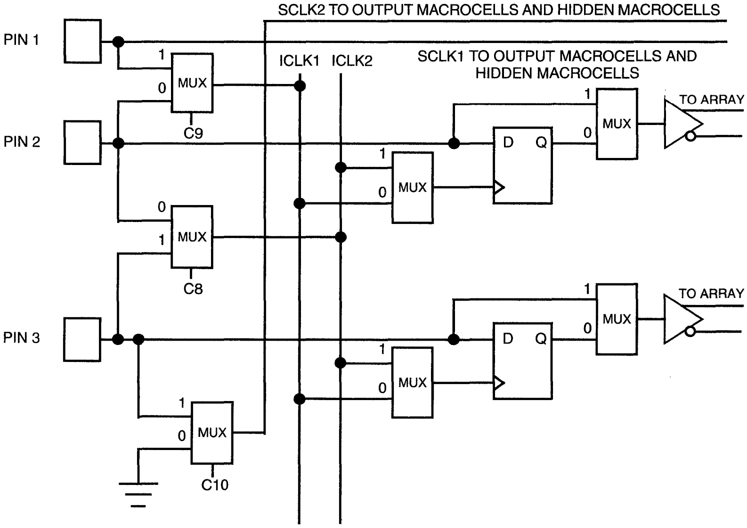

SCLK2 TO OUTPUT MACROCELLS AND HIDDEN MACROCELLS
PIN 1
PIN 2
PIN 3
ICLK1
ICLK2
SCLK1 TO OUTPUT MACROCELLS AND HIDDEN MACROCELLS
MUX
C9
C8
C10
D
Q
TO ARRAY
TO ARRAY
1
0

**Figure 5. CY7C335 Input Clocking Scheme**

## Overview of *Warp2*

*Warp2* is a state-of-the-art VHDL compiler for designing with Cypress PLDs. *Warp2* accepts VHDL designs, synthesizes and optimizes the entered design, and outputs a JEDEC file for the CY7C335. *Warp2* also provides a graphical waveform simulator for functional simulation. *Figure 6* illustrates the *Warp2* design flow.

### VHDL Compiler

As an open, non-proprietary, IEEE1076 compliant language that is the standard for behavioral design entry and simulation, VHDL allows designers to easily describe complex hardware systems.

*Warp2's* VHDL enables designers to describe device-independent designs at different levels of abstraction, including behavioral descriptions, Boolean equations, state tables, and structural descriptions. In addition, VHDL and *Warp2* support hierarchical designs, allowing either a "top-down" or "bottom-up" approach to design.

## Design Examples

The following design examples demonstrate how to use *Warp2* and VHDL to take advantage of the CY7C335 architectural features. The purpose is to show some VHDL constructs that are particularly useful for the CY7C335 architecture as well as point out designs that are well suited for the device. Further information on VHDL constructs may be found in the *Warp2 Reference Manual* or one of several texts available on the language. For each of the examples, the complete VHDL source code and an excerpt of the report file may be found in the appendices.

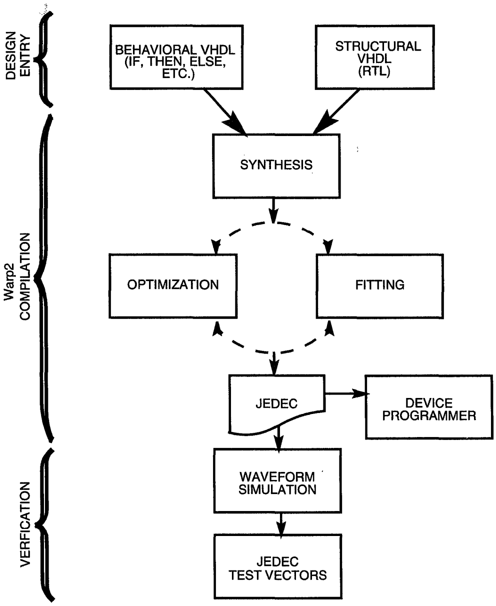


**Figure 6. *Warp2* Design Flow**

### Pipelined Buffer

This example demonstrates how to use VHDL code to implement a pipelined buffer (see *Figure 7*) with multiple clocks and output enables. The CY7C335 is well suited for pipelined applications because it has input registers in both the input macrocells as well as the I/O macrocells.

The complete VHDL source code is printed in Appendix A of this note. The pipeline architecture is reprinted in *Figure 8.*

The pipeline is implemented in three processes. The first process registers (using CLK1 on the input registers) the upper four bits of the input signal I. The second process registers the lower four bits, using CLK2. The signal INTMP represents the q output of these registers. The third process registers the signal INTMP, with OUTTMP being the q output of these registers. This signal reaches the I/O pins if output is enabled, as explained below.

Below the three processes is a generation scheme which is used to instantiate eight triout components (see *Figure 9*). The *triout* components are used to implement an output enable. The upper four bits of the output are enabled by OE1 (which is assigned to pin 14 by *Warp*) and the lower four bits use a product term output enable, PTOE.

The complete VHDL source code for this example is in Appendix A. A report file excerpt, showing resource utilization, is shown in Appendix B. This excerpt shows that 8 of 12 I/O macrocells were utilized. However, not all resources (the input registers, for example) in those macrocells were utilized.

### Comparator with Registered Inputs

In high-speed systems, such as microprocessor local buses that operate at 40, 50, or 66 MHz, data or addresses must be captured from the bus (when qualified with a strobe) with set-up times of 3 to 5 ns. Few logic functions can be implemented in this time, and for this reason data or addresses are captured and then processed in pipeline fashion. The CY7C335, with its input registers, is well suited for such high-speed systems.

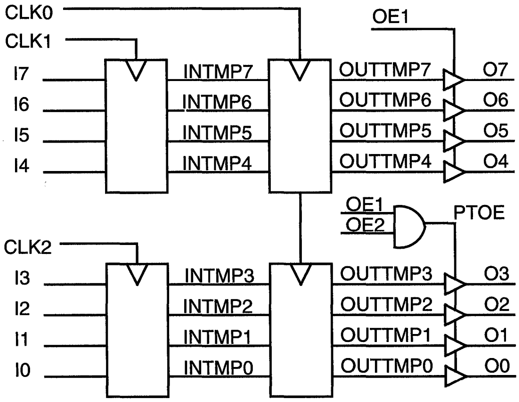
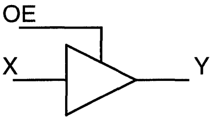


**Figure 7. Pipelined Buffer Block Diagram**

```
use work.rtlpkg.all;
architecture archpipe of pipe is
signal intmp, outtmp: bit_vector(7 downto 0);
signal ptoe: bit;
begin
proc1: process
  begin
    wait until clk1 = '1';
       intmp(7 downto 4) <= i(7 downto 4);
  end process;

proc2: process
  begin
    wait until clk2 = '1';
       intmp(3 downto 0) <= i(3 downto 0);
  end process;

proc3: process
  begin
    wait until clk0 = '1';
       outtmp <= intmp;
  end process;

  ptoe <= oe1 AND oe2;

g1:  for j in 7 downto 0 generate
  g2:if j > 3 generate
       t1x: triout port map(outtmp(j), oe1, o(j));
    end generate;
  g3: if j < 4 generate
       t2x: triout port map(outtmp(j), ptoe, o(j));
    end generate;
  end generate;

end archpipe;
```

**Figure 8. Pipeline Architecture**



```
component triout
 port(
   x: in bit; --input to buffer
   oe: in bit; -- output enable
   y: out bit; -- output
 );
end component;
```

**Figure 9. Triout Component**

In this simple, register-intensive example, all 18 inputs are registered and the output is combinatorial (*Figure 10*). As noted in Appendix D, this design leaves much of the CY7C335's resources free for additional logic. The 22V10, however, would be unable to fit a 5-bit comparator with registered inputs. Ten macrocells would be consumed when registering the inputs, leaving no macrocells for the AEQB combinatorial output. The 22V10 fares poorly in such pipelined systems because it does not have input registers and must therefore waste output macrocell resources.

The VHDL source code can be found in its entirety in Appendix C. The architecture is reprinted below.

```
architecture archcomp of comp is
signal a, b: bit_vector(0 to 8);
begin
proc1:    process begin
 wait until clk = '1';
   a <= a_in;
   b <= b_in;
 end process;

 aeqb <= '1' when a=b else '0';

end archcomp;
```

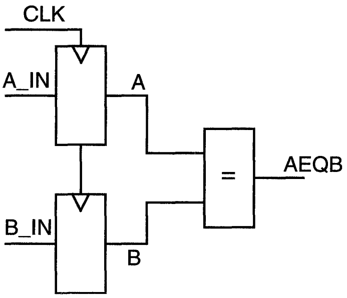


**Figure 10. Comparator**

The process is used to register the inputs on the rising edge of CLK1. The equation for AEQB is placed outside of the process because it is a combinatorial output.

**Multiplexer with Registered Inputs and Outputs**

Registered multiplexers and demultiplexers demand a large number of inputs and outputs. This example (see *Figure 11*) takes advantage of the CY7C335 input and output registers, two groups of six-bit-wide signals are captured via the input registers and signal SEL selects one of the groups, which is then registered on the output. The complete VHDL source code can be found in Appendix E. The architecture is reprinted below.

```
architecture archmux of mux is
signal x, y: bit_vector(5 downto 0);
begin
proc1:    process begin
  wait until clk = '1';
    x <= xin;
    y <= yin;
    if sel = '1' then
      qout <= x;
    elsif sel = '0' then
      qout <= y;
    end if;
  end process;
end archmux;
```

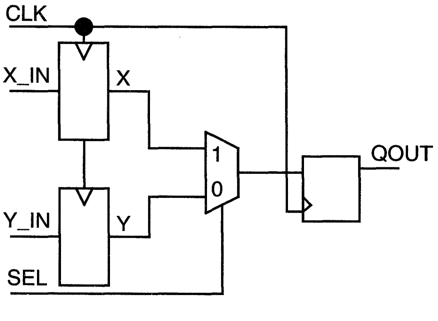


**Figure 11. Multiplexer**

On the rising edge of CLK1, the inputs are registered while the outputs are propagated. Thus, data on the inputs is not propagated to the outputs until the second rising edge.

**Decoder**

Faster microprocessors require decoders to operate at higher frequencies. Many high-density PLDs and FPGAs cannot meet speed requirements, leaving designers to opt for ASIC-based solutions which can be time consuming and expensive. The CY7C335 is another option.

Consider a 16-bit address that requires decoding to address system memory elements (SRAM, PROM, EEPROM and "shadow" RAM) and two peripheral ports. At times other than boot-up, the microprocessor fetches instructions from shadow RAM that is loaded from PROM during boot-up. *Figure 12* shows the VHDL architecture that decodes the memory map shown in *Figure 13.* Appendix H shows that the CY7C335's resources easily handle this application while operating at speeds to 100 MHz.

**Up/Down Counter with Upper and Lower Limits**

This example demonstrates how to use VHDL code to implement the up/down counter shown in *Figure 14.* The CY7C335 is particularly well suited for this design because it supports three clocks and has flexible I/O. This design requires the following resources: three clocks (two inputs and one state), eight input registers for the lower limit, eight input registers for the upper limit, one input each for the preset HIGH, preset LOW, reset, and output enable signals, eight state registers for the counter, one state register each for the comparators, and one state register for the counter direction signal.

A total of 20 inputs and 8 outputs are required; consequently, this design utilizes bidirectional signals. The counter output is three-stated to load six bits of the upper limit into input registers of I/O macrocells. For example, the least-significant counter bit is stored in a state register and the least-significant upper-limit bit is stored in the input register of the same macrocell. The least-significant upper-limit bit feeds into the array via the shared input multiplexer. (The shared-input multiplexers are placed between adjacent I/O macrocells, and allow for input when the macrocell register is buried.) The CY7C335 provides six of these multiplexers. The two most significant bits of the upper limit are passed into the array through an I/O pin configured as a dedicated input. The two most significant bits of the upper limit and counter may be externally tied together so the design can be bidirectional.

The up/down counter counts between limits stored in the input registers. The lower-limit (LL) is loaded into the registers on the rising edge of CLK1 while the upper limit is loaded on the rising edge of CLK2. On CLK0, if preH is asserted, then the upper limit is loaded into the counter, and if preL is asserted, then the lower limit is loaded into the counter.

The 22V10 would not suffice for this design. Although the 22V10 has been an attractive choice of devices to implement counters and state machines, it suffers a limitation in addition to its poor handling of pipelined systems: it does not have any buried registers.

In counters and encoded state machines, registers often need not be apparent to the outside, meaning the registers can be buried within the device. In the 22V10, all macrocells are connected to I/O pins. Thus, even when a macrocell register is being used in a buried sense, the I/O pin is committed, thereby preventing the pin from being used as an additional input to the device.

In addition to overcoming the 22V10's shortcoming with pipelined systems by having input registers in both the input and I/O macrocells, the CY7C335 provides a solution to the 22V10's density problems with regards to counters and state machines by providing four buried registers. Additionally, pairs of macrocells have a shared input multiplexer that allows up to six additional inputs, even when all twelve I/O macrocells have their registered outputs feeding back into the AND array.

```
use work.bv_math.all;
architecture behav of decode is
signal address: bit_vector(15 downto 0);
begin
  address <= a & "000";

proc1: process begin
  wait until clk = '1';
     promsel <= '0';
     shadowsel <= '0';
     periph1 <= '0';
     periph2 <= '0';
     sramsel <= '0';
     eesel <= '0';
     if valid = '1' then
       if address >= x"0000" and address < x"4000" then
         if bootover = '0' then
           promsel <= '1';
         else
           shadowsel <= '1';
         end if;
       elsif address >= x"4000" and address < x"4008" then
         periph1 <= '1';
       elsif address >= x"4008" and address < x"4010" then
         periph2 <= '1';
         elsif address >= x"8000" and address < x"C000" then
         sramsel <= '1';
       else address >= x"C000" then
         eesel <= '1';
       end if;
     end if;
  end process;
end behav;
```

**Figure 12. VHDL Architecture**

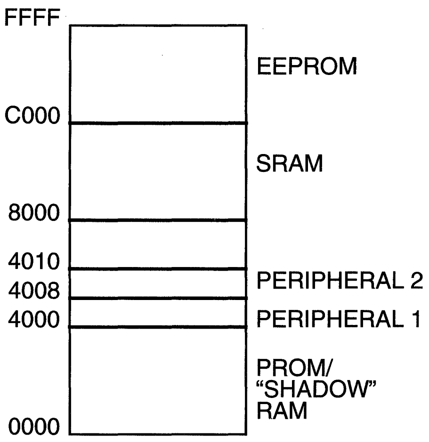


**Figure 13. Decoder Memory Map**

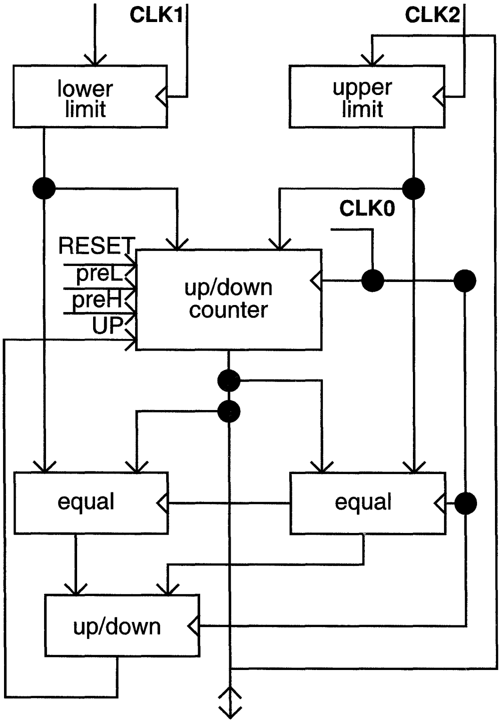


**Figure 14. Up/Down Counter**

The VHDL source code for this example is in Appendix I of this note, and the architecture is reprinted in *Figure 15.*

The up/down counter is implemented in three processes, a generation scheme, and two concurrent statements. In the first process, the lower limit is registered on the rising edge of CLK1. The signal LOWER registers the input signal LL. The second process registers the upper limit on the rising edge of CLK2. The third process implements (1) the up/down counter with reset, preset LOW, and preset HIGH, (2) two comparators, and (3) the direction signal (DIR) that indicates to count up (logic 1) or down (logic 0). The comparators and the direction signal are clocked by CLK0, forcing the counter to change direction from up to down or vice-versa two clock cycles after the count matches one of the limits. For this reason, the upper limit should be loaded with a value two less than the greatest desired count, and the lower limit should be loaded with a value two greater than the least desired count.

The generation scheme below the three processes is a means to instantiate 6 *bufoe* components (see *Figure 16*) and two *triout* components. The *bufoe* components are used to implement the output enable and provide a feedback path for the upper limit. The CY7C335 has six shared input multiplexers that allow six bits of the signal count to utilize the state registers while enabling six bits of the upper limit to be loaded into the input register associated with the same macrocell. The remaining two bits of count will be placed in I/O macrocells in which the input registers are not used, and the two bits of the upper limit will be in two I/O macrocells configured as inputs. To enable bidirectional operation, the input and output pins for the associated upper limit and count bits can be connected externally. This is the reason for instantiating two triout components on the most significant two bits of the count.

### Serial Decoder

The CY7C335's state registers and abundant product terms make it a good choice in which to implement state machines. The following VHDL code uses a state machine to implement a serial decoder that searches for a synchronization word within serially transmitted data. The sync word is the byte 11101000 and is expected to be repeated every 16 bytes. When the sync word is found, MATCH is asserted. When the sync word is found separated by 15 bytes three consecutive times, LOCK is asserted. The state diagram for this example is shown in *Figure 17.*

The architecture of this design is printed in *Figure 18* and the complete VHDL code is in Appendix K. The resources that this design uses (Appendix L) show that there is room for more logic within the device. For instance, the comparator with registered inputs described earlier could fit in the device along with this design.

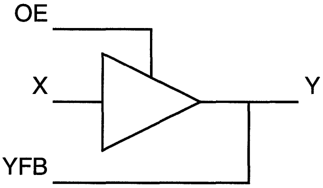
The first process within the architecture defines the state transitions. The second process is one that is synchronized by the clock. The output MATCH is determined by the present inputs and the currents state. This implements a Mealy machine. The counter process counts the number of bits after a match, and the synchronizer process checks to see if a match occurs 15 bytes after the previous one. If a match is separated by 15 bytes for three consecutive times, then on the fourth consecutive match separated by 15 bytes, LOCK is asserted.

```
use work.bv_math.all;
use work.rtlpkg.all;

architecture archupdown of updown is
signal lower, upper, ul, count: bit_vector(0 to 7);
signal cequ, ceql, dir: bit;
begin
proc1: process
  begin
     wait until clk1 = '1';
       lower <= ll;
  end process;

proc2: process
  begin
     wait until clk2 = '1';
       upper <= ul;
  end process;

proc3: process
  begin
     wait until clk0 = '1';
     -- implement counter
     if reset = '1' then
       count <= x"00";
     elsif preL = '1' then
       count <= lower;
     elsif preH = '1' then
       count <= upper;
     elsif (dir = '1') then
       count <= inc_bv(count);
     else
       count <= dec_bv(count);
     end if;-- implement comparators & direction signal
     if count = upper then
       cequ <= '1';
     else
       cequ <= '0';
     end if;
     if count = lower then
       ceql <= '1';
     else
       ceql <= '0';
     end if;

     if ceql = '1' then
       dir <= '1';
     elsif cequ = '1' then
       dir <= '0';
     else
       dir <= dir;
     end if;
  end process;
```

**Figure 15. Architecture**

```
g1:  for i in 0 to 7 generate
  bidir:if i < 6 generate
     bx:bufoe port map(count(i), outen, countio(i), ul(i));
     end generate;
  trist:if i > 5 generate
     tx:triout port map(count(i), outen, countio(i));
     end generate;
  end generate;

  ul(6) <= ul6;
  ul(7) <= ul7;

end archupdown;
```

**Figure 15. Architecture** (continued)



```
component bufoe
 port(
   x: in bit; --input to buffer
   oe: in bit; --output enable
   y: inout x01z; --x01z output
   yfb: out bit; -- feedback
   );
end component;
```

**Figure 16. bufoe Component**

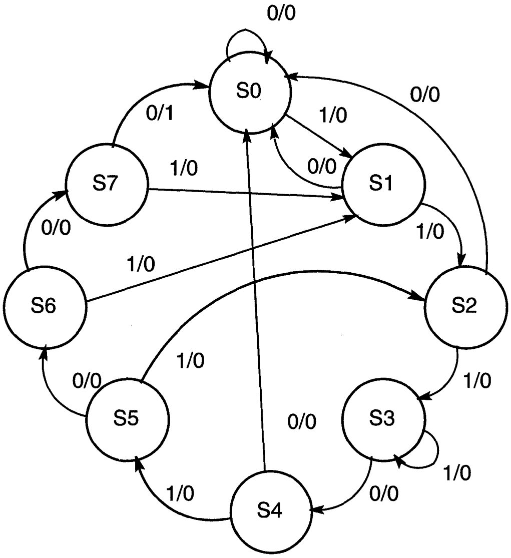


**Figure 17. State Diagram**

```
use work.int_math.all;
use work.bv_math.all;
architecture archserial of serial is
type states is (state0, state1, state2, state3, state4, state5, state6,
     state7);
signal state, nextstate: states;
signal match_cnt: bit_vector(1 downto 0);
signal bit_cnt: bit_vector(6 downto 0);

begin
fsm:
process  begin
 match <= ’0’;
 case state is
    when state0 =>
      if data = ’1’ and (lock = ’0’ or bit_cnt = “1111000") then
        nextstate <= state1;
      else
        nextstate <= state0;
      end if;
    when state1 =>
      if data = ’1’ then
        nextstate <= state2;
      else
        nextstate <= state0;
      end if;
when state2 =>
      if data = ’1’ then
        nextstate <= state3;
      else
        nextstate <= state0;
      end if;
    when state3 =>
      if data = ’0’ then
        nextstate <= state4;
      else
        nextstate <= state3;
      end if;
    when state4 =>
      if data = ’1’ then
        nextstate <= state5;
      else
        nextstate <= state0;
      end if;
    when state5 =>
      if data = ’0’ then
        nextstate <= state6;
      else
        nextstate <= state2;
      end if;
```

**Figure 18.**

```
    when state6 =>
      if data = '0' then
        nextstate <= state7;
      else
        nextstate <= state1;
      end if;
    when state7 =>
      if data = '0' then
        nextstate <= state0;
        match <= '1';
      else
        nextstate <= state1;
      end if;
--No "when others" needed since CASE is completely defined.
 end case;
end process;

mealy:
process begin
 wait until clk = '1';
    state <= nextstate;
end process;

counter:
process begin
 wait until clk = '1';
 if match = '1' then
    bit_cnt <= "0000000";
 else
    bit_cnt <= inc_bv(bit_cnt);
 end if;
end process;

synchronizer:
process begin
 wait until clk = '1';
 if bit_cnt = "1111111" then
    if match = '1' then
      if match_cnt = "11" then
        lock <= '1';
      else
        match_cnt <= inc_bv(match_cnt);
      end if;
    else
      match_cnt <= "00";
      lock <= '0';
    end if;
 end if;
end process;

end archserial;
```

**Figure 18.** (continued)

## Appendix A. *Warp2* VHDL Source Code for Pipelined Buffer

```
entity pipe is
  port(clk0, clk1, clk2: in bit;
     oe1, oe2: in bit;
  i: in bit_vector(7 downto 0);
  o: out x01z_vector(7 downto 0));
end pipe;

use work.rtlpkg.all;
architecture archpipe of pipe is
signal intmp, outtmp: bit_vector(7 downto 0);
signal ptoe: bit;
begin
proc1: process
  begin
     wait until clk1 = '1';
      intmp(7 downto 4) <= i(7 downto 4);
  end process;

proc2: process
  begin
    wait until clk2 = '1';
      intmp(3 downto 0) <= i(3 downto 0);
  end process;

proc3: process
  begin
    wait until clk0 = '1';
      outtmp <= intmp;
  end process;

  ptoe <= oe1 AND oe2;

g1:  for j in 7 downto 0 generate
  g2:if j > 3 generate
      t1x: triout port map(outtmp(j), oe1, o(j));
    end generate;
  g3: if j < 4 generate
      t2x: triout port map(outtmp(j), ptoe, o(j));
    end generate;
  end generate;

end archpipe;
```

## Appendix B. *Warp2* Report File Excerpt for Pipelined Buffer

```
Information: Macrocell Utilization.

                    Description          Used      Max
                ______________________________________
                | Dedicated Inputs    |    9   |    9  |
                | Clock/Inputs        |    3   |    3  |
                | Enable/Inputs       |    1   |    1  |
                | I/O Macrocells      |    8   |   12  |
                | Buried Macrocells   |    0   |    4  |
                ______________________________________
                                          21   /   29    = 72   %

  Information: Output Logic Product Term Utilization.

                 Node#  Output Signal Name  Used   Max
                ________________________________________
                | 15   | o_0_             |  1  |   9  |
                | 16   | Unused           |  0  |  19  |
                | 17   | o_2_             |  1  |  11  |
                | 18   | Unused           |  0  |  17  |
                | 19   | o_4_             |  1  |  13  |
                | 20   | o_7_             |  1  |  15  |
                | 23   | o_6_             |  1  |  15  |
                | 24   | o_5_             |  1  |  13  |
                | 25   | Unused           |  0  |  17  |
                | 26   | o_3_             |  1  |  11  |
                | 27   | Unused           |  0  |  19  |
                | 28   | o_1_             |  1  |   9  |
                | 29   | Unused           |  0  |   1  |
                | 30   | Unused           |  0  |   1  |
                | 31   | Unused           |  0  |  13  |
                | 32   | Unused           |  0  |  17  |
                | 33   | Unused           |  0  |  11  |
                | 34   | Unused           |  0  |  19  |
                ________________________________________
                                             8   / 230    = 3    %
```

## Appendix C. *Warp2* Source Code for Comparator

```
entity comp is port (
  clk: in bit;
  a_in, b_in: bit_vector(0 to 8);
  aeqb: out bit);
end comp;

architecture archcomp of comp is
signal a, b: bit_vector(0 to 8);
begin
proc1: process begin
  wait until clk = ’1’;
    a <= a_in;
    b <= b_in;
  end process;

  aeqb <= ’1’ when a=b else ’0’;

end archcomp;
```

### Appendix D. *Warp2* Report File Excerpt for Comparator

```
Information: Macrocell Utilization.

                 Description          Used      Max
           ___________________________________________
           | Dedicated Inputs     |     9   |     9   |
           | Clock/Inputs         |     3   |     3   |
           | Enable/Inputs        |     1   |     1   |
           | I/O Macrocells       |     7   |    12   |
           | Buried Macrocells    |     0   |     4   |
           ___________________________________________
                                       20   /    29    = 68  %

 Information: Output Logic Product Term Utilization.

            Node#  Output Signal Name  Used   Max
           ________________________________________
           | 15  |  Used As Input    |  0  |   9  |
           | 16  |  Used As Input    |  0  |  19  |
           | 17  |  Used As Input    |  0  |  11  |
           | 18  |  Used As Input    |  0  |  17  |
           | 19  |  Used As Input    |  0  |  13  |
           | 20  |  Used As Input    |  0  |  15  |
           | 23  |  Unused           |  0  |  15  |
           | 24  |  Unused           |  0  |  13  |
           | 25  |  Unused           |  0  |  17  |
           | 26  |  Unused           |  0  |  11  |
           | 27  |  aeqb             | 18  |  19  |
           | 28  |  Unused           |  0  |   9  |
           | 29  |  Unused           |  0  |   1  |
           | 30  |  Unused           |  0  |   1  |
           | 31  |  Unused           |  0  |  13  |
           | 32  |  Unused           |  0  |  17  |
           | 33  |  Unused           |  0  |  11  |
           | 34  |  Unused           |  0  |  19  |
           ________________________________________
                                       18  / 230    = 7    %
```

## Appendix E. *Warp2* Source Code for Multiplexer

```
entity mux is port(
  clk, sel: in bit;
  xin, yin: in bit_vector(5 downto 0);
  qout: out bit_vector(5 downto 0));
end mux;

architecture archmux of mux is
signal x, y: bit_vector(5 downto 0);
begin
proc1: process begin
  wait until clk = '1';
    x <= xin;
    y <= yin;
    if sel = '1' then
      qout <= x;
    elsif sel = '0' then
      qout <= y;
    end if;
  end process;
end archmux;
```

## Appendix F. *Warp2* Report File Excerpt for Multiplexer

```
Information: Macrocell Utilization.

                    Description          Used      Max
                ______________________________________
                | Dedicated Inputs     |    9   |    9   |
                | Clock/Inputs         |    3   |    3   |
                | Enable/Inputs        |    1   |    1   |
                | I/O Macrocells       |    7   |   12   |
                | Buried Macrocells    |    0   |    4   |
                ______________________________________
                                           20   /   29    = 68  %

 Information: Output Logic Product Term Utilization.

                 Node#  Output Signal Name  Used   Max
                ________________________________________
                | 15   |  qout_0_          |  2  |   9  |
                | 16   |  Used As Input    |  0  |  19  |
                | 17   |  qout_2_          |  2  |  11  |
                | 18   |  Unused           |  0  |  17  |
                | 19   |  qout_4_          |  2  |  13  |
                | 20   |  Unused           |  0  |  15  |
                | 23   |  Unused           |  0  |  15  |
                | 24   |  qout_5_          |  2  |  13  |
                | 25   |  Unused           |  0  |  17  |
                | 26   |  qout_3_          |  2  |  11  |
                | 27   |  Unused           |  0  |  19  |
                | 28   |  qout_1_          |  2  |   9  |
                | 29   |  Unused           |  0  |   1  |
                | 30   |  Unused           |  0  |   1  |
                | 31   |  Unused           |  0  |  13  |
                | 32   |  Unused           |  0  |  17  |
                | 33   |  Unused           |  0  |  11  |
                | 34   |  Unused           |  0  |  19  |
                ________________________________________
                                            12  / 230    = 5    %
```

## Appendix G. *Warp2* VHDL Source Code for Decoder

```
entity decode is port(
  a: in bit_vector(15 downto 3);
  rdwritebar, valid, bootover, clk: in bit;
  sramsel, promsel, eesel, shadowsel, periph1, periph2: out bit);
end decode;

use work.bv_math.all;
architecture behav of decode is
signal address: bit_vector(15 downto 0);
begin
  address <= a & ″000";

proc1: process begin
  wait until clk = '1';
    promsel <= '0';
    shadowsel <= '0';
    periph1 <= '0';
    periph2 <= '0';
    sramsel <= '0';
    eesel <= '0';
    if valid = '1' then
      if address >= x″0000" and address < x″4000" then
         if bootover = '0' then
           promsel <= '1';
         else
           shadowsel <= '1';
         end if;
      elsif address >= x″4000" and address < x″4008" then
         periph1 <= '1';
      elsif address >= x″4008" and address < x″4010" then
         periph2 <= '1';
         elsif address >= x″8000" and address < x″C000" then
         sramsel <= '1';
      else address >= x″C000" then
         eesel <= '1';
      end if;
    end if;
  end process;
end behav;
```

## Appendix H. *Warp2* Report File Excerpt for Decoder

```
Information: Macrocell Utilization.

                Description          Used       Max
          ______________________________________________
          | Dedicated Inputs    |     9   |     9   |
          | Clock/Inputs        |     3   |     3   |
          | Enable/Inputs       |     1   |     1   |
          | I/O Macrocells      |     9   |    12   |
          | Buried Macrocells   |     0   |     4   |
          ______________________________________________
                                     22   /    29    = 75   %

Information: Output Logic Product Term Utilization.

           Node#   Output Signal Name   Used    Max
          _________________________________________________
          | 15   |  eesel             |   1  |    9  |
          | 16   |  Used As Input     |   0  |   19  |
          | 17   |  periph2           |   1  |   11  |
          | 18   |  Used As Input     |   0  |   17  |
          | 19   |  shadowsel         |   1  |   13  |
          | 20   |  Used As Input     |   0  |   15  |
          | 23   |  Unused            |   0  |   15  |
          | 24   |  promsel           |   1  |   13  |
          | 25   |  Unused            |   0  |   17  |
          | 26   |  periph1           |   1  |   11  |
          | 27   |  Unused            |   0  |   19  |
          | 28   |  sramsel           |   1  |    9  |
          | 29   |  Unused            |   0  |    1  |
          | 30   |  Unused            |   0  |    1  |
          | 31   |  Unused            |   0  |   13  |
          | 32   |  Unused            |   0  |   17  |
          | 33   |  Unused            |   0  |   11  |
          | 34   |  Unused            |   0  |   19  |
          _________________________________________________
                                          6  / 230    = 2    %
```

## Appendix I. *Warp2* Source Code for UpDown

```
entity updown is
  port(clk0, clk1, clk2: in bit;
    outen, preL, preH, reset: in bit;
    ll: in bit_vector(0 to 7);
    ul6, ul7: in bit;
    countio: inout x01z_vector(0 to 7));
end updown;

use work.bv_math.all;
use work.rtlpkg.all;

architecture archupdown of updown is
signal lower, upper, ul, count: bit_vector(0 to 7);
signal cequ, ceql, dir: bit;
begin
proc1: process
  begin
    wait until clk1 = '1';
      lower <= ll;
  end process;

proc2: process
  begin
    wait until clk2 = '1';
      upper <= ul;
  end process;

proc3: process
  begin
    wait until clk0 = '1';
    if reset = '1' then
      count <= x"00";
    elsif preL = '1' then
      count <= lower;
    elsif preH = '1' then
      count <= upper;
    elsif (dir = '1') then
      count <= inc_bv(count);
    else
      count <= dec_bv(count);
    end if;
  end process;

proc4: process
  begin
    wait until clk0 = '1';
    if count = upper then
      cequ <= '1';
```

## Appendix I. *Warp2* Source Code for UpDown (continued)

```
    else
      cequ <= ’0’;
    end if;
    if count = lower then
      ceql <= ’1’;
    else
      ceql <= ’0’;
    end if;
    if ceql = ’1’ then
      dir <= ’1’;
    elsif cequ = ’1’ then
      dir <= ’0’;
    else
      dir <= dir;
    end if;
  end process;

g1:  for i in 0 to 7 generate
  bidir:if i < 6 generate
    bx:bufoe port map(count(i), outen, countio(i), ul(i));
    end generate;
  trist:if i > 5 generate
    tx:triout port map(count(i), outen, countio(i));
    end generate;
  end generate;

  ul(6) <= ul6;
  ul(7) <= ul7;

end archupdown;
```

## Appendix J. *Warp2* Report File Excerpt for UpDown

```
Information: Macrocell Utilization.

                    Description        Used     Max

                ____________________________________________
                | Dedicated Inputs    |     9  |     9  |
                | Clock/Inputs        |     3  |     3  |
                | Enable/Inputs       |     1  |     1  |
                | I/O Macrocells      |    12  |    12  |
                | Buried Macrocells   |     3  |     4  |
                ____________________________________________
                                          28  /    29    = 96  %

  Information: Output Logic Product Term Utilization.

                  Node#  Output Signal Name  Used   Max

                ________________________________________________
                | 15  |  countio_2_      |   7  |   9  |
                | 16  |  Used As Input   |   0  |  19  |
                | 17  |  countio_0_      |   3  |  11  |
                | 18  |  Used As Input   |   0  |  17  |
                | 19  |  countio_6_      |   7  |  13  |
                | 20  |  countio_4_      |   7  |  15  |
                | 23  |  Used As Input   |   0  |  15  |
                | 24  |  countio_5_      |   7  |  13  |
                | 25  |  countio_3_      |   7  |  17  |
                | 26  |  countio_7_      |   7  |  11  |
                | 27  |  Used As Input   |   0  |  19  |
                | 28  |  countio_1_      |   6  |   9  |
                | 29  |  Unused          |   0  |   1  |
                | 30  |  Unused          |   0  |   1  |
                | 31  |  Unused          |   0  |  13  |
                | 32  |  ceql_BEH_i27..  |  16  |  17  |
                | 33  |  dir             |   2  |  11  |
                | 34  |  cequ            |  16  |  19  |
                ________________________________________________
                                            85  / 230   = 36  %
```

## Appendix K. *Warp2* VHDL Source Code for Serial Decoder

```
entity serial is port(
  clk, reset, data: in bit;
  match: buffer bit;
  lock: buffer bit);
end serial;

use work.int_math.all;
use work.bv_math.all;
architecture archserial of serial is
type states is (state0, state1, state2, state3, state4, state5, state6,
    state7);
signal state, nextstate: states;
signal match_cnt: bit_vector(1 downto 0);
signal bit_cnt: bit_vector(6 downto 0);

begin
fsm:
process   begin
  match <= ’0’;
  case state is
    when state0 =>
      if data = ’1’ and (lock = ’0’ or bit_cnt = ”1111000") then
      nextstate <= state1;
      else
         nextstate <= state0;
      end if;
    when state1 =>
      if data = ’1’ then
         nextstate <= state2;
      else
         nextstate <= state0;
      end if;
    when state2 =>
      if data = ’1’ then
         nextstate <= state3;
      else
         nextstate <= state0;
      end if;
    when state3 =>
      if data = ’0’ then
         nextstate <= state4;
      else
         nextstate <= state3;
      end if;
    when state4 =>
      if data = ’1’ then
         nextstate <= state5;
      else
```

## Appendix K. *Warp2* VHDL Source Code for Serial Decoder (continued)

```
         nextstate <= state0;
      end if;
    when state5 =>
      if data = '0' then
         nextstate <= state6;
      else
         nextstate <= state2;
       end if;
    when state6 =>
       if data = '0' then
         nextstate <= state7;
       else
         nextstate <= state1;
       end if;
    when state7 =>
       if data = '0' then
         nextstate <= state0;
         match <= '1';
       else
         nextstate <= state1;
       end if;
   end case;
end process;

mealy:
process begin
  wait until clk = '1';
    state <= nextstate;
end process;

counter:
process begin
  wait until clk = '1';
  if match = '1' then
    bit_cnt <= "0000000";
  else
    bit_cnt <= inc_bv(bit_cnt);
  end if;
end process;

synchronizer:
process begin
  wait until clk = '1';
  if bit_cnt = "1111111" then
    if match = '1' then
       if match_cnt = "11" then
         lock <= '1';
```

## Appendix K. *Warp2* VHDL Source Code for Serial Decoder (continued)

```
          else
            match_cnt <= inc_bv(match_cnt);
          end if;
       else
          match_cnt <= "00";
          lock <= '0';
       end if;
     end if;
   end process;

   end archserial;
```

## Appendix L. *Warp2* Report File Excerpt for Serial Decoder

```
Information: Macrocell Utilization.

                 Description        Used     Max
             ______________________________________
             | Dedicated Inputs   |    0   |    9  |
             | Clock/Inputs       |    1   |    3  |
             | Enable/Inputs      |    0   |    1  |
             | I/O Macrocells     |   10   |   12  |
             | Buried Macrocells  |    4   |    4  |
             ______________________________________
                                      15   /   29    = 51  %

Information: Output Logic Product Term Utilization.

              Node#  Output Signal Name  Used   Max
             ________________________________________
             | 15  |  lock              |  9  |   9  |
             | 16  |  Unused            |  0  |  19  |
             | 17  |  data              |  5  |  11  |
             | 18  |  bit_cnt_0_        |  1  |  17  |
             | 19  |  serial_0_sta..    |  4  |  13  |
             | 20  |  bit_cnt_2_        |  3  |  15  |
             | 23  |  bit_cnt_1_        |  2  |  15  |
             | 24  |  serial_0_sta..    |  4  |  13  |
             | 25  |  match             |  1  |  17  |
             | 26  |  bit_cnt_3_        |  4  |  11  |
             | 27  |  Unused            |  0  |  19  |
             | 28  |  bit_cnt_4_        |  5  |   9  |
             | 29  |  Unused            |  0  |   1  |
             | 30  |  Unused            |  0  |   1  |
             | 31  |  match_cnt_0_      |  8  |  13  |
             | 32  |  bit_cnt_6_        |  7  |  17  |
             | 33  |  match_cnt_1_      |  9  |  11  |
             | 34  |  bit_cnt_5_        |  6  |  19  |
             ________________________________________
                                         68  / 230    = 29  %
```

Windows is a trademark of Microsoft Corporation.
OpenLook is a trademark of UNIX System Laboratories.
Motif is a trademark of Open Software Foundation, Inc.
*Warp2* is a registered trademark of Cypress Semiconductor Corporation.

# Getting Started Converting .ABL Files to VHDL

## Introduction

This application note is intended to assist *Warp*™ users in converting designs written in DATA I/O's ABEL™7 hardware description language to IEEE 1076 VHDL. It contains several language cross reference tables and many helpful hints. It also includes two real-world designs that have been converted from MACH™ 210–ABEL descriptions to FLASH371–VHDL descriptions.

## VHDL versus ABEL

VHDL is different from ABEL and virtually all other popular hardware description languages in one very significant way. It is an open language based on IEEE standard number 1076.

VHDL is different in other ways, too. VHDL is a high-level language. As such, it is much more powerful than ABEL. For instance, it supports hierarchical design entry, structural (low-level instantiation of components) and behavioral (IF-THEN- ELSE) design entry. VHDL supports process and concurrent process statements. It also supports various types of signals such as integer, real, character, bit, Boolean, physical unit, and any others that a user can define. It supports sequential and concurrent statements, variables, and signals. VHDL supports sub-programs, FOR loops, WHILE loops, arrays, concurrent procedure calls, and more.

Surprisingly, certain aspects of VHDL related to low-level behavioral logic description are very similar to ABEL. In fact, some key words and relational operators are identical or logically similar.

## Conversion Preparation

Preparing to convert an ABEL (.ABL) file should include the following steps:

1. Locate and have at hand one good VHDL language reference book. See the *Warp* documentation for a bibliography.
2. Obtain copies of the *Warp* VHDL design examples titled Basic, Intermediate, and Advanced.
3. Locate two Cypress application notes, one titled "Designing State Machines with *Warp2*® VHDL" and another titled "VHDL Techniques for Optimal Design Fitting." Both are contained in the *Cypress Applications Handbook* (1993).
4. Install the *Warp* VHDL compiler on your hardware platform.

## Conversion Approach

There are many different ways to convert a given design. The same design may be expressed in a number of different ways, all yielding compiled designs with the same functionality. The general approach suggested for converting ABEL (.ABL) files to VHDL (.VHD) consists of five basic steps:

1. Analyze the design and determine:
   a. Which signals are registered and which are not. Group them into two categories.
   b. Which types of design entry the .ABL file includes: state machines, comparators, counters, decoders, multiplexers, adders, multipliers, shift registers, or state_tables.
   c. Whether or not group (set) declarations are used.
   d. Which signals are input, output, I/O, and/or active LOW.
2. Replace as many of the keywords and operators with the corresponding VHDL keywords and operators using your favorite SEARCH and REPLACE text editor and a backup copy of the .ABL or .DOC file.
3. Add the VHDL entity (black box inputs and outputs), architecture (description of the logic circuitry), and process (encapsulates a set of sequential-behavioral functions) statements.
4. Add the proper library USE statements to the file such as USE WORK.CYPRESS.ALL.
5. Iteratively compile the design revising incomplete or incorrectly converted syntax.

Some designs will be much easier to convert than others. The more regular the design the easier it will be to convert. The most efficient and highest level of conversion will be achieved by using the source (.ABL) file, the five steps above, and the cross reference information below.

The simpler approach is to use the .DOC file exclusively. Using the .DOC file works but does have one significant drawback. The .DOC file tends to be verbose. It is verbose because it describes the design at a low level. A converted ABEL (.DOC) design thus results in unnecessarily verbose VHDL. In other words, it results in inefficient code.

When converting using the .DOC file, place all of the registered signals into a process with a WAIT UNTIL CLOCK = '1' and place all of the combinatorial signals outside the process. This avoids the necessity of PROCESS sensitivity lists and IF-THEN-ELSE statements. The converted designs below and all of the *Warp* example designs attempt to describe functions behaviorally and at a higher level. For this reason no low-level design conversion examples are included.

Refer to the following sections and tables for helpful information when converting ABEL .ABL and .DOC descriptions.

## Comments

Comments in ABEL are denoted by the quote symbol ("). Comments in VHDL are denoted by two consecutive dashes ––. For example:

| ABEL | VHDL |
|---|---|
| "Inputs | ––Inputs |
| "Outputs | ––Outputs |

## VHDL-ABEL Special Constant Cross Reference

| ABEL | VHDL | Description |
|---|---|---|
| .C. | requires user definition | Clocked input (0 –>1–> 0) |
| .D. | requires user definition | Clock down edge (1 –> 0) |
| .F. | requires user definition | Floating input or output |
| .K. | requires user definition | Clocked input (1–> 0–> 1) |
| .P. | requires user definition | Register preload |
| .SVn. | requires user definition | Super voltage (2≤n≤9) |
| .U. | requires user definition | Clock up edge (0 – > 1) |
| .X. | requires user definition | Don't care condition |
| .Z. | requires user definition | High impedance |

In VHDL a constant is an object whose value may not be changed. The syntax for declaring a constant in VHDL is:

constant identifier_list : type [ :=expression];

An example of this is:

TYPE stvar is bit_vector (0 to 1);

constant State0 : stvar := "00";

This example declares a constant that is identified by the name State0, is of type stvar, which has been previously defined as a bit_vector subtype of length 2. This constant is given the value 00. Defining a constant of user-defined type called state variable (stvar) is useful when designing state machines in VHDL. See the State Machine section of this application note.

Special constants in ABEL are used for simulation vectors that are included in the source file (.ABL). *Warp* does not provide simulation support directly within the source file. So, the conversion recommendation for files containing simulation vectors is to delete or comment them out of the .VHD file. *Warp* provides simulation separately from the design file (.VHD). Simulation can take one of two forms. The first, functional waveform based design verification using NOVA. Second, full AC timing verification via VIEWSIM and VIEWTRACE. VIEWSIM and VIEWTRACE support both pattern files and waveforms. Both forms of *Warp* VHDL simulation exceed the capabilities of ABEL simulation.

## VHDL-ABEL Dot Extension Cross Reference

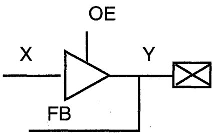
| ABEL | VHDL | Function |
|---|---|---|
| .CLK | none | Clock input to flip-flop |
| .PIN | none | Pin feedback |
| .FB | none | Register feedback |
| .D | none | D-type flip-flop |
| .J | none | J input to JK flip-flop |
| .K | none | K input to JK flip-flop |
| .S | none | S input to SR flip-flop |
| .R | none | R input to SR flip-flop |
| .T | none | T-type flip-flop |
| .Q | none | Register feedback |
| .PR | none | Register preset |
| .RE | none | Register reset |
| .SP | none | Synchronous reg preset |
| .SR | none | Synchronous reg reset |
| .LE | none | Latch-enable input |
| .LH | none | Latch-enable input (HIGH) |
| .LD | none | Register load input |
| .CE | none | Clock enable input |
| .AP | none | Asynchronous preset |
| .AR | none | Asynchronous reset |
| .OE | none | Three-state output enable |
| .CLR | none | Synchronous clear |

## VHDL-ABEL Dot Extension Cross Reference (continued)

| ABEL | VHDL | Function |
|---|---|---|
| .ACLR | none | Asynchronous clear |
| .SET | none | Synchronous set |
| .ASET | none | Asynchronous clear |
| .COM | none | Combinational feedback |
| .FC | none | Flip-flop mode control |

Although VHDL is capable of supporting these constructs, it directly does not. Indirectly, through behavioral description, structural description, and an intelligent compiler, all of these constructs are supported. For example, *Warp* does provide predefined register transfer level (RTL) components (such as D- and T-type flip-flops). These RTL components can be structurally instantiated to model the ABEL extensions listed above. Specifically, to model the .OE ABEL extension, use an RTL component called *bufoe*. The syntax and port map (inputs and outputs) of a *bufoe* component is the following:

Label: BUFOE PORT MAP(X, OE, Y, FB)



X, OE, Y, and FB are sample signal names. The position each one occupies in the port map is the mechanism VHDL uses to correctly connect the signal names to the actual component in the architecture of the target device.

The behavioral equivalent of structurally instantiating a bidirectional buffer is called a behavioral three-state.

If the ABEL description equation is written in .T (T-type) flip-flop style, the recommended conversion method is to rewrite the equations as D-type (XOR the original equation with the flip-flop output signal name) and let *Warp* optimize the equation for either D- or T-type. See the real-world design conversion example in Appendix A called FLAGCTLR.

## IS_TYPE Attribute Cross Reference

| ABEL | VHDL | Description |
|---|---|---|
| 'buffer' | none, may use RTL – buf | Macrocell has no inverter between reg and pin |
| 'com' | none, may use RTL – buf | Signal is combinatorial |
| 'invert' | none, NOT RHS of equation | Macrocell has an inverter between reg and pin |
| 'neg' | none, NOT signal in equation | Complement Sum of Products |
| 'pos' | none | Do not Complement Sum of Products |
| 'reg' | none, place sig in process | Generic flip-flop |
| 'reg_D' | none, may use RTL – dff | D-type flip-flop |
| 'reg_T' | none, may use RTL – tff | T-type flip-flop |
| 'reg_SR' | none, may use RTL – srff | SR-type flip-flop |
| 'reg_JK' | none, may use RTL – jkff | JK-type flip-flop |
| 'reg_G' | none | D-flip-flop w/gated clock |
| 'xor' | none, may use RTL – xbuf | Target architecture has XOR |

## Operator Cross Reference

| ABEL | Order of Precedence | VHDL | Order of Precedence | Operation |
|---|---|---|---|---|
| ! | 1 | NOT | Context dependent | NOT (invert) |
| & | 2 | AND | 1 | AND |
| * | 2 | * | 5 | Multiplication |
| / | 2 | / | 5 | Division |
| % | 2 | mod | 5 | Modulus |
| << | 2 | sll |  | Shift left |
| >> | 2 | slr |  | Shift right |
| + | 3 | + | 3 | Arithmetic addition |
| – | 3 | – | 3 | Arithmetic subtr. |
| $ | 3 | XOR | 1 | XOR |
| !$ | 3 | NOT XOR |  | XNOR |
| # | 3 | OR | 1 | OR |
| == | 4 | = | 2 | Equal |
| != | 4 | /= | 2 | Not equal |
| < | 4 | < | 2 | Less than |
| <= | 4 | <= | 2 | Less than or equal |
| > | 4 | > | 2 | Greater than |
| >= | 4 | >= | 2 | Greater or equal |
| none |  | NAND | 1 | NAND |
| none |  | NOR | 1 | NOR |
| none |  | & | 3 | Concatenation |
| none |  | rem | 5 | Remainder |
| none |  | abs | 6 | Absolute Value |
| none |  | ** | 6 | Exponentiation |
| ? |  | + | 4 | Sign |
| ? |  | – | 4 | Sign |
| = or := |  | <= |  | Signal assignment |
| none |  | := |  | Variable assignment |
| = |  | <= |  | Comb. assignment |
| := |  | <= |  | Reg. assignment |
| none |  | => |  | Association |

## Keyword (Statement) Cross Reference

| ABEL Keyword | VHDL Equivalent |
|---|---|
| CASE | CASE |
| DECLARATIONS | Note 1 |
| DEVICE | ATTRIBUTE PART_NAME IS ... |
| ELSE | ELSE |
| ENABLE (Obsolete) | none |
| ENDCASE | END CASE |
| ENDWITH | Note 2 |
| EQUATIONS | Note 3 |
| FLAG (Obsolete) | none |
| FUSES | Note 4 |
| GOTO | EXIT - Note 5 |
| IF | IF |
| IN (Obsolete) | none |
| ISTYPE | Note 6 |
| LIBRARY | USE |
| MACRO | FUNCTION - Note 7 |
| MODULE | FUNCTION - Note 8 |
| NODE | SIGNAL |
| OPTIONS | Note 9 |
| PIN | Note 10 |
| PROPERTY | none |
| STATE | Note 11 |
| STATE_DIAGRAM | Note 11 |
| TEST_VECTORS | Note 12 |
| THEN | THEN |
| TITLE | Note 13 |
| TRACE | Note 14 |
| TRUTH_TABLE | Note 15 |
| WHEN | WHEN |
| WITH | Note 16 |
| ASYNC_RESET | Note 17 |
| SYNC_RESET | Note 17 |
| STATE_REGISTER | Note 18 |
| XOR_FACTORS | Note 19 |

VHDL does not use the term keyword. Analogous to ABEL's use of the term keyword, VHDL uses the terms statement, reserved word, and identifier.

**Notes**

1. There is not a DECLARATIONS keyword in VHDL. However, the DECLARATIONS keyword is analogous to declaring an ENTITY in VHDL. Within the ENTITY construct inputs, outputs, and I/Os are declared with appropriate mode and type. Mode loosely refers to the pin drive direction, which can be IN, OUT, or INOUT. Refer to your language reference book for a more formal definition of the terms mode, IN, OUT, and INOUT.
2. ENDWITH is part of the WITH-ENDWITH transition structure used with IF-THEN-ELSE or CASE keywords. In VHDL conditional transition is handled via an IF-THEN-ELSE or CASE statement within a PROCESS. The process statement may or may not use a sensitivity list and instead use a WAIT UNTIL (condition) statement. See the application note titled “Designing State Machines with *Warp2* VHDL.”
3. Equations in VHDL are listed within an architecture statement.
4. VHDL and *Warp* do not provide predefined fuse-level program specification.

5. VHDL does not have a GOTO keyword (statement). It provides an EXIT keyword for stopping execution of loops entirely.
6. The IS_TYPE keyword (statement) defines attributes and/or characteristics of pins and nodes. VHDL provides these attributes through behavioral specification. Additionally, *Warp* provides a set of predefined attributes and VHDL provides a mechanism for declaring new attributes. See the attribute table below.
7. VHDL provides function call and return capability. MACRO is more of a low-level substitution technique such that, wherever the MACRO_id occurs, the text associated with that macro will be substituted.
8. The MODULE ... END statement(s) are source file requirements in ABEL. In VHDL the ENTITY-ARCHITECTURE pair are the basic source file requirements. Both the ENTITY and ARCHITECTURE constructs require an END statement.
9. OPTION is a string of processing options that affect the way in which the ABEL source file is processed by the language processor. The analogous control in VHDL is not done in the source file. It in fact is not part of the VHDL language. It is simply a menu of compiler options that are set when using *Warp* to synthesize the design.
10. PIN is used to declare input and output signals that must be available on a device I/O pin. The analogous PIN specification is implied in VHDL via the port map list in the ENTITY construct. All signal names listed in the entity port map are input, output, and I/Os of the entity.
11. See application note titled “Describing State Machines with *Warp*2 VHDL.”
12. Test vectors are not directly supported by VHDL. However, both behavioral simulation and full AC timing simulation are available for design verification.
13. The TITLE statement is used to give an ABEL MODULE a title that will appear as a header in both the programmer load file (the JEDEC file) and the documentation file. When compiling a VHDL design using *Warp*, the filename of the VHDL (.VHD) design file is passed through to the programmer load file (.JED) as well as the documentation file (.RPT).
14. The TRACE statement controls the display features of ABEL's simulator. There is not a similar keyword in VHDL because simulation is separate from the source file description.
15. The TRUTH_TABLE keyword is used in ABEL to specify outputs as functions of different input combinations in a tabular form. VHDL does not directly provide a TRUTH_TABLE keyword. However, the common library (directory) included in *Warp*, contains a FUNCTION called TTF. TTF is a predefined truth table function that can be used for both combinatorial truth tables and for state transition tables. See the application note titled “Describing State Machines in *Warp*2 VHDL” regarding use of the TTF function.
16. WITH is part of the WITH-ENDWITH transition structure used with IF-THEN-ELSE or CASE statements. In VHDL, conditional transition is handled via an IF-THEN-ELSE or CASE statement within a PROCESS. The process statement may use a sensitivity list or may include a WAIT UNTIL (clock = ‘1’) statement.
17. ASYNC_RESET and SYNC_RESET statements are used in Symbolic State descriptions. They symbolically specify what state the machine should asynchronously or synchronously reset to, based upon a signal or an expression. In VHDL, asynchronous and synchronous resets are best handled from a behavioral perspective the Resets and Presets section of this note for more detail.
18. STATE_REGISTER is a mechanism whereby specific states of a machine can be identified symbolically. See the State Machine section of this note for more detail.
19. XOR_FACTORS is a keyword that is useful for factoring logic designs that target a device which features XOR gates. There is not an analogous keyword in VHDL. However, the functional aspect of this keyword is part of the *Warp* Compiler Option menu. For more details see the *Warp* Compiler Options Documentation.

## Predefined Attributes Supported by *Warp*

| | |
|---|---|
| Value Attributes | 'Left<br>'Right<br>'High<br>'Low<br>'Length |
| Function Attributes (types) | 'Pos<br>'Val<br>'Leftof<br>'Succ<br>'Rightof<br>'Pred |
| Function Attributes (objects) | 'Left<br>'Right<br>'High<br>'Low<br>'Length |
| Function Attributes (signals) | 'Event |
| Type Attributes | 'Base |
| Range Attributes | 'Range<br>'Reverse_range |

Other user-defined attributes include: Enum-encoding, Flip-flop-type, Order_code, Part_name, Pin_numbers, Polarity, State_encoding, and State_Variable. See *Warp* documentation for details.

## Number Representations

| ABEL | VHDL | Radix |
|---|---|---|
| ^b | b“ ” or “ ” or ‘ ’(default)[20] | Binary |
| ^o | o“ ” | Octal |
| ^d (default) | Note 21 | Decimal |
| ^h | x“ ” | Hexadecimal |

**Notes**

20.The default number representation in ABEL is decimal. The default number representation in VHDL is binary.

21.The default number representation in VHDL is binary. Decimal representations of numbers in VHDL require the user to define a signal or variable with type integer or use an integer number and then type-convert it to bit_vector. This is easier than it sounds. In the common library directory within *Warp* there is a file called LIBINT.VHD that contains a predefined function called i2bv. This function takes an integer and returns a bit_vector. So, using a decimal number is not too difficult, but one must know that an integer must be used and then type-converted to bit_vector.

For example:

| ABEL syntax | VHDL syntax | Description |
|---|---|---|
| ^b1 | ’1’ | binary 1 |
| ^b0 | ’0’ | binary 0 |
| ^b101010000 | “10101000" | binary 10101000 |
| ^hF | x“F” | hex F |
| ^hF1 | x“F1" | hex F1 |
| ^hAAA | x“AAA” | hex AAA |
| ^oF0F0 | o“F0F0" | octal F0F0 |
| ^d23 | i2bv(23,5) | decimal 23 |
| ^d99 | i2bv(99,7) | decimal 99 |

## Polarity Conventions

VHDL does not know whether a signal name should be interpreted as an active HIGH or an active LOW. Therefore, a signal named SHIFT4 will be interpreted logically the same as one named L_SHIFT4, and as one named SHIFT4_NOT. In other words, the behavioral equations must test with the proper level and assert with the desired level.

During logic synthesis and optimization, the software may determine that by flipping the polarity of a function the logic required will be optimized.

## Identifiers

VHDL is not case sensitive, so a signal named SHIFT0 is identical to one named sHIFT0.

## Resets and Presets

Although there are a variety of ways to specify a reset or preset, the best method is behavioral specification. If the reset or preset is asynchronous, use the following:

Place an IF-THEN-ELSIF-ENDIF inside a process with a (CLK’EVENT and CLK=’1’) placed as the condition for the ELSIF. In the first IF, place your reset and preset condition test and your signal assignments. In other words, the first part of the IF contains the asynchronous or combinatorial logic description and the second part, the ELSIF, contains the clocked logic description. In the process statement use a sensitivity list that includes the clock, and reset/preset for the design. Don’t forget that statements in a process are considered sequential and are only updated upon changes in signals listed in the sensitivity list. See the basic example called COUNTER2.VHD and the real-world converted design example called FLAGCTLR below.

If the reset or preset is synchronous, place the condition inside the clocked portion of the IF-THEN-ELSIF-ENDIF mentioned above and perform the appropriate signal assignments.

This methodology ensures that behavioral operation is preserved and no device specific attributes are required.

## Groups

ABEL allows declaration of groups or sets. Sets are groupings of signals. For example a bus is a set of signals. To create a set of signals in VHDL use the bit_vector type declaration. To perform Boolean operations on these new sets use IF-THEN-ELSE and FOR LOOPs to index the individual elements. See the special type conversion function and the real-world examples below.

## Special VHDL Type Conversion Function (Advanced)

VHDL is a strongly typed language. ABEL on the other hand is not a strongly typed language. ABEL allows a user to mix Boolean operations with relational operations on sets. To concisely convert ABEL equations that contain relational operations on sets (converted to VHDL type BIT_VECTOR) combined with Boolean operations on signals (converted to VHDL type BIT), use the following type-conversion function. All equations requiring this type-conversion function call can be modified easily with a SEARCH and REPLACE text editor.

```
----------------------------- cut here ------------------------------------

FUNCTION frbl_to_b (in1:Boolean)      RETURN bit IS
BEGIN
IF (in1=TRUE) THEN
     RETURN '1';
ELSE
     RETURN '0';END IF;
END frbl_to_b;
--  This type conversion function converts a signal or relational operation
--  result from type BOOLEAN to type BIT.  A Boolean can have a value of
--  either 'TRUE' or 'FALSE'.  A bit can have a value of either '0' or '1'.
---------------------------- cut here -----------------------------------
```

For example if you had an equation in ABEL such as:

```
ramwr = !addren & ba16 & !write & ((addr ==^h210)
     # (addr==^h212)
     # (addr==^h214)
     # (addr==^h216));
```

Where addr is a set of 16 address bits,

This equation could be converted to VHDL in at least two ways:

```
ramwr <= not addren and ba16 and not write and (fr_bl_to_b(addr =x"210")
     or fr_bl_to_b(addr=x"212")
     or fr_bl_to_b(addr=x"214")
     or fr_bl_to_b(addr=x"216"));
```

OR,

```
ramwr <= not addren and ba15 and not write and(
      (not addr(11) and not addr(10) and addr(9) and not addr(8) and not addr(7)
       and not addr(6) and not addr(5) and addr(4) and not addr(3) and not addr(2)
       not addr(1) and not addr(0))
   OR  (not addr(11) and not addr(10) and addr(9) and not addr(8) and not
       addr(7) and not addr(6) and not addr(5) and addr(4) and not addr(3) and not
       addr(2) and addr(1) and not addr(0))
   OR  (not addr(11) and not addr(10) and addr(9) and not addr(8) and not
      addr(7) and not addr(6) and not addr(5) and addr(4) and not addr(3) and
       addr(2) and not addr(1) and not addr(0))
   OR (not addr(11) and not addr(10) and addr(9) and not addr(8) and not addr(7)
      and not addr(6) and not addr(5) and addr(4) and not addr(3) and addr(2) and
      addr(1) and not addr(0)));
```

This example assumes all of the signals from the ABEL equations are converted to signals of type BIT except the set called 'addr', which is converted to type BIT_VECTOR.

This special type-conversion function has a obvious advantage and is well suited for use in converting descriptions to VHDL. By no means is it a requirement that descriptions use this function. It should be used for one reason only, to make a VHDL description concise. See the real-world design example in Appendix A called FLAGCTLR.

## State Machines

See the *Warp* design examples titled intermediate TRAFFIC.VHD, intermediate DRINK.VHD and advanced TTF.VHD. See the application note titled "Describing State Machines in *Warp2* VHDL." Also refer to pitfall numbers five and seven below.

## Decoders

See the *Warp* design example titled basic DECODER.VHD and the special type-conversion function above.

## Comparators

See the *Warp* design examples titled intermediate COMPARE.VHD and COMPARE2.VHD.

## Counters

See the *Warp* design examples titled basic COUNTER.VHD, basic COUNTER2.VHD, intermediate COUNTER3.VHD, advanced COUNTER4.VHD, and advanced COUNTER5.VHD.

## Multiplexers

Use the truth table function that is shown in the application note titled "Describing State Machines in *Warp2* VHDL" or create a multiplexer using Boolean equations.

## Shift Registers

See the *Warp* design example titled advanced SHIFTN.VHD. This example illustrates the use of the GENERATE statement.

## Adders

See the *Warp* design examples titled basic ADDER1.VHD and basic ADDER2.VHD.

## Repetitive Logic

The VHDL GENERATE statement lends itself to regular or repetitive logic structures. For example, n-bit registers, n-bit counters, n-bit shift registers, n-bit multiplexers, n-bit adders, and n-bit comparators may be concisely described by using the GENERATE statement. See the *Warp* design examples titled advanced SHIFTN.VHD and advanced COUNTER4.VHD.

## Pitfalls

There are potential pitfalls. Some of the common mistakes made during conversion are:

1. Incorrect order of precedence of operators. For instance, all of the logical operators in VHDL have the same level of precedence. In other words, an equation that has both AND and OR operators requires parenthesis around the

Ended terms for proper logic synthesis. Refer to the cross reference and order of precedence table above.

2. Incomplete separation of clocked signals from combinatorial signals. Two simple ways to ensure proper logic synthesis of clocked signals and combinatorial signals are:
   a. Use a process for all signals, but use an IF-THEN-ELSIF-ENDIF within the process that groups all combinatorial signals under the IF, and groups all registered signals under the ELSIF. See the real-world design example in Appendix A called FLAGCTLR.
   b. Place all registered signals within a process (using a WAIT UNTIL CLOCK ='1') and place all combinatorial signals outside the process.
3. Using loops and variables outside of a process. VHDL requires that loops and variables be used inside a process. If there is more than one process, signals communicate between processes.
4. Using the incorrect mode for either output or bidirectional signals. Refer to your language reference book for a formal definition of mode.
5. Incomplete state specification for state machines. When designing a state machine, you **MUST** do one of the following:
   a. Specify all output values in each state of the machine.

      or
   b. Specify default values for all outputs at the beginning of the process.

   The reason for this has to do with the way a process works. Each time a process is run (i.e., a clock event has occurred) the outputs that are specified in the particular pass through the process are updated. If a branch exists within the states of the machine that allows a pass through the process with one or more outputs not assigned a value, the logic synthesis engine either (a) assumes that the last statement for an unassigned output is valid and should be latched, or (b) that it is allowed to change with the clock. In other words, it is legal in VHDL to not specify all output values in each state of the machine, or not specify default values for all outputs at the beginning of the process, or not specify either one. If this subtle detail is overlooked, the design will compile and appear to synthesize successfully, but functional operation may not be correct. It is also possible that the logic synthesized will not be minimal. In other words, use defaults or specify the value of all outputs within each state of the machine.
6. Incorrect set or reset operation found in simulation. Polarity optimization settings used during logic synthesis and fitting can cause set and or reset operations to appear to operate inconsistently. During logic synthesis and fitting, the fitter can decide, by flipping the polarity of a function, the logic required will be minimized. This can have an adverse effect on the user selection of set or reset. (Note this pitfall only applies to 22V10s and FLASH370 where the polarity inversion is located between the output of the register and the pin.) See the polarity attribute in the *Warp* documentation for more details.
7. Failure to close, or complete, IF-THEN-ELSE-ENDIF and CASE statements. In other words, design descriptions that contain an IF must contain an ELSE, and descriptions containing a CASE-WHEN (condition), must contain a WHEN-OTHERS statement. This is required so that unnecessary implicit memory elements are not synthesized. See the application note titled "VHDL Techniques for Optimal Design Fitting" for more information.

## Logic Synthesis

Proper logic synthesis is the goal of conversion. If the converted design compiles and synthesizes without errors, but the logic equations in the report file are not as expected (or simulation results are not as desired) consult the pitfalls section above. Also, consult your *Warp* – GALAXY compiler options documentation and *Warp* – NOVA user's guide. If all else fails, contact your local Cypress field application engineer.

## Real-World Converted Designs

The designs in Appendix A originally were intended to fit into MACH110s. However, due to product term and internal fanout requirements, MACH210s were required. The designs were later converted to FLASH371s. Consult your Cypress data book for more information on the CY7C371's architecture.

## Summary

Any design that has been described in Data I/O's ABEL language can be converted to VHDL. From an overall capability perspective, VHDL can be considered a superset of ABEL. Two designs documented in Appendix A were successfully converted using the cross reference tables and helpful hints contained within this application note.

**Note**: All references to components and functions made in this application note can be accessed only from "lib35" library included in *Warp.* Warp release 3.5 library components are IEEE 1064 compatible and are defined in terms of bit and bit_vector. *Warp* release 4.0 or higher supports IEEE 1164 VHDL. The corresponding IEEE1164 VHDL compatible functions and components are available in "lib40" library in Warp. You can easily switch between lib35 and lib40 libraries by changing the *Warp* library pointer. However you cannot mix lib35 library components with lib40 components in the same project.

## Appendix A. Real-World Converted Designs

```
----------------------------------- cut here -----------------------------------

  Module FLAGCTLR
  Title 'Flag Controller 1 - Uxx_xx
  Revision 01'

  "ALGORITHM
  "
  "

  FLAGCTLR           device 'mach210a';

  "Inputs:

      R_40MHZ      pin ; "
      H_FEP_S0     pin ; "
      H_FEP_S1     pin ; "
      H_FEP_S2     pin ; "
      H_FEP_S3     pin ; "
      H_FEP_SET    pin ; "
      L_FEP_WE     pin ; "
      H_PPA_S0     pin ; "
      H_PPA_S1     pin ; "
      H_PPA_S2     pin ; "
      H_PPA_S3     pin ; "
      H_PPA_SET    pin ; "
      L_PPA_WE     pin ; "
      H_PPB_S0     pin ; "
      H_PPB_S1     pin ; "
      H_PPB_S2     pin ; "
      H_PPB_S3     pin ; "
      H_PPB_SET    pin ; "
      L_PPB_WE     pin ; "
      L_RESET      pin ; "

  "Outputs:

      H_FA0     pin  istype 'reg,buffer'; "
      H_FA1     pin  istype 'reg,buffer'; "
      H_FA2     pin  istype 'reg,buffer'; "
      H_FA3     pin  istype 'reg,buffer'; "
      H_FA4     pin  istype 'reg,buffer'; "
      H_FA5     pin  istype 'reg,buffer'; "
      H_FA6     pin  istype 'reg,buffer'; "
      H_FA7     pin  istype 'reg,buffer'; "

      H_FB0     pin  istype 'reg,buffer'; "
      H_FB1     pin  istype 'reg,buffer'; "
      H_FB2     pin  istype 'reg,buffer'; "
      H_FB3     pin  istype 'reg,buffer'; "
      H_FB4     pin  istype 'reg,buffer'; "

      H_AB0     pin  istype 'reg,buffer'; "
      H_AB1     pin  istype 'reg,buffer'; "
      H_AB2     pin  istype 'reg,buffer'; "
      H_AB3     pin  istype 'reg,buffer'; "
      H_AB4     pin  istype 'reg,buffer'; "
```

## Appendix A. Real-World Converted Designs (continued)

```
Declarations

    X = .X.;
    C = .C.;
    Z = .Z.;

    FA =     [H_FA7,H_FA6,H_FA5,H_FA4,
              H_FA3,H_FA2,H_FA1,H_FA0];

    FB =     [H_FB4,H_FB3,H_FB2,H_FB1,H_FB0];
    AB =     [H_AB4,H_AB3,H_AB2,H_AB1,H_AB0];
    PPA_SEL  = [H_PPA_S3,H_PPA_S2,H_PPA_S1,H_PPA_S0];
    PPB_SEL  = [H_PPB_S3,H_PPB_S2,H_PPB_S1,H_PPB_S0];
    FEP_SEL  = [H_FEP_S3,H_FEP_S2,H_FEP_S1,H_FEP_S0];

Equations

  FA.CLK = R_40MHZ;
  FB.CLK = R_40MHZ;
  AB.CLK = R_40MHZ;

  FA.RE  = !L_RESET;
  FB.RE  = !L_RESET;
  AB.RE  = !L_RESET;


  H_FA0.T = (!H_FA0.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h0)
           #  H_FA0.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h0)
           # !H_FA0.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h0)
           #  H_FA0.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h0));

  H_FA1.T = (!H_FA1.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h1)
           #  H_FA1.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h1)
           # !H_FA1.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h1)
           #  H_FA1.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h1));


  H_FA2.T = (!H_FA2.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h2)
           #  H_FA2.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h2)
           # !H_FA2.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h2)
           #  H_FA2.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h2));

  H_FA3.T = (!H_FA3.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h3)
           #  H_FA3.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h3)
           # !H_FA3.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h3)
           #  H_FA3.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h3));


  H_FA4.T = (!H_FA4.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h4)
           #  H_FA4.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h4)
           # !H_FA4.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h4)
           #  H_FA4.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h4));

  H_FA5.T = (!H_FA5.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h5)
           #  H_FA5.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h5)
           # !H_FA5.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h5)
           #  H_FA5.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h5));
```

## Appendix A. Real-World Converted Designs (continued)

```
H_FA6.T = (!H_FA6.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h6)
         #  H_FA6.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h6)
         # !H_FA6.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h6)
         #  H_FA6.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h6));

H_FA7.T = (!H_FA7.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h7)
         #  H_FA7.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h7)
         # !H_FA7.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h7)
         #  H_FA7.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h7));


H_FB0.T = (!H_FB0.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h0)
         #  H_FB0.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h0)
         # !H_FB0.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h8)
         #  H_FB0.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h8));

H_FB1.T = (!H_FB1.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h1)
         #  H_FB1.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h1)
         # !H_FB1.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h9)
         #  H_FB1.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^h9));


H_FB2.T = (!H_FB2.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h2)
         #  H_FB2.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h2)
         # !H_FB2.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^ha)
         #  H_FB2.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^ha));

H_FB3.T = (!H_FB3.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h3)
         #  H_FB3.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h3)
         # !H_FB3.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^hb)
         #  H_FB3.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^hb));


H_AB0.T = (!H_AB0.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h8)
         #  H_AB0.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h8)
         # !H_AB0.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h8)
         #  H_AB0.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h8));

H_AB1.T = (!H_AB1.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h9)
         #  H_AB1.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h9)
         # !H_AB1.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h9)
         #  H_AB1.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^h9));

H_AB2.T = (!H_AB2.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^ha)
         #  H_AB2.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^ha)
         # !H_AB2.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^ha)
         #  H_AB2.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^ha));

H_AB3.T = (!H_AB3.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^hb)
         #  H_AB3.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^hb)
         # !H_AB3.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^hb)
         #  H_AB3.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^hb));

H_FB4.T = (!H_FB4.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h4)
         #  H_FB4.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^h4)
         # !H_FB4.Q &  H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^hc)
         #  H_FB4.Q & !H_FEP_SET & !L_FEP_WE & (FEP_SEL == ^hc));
```

## Appendix A. Real-World Converted Designs (continued)

```
  H_AB4.T = (!H_AB4.Q &  H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^hc)
          #  H_AB4.Q & !H_PPB_SET & !L_PPB_WE & (PPB_SEL == ^hc)
          # !H_AB4.Q &  H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^hc)
          #  H_AB4.Q & !H_PPA_SET & !L_PPA_WE & (PPA_SEL == ^hc));

test_vectors ([R_40MHZ,L_RESET,
               L_FEP_WE, FEP_SEL, H_FEP_SET,
               L_PPA_WE, PPA_SEL, H_PPA_SET,
               L_PPB_WE, PPB_SEL, H_PPB_SET]
          -> [H_FA7, H_FA6, H_FA5, H_FA4, H_FA3, H_FA2, H_FA1, H_FA0,
               H_FB4, H_FB3, H_FB2, H_FB1, H_FB0,
               H_AB4, H_AB3, H_AB2, H_AB1, H_AB0])
[C,1,1,^h0,0,1,^h1,0,1,^h0,0]->[X,X,X,X,X,X,X,X, X,X,X,X,X, X,X,X,X,X];
[C,1,0,^h0,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h1,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h2,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h3,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h4,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h5,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h6,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h7,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h8,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h9,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^hA,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^hB,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^hC,0,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h0,1,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,0,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h1,1,1,^h1,0,1,^h0,0]->[0,0,0,0,0,0,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h2,1,1,^h1,0,1,^h0,0]->[0,0,0,0,0,1,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h3,1,1,^h1,0,1,^h0,0]->[0,0,0,0,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h4,1,1,^h1,0,1,^h0,0]->[0,0,0,1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h5,1,1,^h1,0,1,^h0,0]->[0,0,1,1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h6,1,1,^h1,0,1,^h0,0]->[0,1,1,1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h7,1,1,^h1,0,1,^h0,0]->[1,1,1,1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0];
[C,1,0,^h8,1,1,^h1,0,1,^h0,0]->[1,1,1,1,1,1,1,1, 0,0,0,0,1, 0,0,0,0,0];
[C,1,0,^h9,1,1,^h1,0,1,^h0,0]->[1,1,1,1,1,1,1,1, 0,0,0,1,1, 0,0,0,0,0];
[C,1,0,^hA,1,1,^h1,0,1,^h0,0]->[1,1,1,1,1,1,1,1, 0,0,1,1,1, 0,0,0,0,0];
[C,1,0,^hB,1,1,^h1,0,1,^h0,0]->[1,1,1,1,1,1,1,1, 0,1,1,1,1, 0,0,0,0,0];
[C,1,0,^hC,1,1,^h1,0,1,^h0,0]->[1,1,1,1,1,1,1,1, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h0,0,1,^h0,0]->[1,1,1,1,1,1,1,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h1,0,1,^h0,0]->[1,1,1,1,1,1,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h2,0,1,^h0,0]->[1,1,1,1,1,0,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h3,0,1,^h0,0]->[1,1,1,1,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h4,0,1,^h0,0]->[1,1,1,0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h5,0,1,^h0,0]->[1,1,0,0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h6,0,1,^h0,0]->[1,0,0,0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,0,^h7,0,1,^h0,0]->[0,0,0,0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h0,0]->[0,0,0,0,0,0,0,0, 1,1,1,1,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h1,0]->[0,0,0,0,0,0,0,0, 1,1,1,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h2,0]->[0,0,0,0,0,0,0,0, 1,1,0,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h3,0]->[0,0,0,0,0,0,0,0, 1,0,0,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h4,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h8,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^h9,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^hA,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
[C,1,1,^h7,1,1,^h7,0,0,^hB,0]->[0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
```

## Appendix A. Real-World Converted Designs (continued)

```
   [C,1,1,^h7,1,1,^h7,0,0,^hC,0]->[0,0,0,0,0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0];
   [C,1,1,^h7,1,1,^h7,0,0,^hC,1]->[0,0,0,0,0,0,0,0,0, 0,0,0,0,0, 1,0,0,0,0];
   [C,1,1,^h7,1,1,^h7,0,0,^hB,1]->[0,0,0,0,0,0,0,0,0, 0,0,0,0,0, 1,1,0,0,0];
   [C,1,1,^h7,1,1,^h7,0,0,^hA,1]->[0,0,0,0,0,0,0,0,0, 0,0,0,0,0, 1,1,1,0,0];
   [C,1,1,^h7,1,1,^h7,0,0,^h9,1]->[0,0,0,0,0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,0];
   [C,1,1,^h7,1,1,^h7,0,0,^h8,1]->[0,0,0,0,0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1];
   [C,1,1,^h7,1,1,^h7,1,0,^h4,1]->[0,0,0,0,0,0,0,0,0, 1,0,0,0,0, 1,1,1,1,1];
   [C,1,1,^h7,1,1,^h7,1,0,^h3,1]->[0,0,0,0,0,0,0,0,0, 1,1,0,0,0, 1,1,1,1,1];
   [C,1,1,^h7,1,1,^h7,1,0,^h2,1]->[0,0,0,0,0,0,0,0,0, 1,1,1,0,0, 1,1,1,1,1];
   [C,1,1,^h7,1,1,^h7,1,0,^h1,1]->[0,0,0,0,0,0,0,0,0, 1,1,1,1,0, 1,1,1,1,1];
   [C,1,1,^h7,1,1,^h7,1,0,^h0,1]->[0,0,0,0,0,0,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h7,1,1,^h0,1]->[1,0,0,0,0,0,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h6,1,1,^h0,1]->[1,1,0,0,0,0,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h5,1,1,^h0,1]->[1,1,1,0,0,0,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h4,1,1,^h0,1]->[1,1,1,1,0,0,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h3,1,1,^h0,1]->[1,1,1,1,1,0,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h2,1,1,^h0,1]->[1,1,1,1,1,1,0,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h1,1,1,^h0,1]->[1,1,1,1,1,1,1,0,0, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h0,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,0,^h8,0,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,1,0];
   [C,1,1,^h7,1,0,^h9,0,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,0,0];
   [C,1,1,^h7,1,0,^hA,0,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,0,0,0];
   [C,1,1,^h7,1,0,^hB,0,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,0,0,0,0];
   [C,1,1,^h7,1,0,^hC,0,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 0,0,0,0,0];
   [C,1,1,^h7,1,0,^hC,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,0,0,0,0];
   [C,1,1,^h7,1,0,^hB,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,0,0,0];
   [C,1,1,^h7,1,0,^hA,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,0,0];
   [C,1,1,^h7,1,0,^h9,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,1,0];
   [C,1,1,^h7,1,0,^h8,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,1,1];
   [C,1,1,^h7,1,1,^h0,1,1,^h0,1]->[1,1,1,1,1,1,1,1,1, 1,1,1,1,1, 1,1,1,1,1];

END FLAGCTLR;

------------------------------------ cut here ----------------------------------

--  Converted to IEEE 1076 VHDL

--  Module FLAGCTLR
```

## Appendix A. Real-World Converted Designs (continued)

```
--  Title ’Flag Controller 1 - Uxx_xx
--  Revision 01’

use work.cypress.all;
use work.rtlpkg.all;
use work.int_math.all;

ENTITY FLAGCTLR IS PORT(
  R_40MHZ,H_FEP_SET,L_FEP_WE,H_PPA_SET,
  L_PPA_WE,H_PPB_SET,L_PPB_WE,L_RESET    : IN BIT;
  PPA_SEL,PPB_SEL,FEP_SEL                : IN BIT_VECTOR (3 downto 0);
  FA                                     : INOUT BIT_VECTOR (7 downto 0);
  FB,AB                                  : INOUT BIT_VECTOR (4 downto 0));
attribute part_name of eventflg: entity is ″c371";
END FLAGCTLR;

ARCHITECTURE CONVERTED_ABL OF FLAGCTLR IS

FUNCTION frbl_to_b(in1:Boolean)     RETURN bit IS
BEGIN
IF (in1=true) THEN
     RETURN ’1’;
ELSE
     RETURN ’0’;
END IF;
END frbl_to_b;
--  This type conversion function converts a signal or relational operation
--  result from type BOOLEAN to type BIT.  A Boolean can have a value of either
-- ’TRUE’ or’FALSE’.  A bit can have a value of either ’0’ or ’1’.

BEGIN
PROCESS (R_40MHZ, L_RESET)
BEGIN

IF (L_RESET =’0’) THEN

      FOR i IN 0 TO 4 LOOP
           FA(i) <= ’0’; FB(i) <= ’0’; AB(i) <= ’0’;
      END LOOP;

      FOR i IN 5 TO 7 LOOP
           FA(i) <= ’0’;
      END LOOP;

ELSIF (R_40MHZ’EVENT AND R_40MHZ =’1’) THEN

  FA(0) <= FA(0) XOR
         ((NOT FA(0) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x″0"))
      OR (FA(0) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x″0"))
      OR (NOT FA(0) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x″0"))
      OR (FA(0) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x″0")));

  FA(1) <= FA(1) XOR
         ((NOT FA(1) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x″1"))
      OR (FA(1) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x″1"))
      OR (NOT FA(1) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x″1"))
      OR (FA(1) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x″1")));
```

## Appendix A. Real-World Converted Designs (continued)

```
 FA(2) <= FA(2) XOR
       ((NOT FA(2) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"2"))
    OR (FA(2) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"2"))
    OR (NOT FA(2) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"2"))
    OR (FA(2) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"2")));

FA(3) <= FA(3) XOR
       ((NOT FA(3) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"3"))
    OR (FA(3) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"3"))
    OR (NOT FA(3) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"3"))
    OR (FA(3) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"3")));

FA(4) <= FA(4) XOR
       ((NOT FA(4) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"4"))
    OR (FA(4) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"4"))
    OR (NOT FA(4) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"4"))
    OR (FA(4) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"4")));

FA(5) <= FA(5) XOR
       ((NOT FA(5) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"5"))
    OR (FA(5) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"5"))
    OR (NOT FA(5) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"5"))
    OR (FA(5) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"5")));

FA(6) <= FA(6) XOR
       ((NOT FA(6) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"6"))
    OR (FA(6) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"6"))
    OR (NOT FA(6) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"6"))
    OR (FA(6) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"6")));

FA(7) <= FA(7) XOR
       ((NOT FA(7) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"7"))
    OR (FA(7) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"7"))
    OR (NOT FA(7) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"7"))
    OR (FA(7) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"7")));

FB(0) <= FB(0) XOR
       ((NOT FB(0) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"0"))
    OR (FB(0) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"0"))
    OR (NOT FB(0) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"8"))
    OR (FB(0) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"8")));

FB(1) <= FB(1) XOR
       ((NOT FB(1) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"1"))
    OR (FB(1) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"1"))
    OR (NOT FB(1) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"9"))
    OR (FB(1) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"9")));

FB(2) <= FB(2) XOR
       ((NOT FB(2) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"2"))
    OR (FB(2) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"2"))
    OR (NOT FB(2) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"A"))
    OR (FB(2) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"A")));
```

## Appendix A. Real-World Converted Designs (continued)

```
  FB(3) <= FB(3) XOR
         ((NOT FB(3) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"3"))
      OR (FB(3) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"3"))
      OR (NOT FB(3) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"B"))
      OR (FB(3) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"B")));

  AB(0) <= AB(0) XOR
         ((NOT AB(0) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"8"))
      OR (AB(0) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"8"))
      OR (NOT AB(0) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"8"))
      OR (AB(0) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"8")));

  AB(1) <= AB(1) XOR
         ((NOT AB(1) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"9"))
      OR (AB(1) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"9"))
      OR (NOT AB(1) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"9"))
      OR (AB(1) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"9")));

  AB(2) <= AB(2) XOR
         ((NOT AB(2) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"A"))
      OR (AB(2) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"A"))
      OR (NOT AB(2) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"A"))
      OR (AB(2) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"A")));

  AB(3) <= AB(3) XOR
         ((NOT AB(3) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"B"))
      OR (AB(3) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"B"))
      OR (NOT AB(3) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"B"))
      OR (AB(3) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"B")));

  FB(4) <= FB(4) XOR
         ((NOT FB(4) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"4"))
      OR (FB(4) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"4"))
      OR (NOT FB(4) AND H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"C"))
      OR (FB(4) AND NOT H_FEP_SET AND NOT L_FEP_WE AND frbl_to_b(FEP_SEL=x"C")));

  AB(4) <= AB(4) XOR
         ((NOT AB(4) AND H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"C"))
      OR (AB(4) AND NOT H_PPB_SET AND NOT L_PPB_WE AND frbl_to_b(PPB_SEL=x"C"))
      OR (NOT AB(4) AND H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"C"))
      OR (AB(4) AND NOT H_PPA_SET AND NOT L_PPA_WE AND frbl_to_b(PPA_SEL=x"C")));

END IF;
END PROCESS;
END CONVERTED_ABL;

------------------------------------ cut here ------------------------------------

Module CONVERTER
Title 'Converter
Revision 01'

" This device converts a 32-bit floating point word from one format to another.
```

## Appendix A. Real-World Converted Designs (continued)

```
CONVERTER                     device       'MACH210A';

"Control Inputs:

CLK           PIN 35;" Clock
OE            PIN 10;" Low Active Output Enable
WE            PIN 11;" Low Active Write Enable
MODE          PIN 13;" Shift Mode

"Data I/O BITS:

D31           PIN 43   ISTYPE    'REG,BUFFER';
D30           PIN 42   ISTYPE    'REG,BUFFER';
D29           PIN 41   ISTYPE    'REG,BUFFER';
D28           PIN 40   ISTYPE    'REG,BUFFER';
D27           PIN 39   ISTYPE    'REG,BUFFER';
D26           PIN 38   ISTYPE    'REG,BUFFER';
D25           PIN 37   ISTYPE    'REG,BUFFER';
D24           PIN 36   ISTYPE    'REG,BUFFER';
D23           PIN 31   ISTYPE    'REG,BUFFER';
D22           PIN 30   ISTYPE    'REG,BUFFER';
D21           PIN 29   ISTYPE    'REG,BUFFER';
D20           PIN 28   ISTYPE    'REG,BUFFER';
D19           PIN 27   ISTYPE    'REG,BUFFER';
D18           PIN 26   ISTYPE    'REG,BUFFER';
D17           PIN 25   ISTYPE    'REG,BUFFER';
D16           PIN 24   ISTYPE    'REG,BUFFER';
D15           PIN 21   ISTYPE    'REG,BUFFER';
D14           PIN 20   ISTYPE    'REG,BUFFER';
D13           PIN 19   ISTYPE    'REG,BUFFER';
D12           PIN 18   ISTYPE    'REG,BUFFER';
D11           PIN 17   ISTYPE    'REG,BUFFER';
D10           PIN 16   ISTYPE    'REG,BUFFER';
D09           PIN 15   ISTYPE    'REG,BUFFER';
D08           PIN 14   ISTYPE    'REG,BUFFER';
D07           PIN  9   ISTYPE    'REG,BUFFER';
D06           PIN  8   ISTYPE    'REG,BUFFER';
D05           PIN  7   ISTYPE    'REG,BUFFER';
D04           PIN  6   ISTYPE    'REG,BUFFER';
D03           PIN  5   ISTYPE    'REG,BUFFER';
D02           PIN  4   ISTYPE    'REG,BUFFER';
D01           PIN  3   ISTYPE    'REG,BUFFER';
D00           PIN  2   ISTYPE    'REG,BUFFER';

H,L,C,Z,X   =  1,0,.C.,.Z.,.X.;

DIN         =  [D19.PIN,D18.PIN,D17.PIN,D16.PIN,
               D15.PIN,D14.PIN,D13.PIN,D12.PIN,
               D11.PIN,D10.PIN,D09.PIN,D08.PIN,
               D07.PIN,D06.PIN,D05.PIN,D04.PIN,
               D03.PIN,D02.PIN,D01.PIN,D00.PIN];

DOUT        =  [D31,D30,D29,D28,D27,D26,D25,D24,
               D23,D22,D21,D20,D19,D18,D17,D16,
               D15,D14,D13,D12,D11,D10,D09,D08,
               D07,D06,D05,D04,D03,D02,D01,D00];
```

## Appendix A. Real-World Converted Designs (continued)

```
DBIDI        =  [D19,D18,D17,D16,D15,D14,D13,D12,
                D11,D10,D09,D08,D07,D06,D05,D04,
                D03,D02,D01,D00];

DOUTFB       =  [L,L,L,L,L,L,L,L,L,L,L,L,
                D19.FB,D18.FB,D17.FB,D16.FB,
                D15.FB,D14.FB,D13.FB,D12.FB,
                D11.FB,D10.FB,D09.FB,D08.FB,
                D07.FB,D06.FB,D05.FB,D04.FB,
                D03.FB,D02.FB,D01.FB,D00.FB];

M0_SHIFT6    =  !WE & !MODE &
                ((D19.PIN == H) # (D18.PIN == H) # (D17.PIN == H));

M0_SHIFT3    =  !WE & !MODE &
                (((D19.PIN == L) & (D18.PIN == L) & (D17.PIN == L)) &
                ((D16.PIN == H) # (D15.PIN == H) # (D14.PIN == H)));

SHIFT0       =  !WE &
                (((D19.PIN == L) & (D18.PIN == L) & (D17.PIN == L)) &
                ((D16.PIN == L) & (D15.PIN == L) & (D14.PIN == L)));

M1_SHIFT6    =  !WE & MODE &
                ((D19.PIN == H) # (D18.PIN == H));

M1_SHIFT4    =  !WE & MODE &
                (((D19.PIN == L) & (D18.PIN == L)) &
                ((D17.PIN == H) # (D16.PIN == H)));

M1_SHIFT2    =  !WE & MODE &
                (((D19.PIN == L) & (D18.PIN == L) &
                (D17.PIN == L) & (D16.PIN == L)) &
                ((D15.PIN == H) # (D14.PIN == H)));

EQUATIONS

DOUT.CLK     =  CLK;

DOUT.OE      =  !OE;
DOUT        :=  M0_SHIFT6  &
                          [L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,H,L,D19.PIN,D18.PIN,
                               D17.PIN,D16.PIN,D15.PIN,D14.PIN,D13.PIN,D12.PIN,
                               D11.PIN,D10.PIN,D09.PIN,D08.PIN,D07.PIN,D06.PIN]

              #

                M0_SHIFT3  &
                          [L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,H,D16.PIN,D15.PIN,
                               D14.PIN,D13.PIN,D12.PIN,D11.PIN,D10.PIN,D09.PIN,
                               D08.PIN,D07.PIN,D06.PIN,D05.PIN,D04.PIN,D03.PIN]

              #

                SHIFT0     &
                          [L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,D13.PIN,D12.PIN,
                               D11.PIN,D10.PIN,D09.PIN,D08.PIN,D07.PIN,D06.PIN,
                               D05.PIN,D04.PIN,D03.PIN,D02.PIN,D01.PIN,D00.PIN]
```

## Appendix A. Real-World Converted Designs (continued)

```
            #

              M1_SHIFT6  &
                        [L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,H,H,D19.PIN,D18.PIN,
                             D17.PIN,D16.PIN,D15.PIN,D14.PIN,D13.PIN,D12.PIN,
                             D11.PIN,D10.PIN,D09.PIN,D08.PIN,D07.PIN,D06.PIN]

            #

              M1_SHIFT4  &
                        [L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,H,L,D17.PIN,D16.PIN,
                             D15.PIN,D14.PIN,D13.PIN,D12.PIN,D11.PIN,D10.PIN,
                             D09.PIN,D08.PIN,D07.PIN,D06.PIN,D05.PIN,D04.PIN]

            #

              M1_SHIFT2  &
                        [L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,H,D15.PIN,D14.PIN,
                             D13.PIN,D12.PIN,D11.PIN,D10.PIN,D09.PIN,D08.PIN,
                             D07.PIN,D06.PIN,D05.PIN,D04.PIN,D03.PIN,D02.PIN]

            #

              WE & DOUTFB;

Test_Vectors

([CLK,OE,WE,MODE,               DBIDI]           ->      DOUT)
 [C,H,L,X,                     ^b00000000101101110110] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000000101101110110;“Read,Shift0
 [C,H,L,L,                     ^b00000110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000100110101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00001010101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000101010101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00010010101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000110010101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00011010101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000111010101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00010110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000110110101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00001110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000101110101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00011110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000000111110101101011;“Read,Shift3/Mode0
 [C,H,L,L,                     ^b00111110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000001000111110101101;“Read,Shift6/Mode0
 [C,H,L,L,                     ^b01011110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000001001011110101101;“Read,Shift6/Mode0
 [C,H,L,L,                     ^b10011110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000001010011110101101;“Read,Shift6/Mode0
 [C,H,L,L,                     ^b10100110101101011011] -> Z;“Write
 [C,L,H,X,Z]    -> ^b00000000000000001010100110101101;“Read,Shift6/Mode0
```

## Appendix A. Real-World Converted Designs (continued)

```
 [C,H,L,L,                   ^b01100110101101011011] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001001100110101101;"Read,Shift6/Mode0
 [C,H,L,L,                   ^b11000110101101011011] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001011000110101101;"Read,Shift6/Mode0
 [C,H,L,L,                   ^b11111110101101011011] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001011111110101101;"Read,Shift6/Mode0
 [C,H,L,H,                   ^b00000100101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000000101001011011101;"Read,Shift2/Mode1
 [C,H,L,H,                   ^b00001000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000000110001011011101;"Read,Shift2/Mode1
 [C,H,L,H,                   ^b00001100101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000000111001011011101;"Read,Shift2/Mode1
 [C,H,L,H,                   ^b00010000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001001000010110111;"Read,Shift4/Mode1
 [C,H,L,H,                   ^b00100000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001010000010110111;"Read,Shift4/Mode1
 [C,H,L,H,                   ^b00110000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001011000010110111;"Read,Shift4/Mode1
 [C,H,L,H,                   ^b00111100101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001011110010110111;"Read,Shift4/Mode1
 [C,H,L,H,                   ^b10000000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001110000000101101;"Read,Shift6/Mode1
 [C,H,L,H,                   ^b01000000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001101000000101101;"Read,Shift6/Mode1
 [C,H,L,H,                   ^b11000000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001111000000101101;"Read,Shift6/Mode1
 [C,H,L,H,                   ^b11010100101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001111010100101101;"Read,Shift6/Mode1
 [C,H,L,H,                   ^b11101000101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001111101000101101;"Read,Shift6/Mode1
 [C,H,L,H,                   ^b11111100101101110110] -> Z;"Write
 [C,L,H,X,Z]    -> ^b00000000000000001111111100101101;"Read,Shift6/Mode1

End CONVERTER;

------------------------------------ cut here -----------------------------------

-- CONVERTED TO IEEE 1076 VHDL

-- Module CONVERTER
-- Title 'Converter
-- Revision 01'

-- This device converts a 32-bit floating point word from one format to another.

-- Control Inputs

use work.cypress.all;
use work.rtlpkg.all;
use work.int_math.all;

ENTITY CONVERTER IS PORT(
CLK,OE,WE,MODE                                : IN BIT;
D                                             : INOUT X01Z_VECTOR (0 TO 31));
```

## Appendix A. Real-World Converted Designs (continued)

```
attribute part_name of CONVERTER: entity is "c371";
END CONVERTER;

ARCHITECTURE CONVERTED_ABL OF CONVERTER IS
SIGNAL SHIFT2_TMP, SHIFT1_TMP, SHIFT0_TMP, SHIFT2,
       SHIFT1, SHIFT0, M0_SHIFT6, M0_SHIFT3,
       M_SHIFT0, M1_SHIFT6, M1_SHIFT4, M1_SHIFT2     : BIT;
SIGNAL D_TMP, D_FB                                   : BIT_VECTOR (0 TO 31);
BEGIN

P1: PROCESS
 BEGIN
  WAIT UNTIL CLK ='1';

   FOR i IN 0 TO 13 LOOP
   D_TMP(i) <=(D_FB(i+6) AND     SHIFT2 AND     SHIFT1 AND NOT SHIFT0)
           OR (D_FB(i+4) AND     SHIFT2 AND NOT SHIFT1 AND NOT SHIFT0)
           OR (D_FB(i+3) AND NOT SHIFT2 AND     SHIFT1 AND     SHIFT0)
           OR (D_FB(i+2) AND NOT SHIFT2 AND     SHIFT1 AND NOT SHIFT0)
           OR (D_FB(i+0) AND NOT SHIFT2 AND NOT SHIFT1 AND NOT SHIFT0)
           OR (WE AND D_TMP(i));
   END LOOP;

   D_TMP(14) <= ('0' AND M0_SHIFT6) OR ('1' AND M0_SHIFT3)
           OR ('0' AND  M_SHIFT0) OR ('1' AND M1_SHIFT6)
           OR ('0' AND M1_SHIFT4) OR ('1' AND M1_SHIFT2)
           OR (WE AND D_TMP(14));

   D_TMP(15) <= ('1' AND M0_SHIFT6) OR ('0' AND M0_SHIFT3)
           OR ('0' AND  M_SHIFT0) OR ('1' AND M1_SHIFT6)
           OR ('1' AND M1_SHIFT4) OR ('0' AND M1_SHIFT2)
           OR (WE AND D_TMP(15));

   FOR i IN 16 TO 28 LOOP
   D_TMP(i) <='0';
   END LOOP;

END PROCESS P1;

M0_SHIFT6  <= (NOT WE AND MODE) AND
            (D_FB(19) OR D_FB(18) OR D_FB(17));

M0_SHIFT3  <= (NOT WE AND NOT MODE AND NOT D_FB(19) AND NOT D_FB(18)
             AND NOT D_FB(17)) AND (D_FB(16) OR D_FB(15) OR D_FB(14));

M_SHIFT0   <= NOT WE AND NOT D_FB(19) AND NOT D_FB(18) AND NOT
            D_FB(17) AND NOT D_FB(16) AND NOT D_FB(15) AND NOT D_FB(14);

M1_SHIFT6  <= (NOT WE AND MODE) AND (D_FB(19) OR D_FB(18));

M1_SHIFT4  <= (NOT WE AND MODE AND NOT D_FB(19) AND NOT D_FB(18))
             AND (D_FB(17) OR D_FB(16));

M1_SHIFT2  <= (NOT WE AND MODE AND NOT D_FB(19) AND NOT D_FB(18) AND
             NOT D_FB(17) AND NOT D_FB(16)) AND (D_FB(15) OR D_FB(14));
```

## Appendix A. Appendix A. Real-World Converted Designs (continued)

```
SHIFT2_TMP <= M0_SHIFT6
             OR M1_SHIFT6
             OR M1_SHIFT4;

SHIFT1_TMP <= M0_SHIFT6
             OR M0_SHIFT3
             OR M1_SHIFT6
             OR M1_SHIFT2;

SHIFT0_TMP <= M0_SHIFT3;

-- Mapping for the Bidirectional buffers
-- D_TMP is the internal signal which drives the output buffer
-- OE    is the signal for output enable (active high)
-- D     is the pin name, matches name in port assignment
-- D_FB  is the signal from pin that feeds back and drives the internal
--       structure
G1: FOR i IN 0 TO 28 GENERATE
      B1:BUFOE PORT MAP(D_TMP(i), OE,D(i), D_FB(i));
END GENERATE;

B2: BUF PORT MAP(SHIFT0_TMP,SHIFT0); -- Forces logic synthesis to "split
B3: BUF PORT MAP(SHIFT1_TMP,SHIFT1); -- sums" into SHIFT codes that are
B4: BUF PORT MAP(SHIFT2_TMP,SHIFT2); -- encoded and placed on outputs D29-31

B5: BUFOE PORT MAP(SHIFT0, OE, D(29), open);
B6: BUFOE PORT MAP(SHIFT1, OE, D(30), open);
B7: BUFOE PORT MAP(SHIFT2, OE, D(31), open);

END CONVERTED_ABL;

------------------------------------ cut here -----------------------------------
```

*Warp* is a trademark and *Warp2* is a registered trademark of Cypress Semiconductor Corporation.
MACH is a trademark of Advanced Micro Devices, Inc.

# Abel™-HDL vs. IEEE-1076 VHDL

## Abstract

Currently there exist several popular Hardware Description Languages (HDLs) that allow designers to describe the function of complex logic circuits textually, as opposed to schematically. One of the most widely used of these languages is Data I/O's Abel-HDL. Abel-HDL, as a language, can be used to describe the behavior of logic circuits that can be fitted to a wide variety of PALs, PLDs, PROMs, and FPGAs from a variety of programmable logic IC manufacturers. IEEE-1076 VHDL (VHSIC Hardware Description Language) has recently been gaining widespread support. VHDL is an open, portable language defined and standardized by the IEEE that can be used to describe the behavior of an entire system from the highest levels of functionality all the way down to the logic-gate level. A majority of CAE vendors, programmable IC manufacturers, and third-party software vendors already have or are planning tools that support VHDL logic synthesis, logic modeling, and/or VHDL simulation.

The purpose of this application note is to compare and contrast the complexity and basic features of Abel-HDL with those of IEEE-1076 VHDL. Both of these languages are very robust in their support of different types of constructs that can be used to describe the same functionality at different levels of abstraction. It is beyond the scope of this document to exhaustively describe these possibilities or to present a complete tutorial for writing code in either language because of the great variety of constructs and syntax available with which to describe the functionality of a given circuit. Rather, a simple example design that contains a mixture of synchronous and asynchronous logic circuits will be shown. Sample code is written in both Abel-HDL and VHDL that describes the example's functionality and synthesizes to create functionally identical hardware. The code written here represents a typical level of abstraction that balances readability with compactness. With experience, designers can develop their own preferences for style. For instance, state machines can be described in a number of ways: state tables, IF-THEN-ELSE statements, CASE-WHEN statements, or explicitly using a combination of Register-Transfer-Level (RTL) code (individually describe each gate/register as a component with its inputs and outputs) and/or Boolean equations.

## Example Description

The following example is a circuit that creates a 50% duty cycle clock with programmable frequency. *Figure 1* shows the block diagram of this Programmable Clock Generator. The output of the circuit is CLK_OUT, whose period is equal to clock(ns)*incnt*2. To program the device, LD_CNT is used to latch the value present at the INCNT(3:0) inputs into the 4-Bit Input Register. The output of this register is ENDCNT(3:0). The clock input is used to clock the 4-bit Up/Down Counter, whose output is COUNT(3:0). The 4-bit Comparator is an asynchronous comparator that compares the values of COUNT and ENDCNT. Its outputs are endeqcnt (ENDCNT = COUNT), endltcnt (ENDCNT < COUNT), endeq0 (ENDCNT = 0), and cnteq0 (COUNT = 0). Note that it is possible for ENDCNT to be less than COUNT if a new value for ENDCNT is loaded into the input registers that is less than the current value of COUNT. The cnt_state State Machine controls the

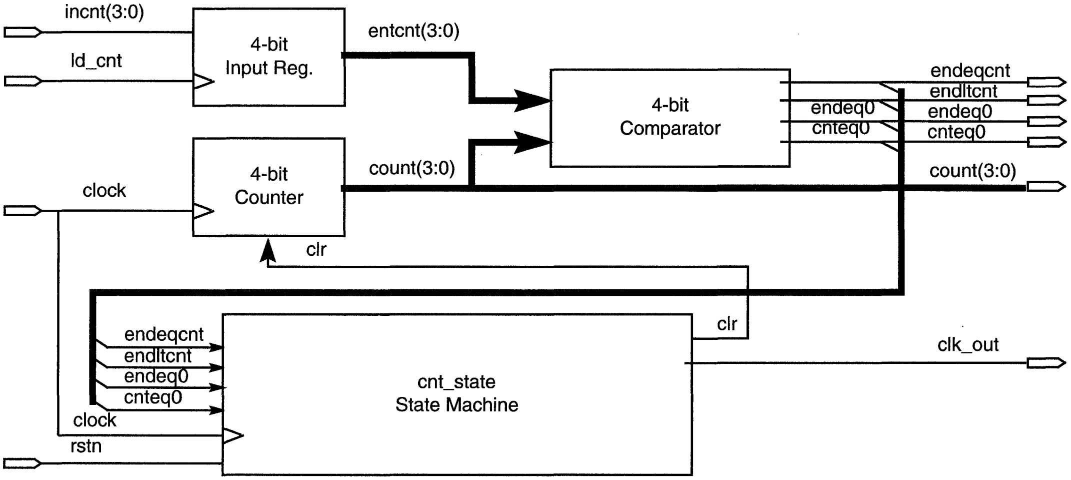


**Figure 1. Block Diagram**

CLK_OUT signal and is clocked and enabled by the clock and en inputs, respectively.

*Figure 2* shows the state diagram for cnt_state. The state machine consists of three states: *reset*, *cnt_up*, and *cnt_down*. Two state bits are used to describe this Mealy-type state machine. The state machine powers-up to the reset state. It also synchronously enters the *reset* state from any state if RSTN=0. Once RSTN=1, the state machine will count up until COUNT equals ENDCNT, at which point it will begin to count down until COUNT equals 0 and then repeat the cycle. If at any time ENDCNT = 0, cnt_state will return to *reset*. Also, if ENDCNT is ever less than COUNT, thus signifying an invalid condition, cnt_state will go to *reset*.

The PLD targeted is the CY7C335. It was chosen because it can be used to illustrate a variety of features contained in some PLDs such as input registers, multiple clocks, buried registers, and synchronous reset/preset. However, any PLD with sufficient resources could be targeted. This is one of the main advantages of using an HDL. High-level languages, by design, allow a designer to write generic code that can be targeted to different devices/architectures with little or no modification. Going one step further, VHDL allows simulation and debugging of the logic from the source code. This can greatly reduce the overall design cycle time by allowing functional verification of the logic prior to targeting a specific physical device. Once the logic has been verified, the designer can then compile and fit the same design into a variety of devices. From here, the designer can decide which implementation best suits his requirements.

## Abel-HDL vs. VHDL

In general, the constructs used to describe logic function in both ABEL-HDL an IEEE-1076 VHDL are very similar. Each can accept Boolean equations, truth tables, state descriptions (IF-THEN and CASE-WHEN), and signal assignments that are quite similar in appearance. However, differences do exist in syntax and in the overhead sections. These are the declarative sections which define hierarchical organization, design libraries, etc. As we shall see, some of these statements found in the VHDL code have no direct counterpart in Abel-HDL. This is because Abel-HDL was created with a single purpose in mind, logic synthesis for PALs and PLDs. Therefore, with Abel-HDL, assumptions can be made by the compiler which can simplify the overhead syntax needed. Because VHDL is a one-language-fits-all standard which applies to synthesis, modeling, and system definition at all levels, some syntax overhead is necessary to fully describe a design in the proper context. For example, the use of standard and user-definable packages and libraries allows many designs to share commonly used definitions, components, macros, functions, etc. Fortunately for the logic designer, these statements are used in most designs so that familiarity with them comes quickly and easily. This commonality also allows ample "cutting-and-pasting" from design to design.

The following sections compare and contrast the source code files for Abel-HDL and VHDL on a logical section-by-section basis. Both of these files, when compiled, create functionally identical logic. Copies of the source code files in their entirety can be found in the Appendices.

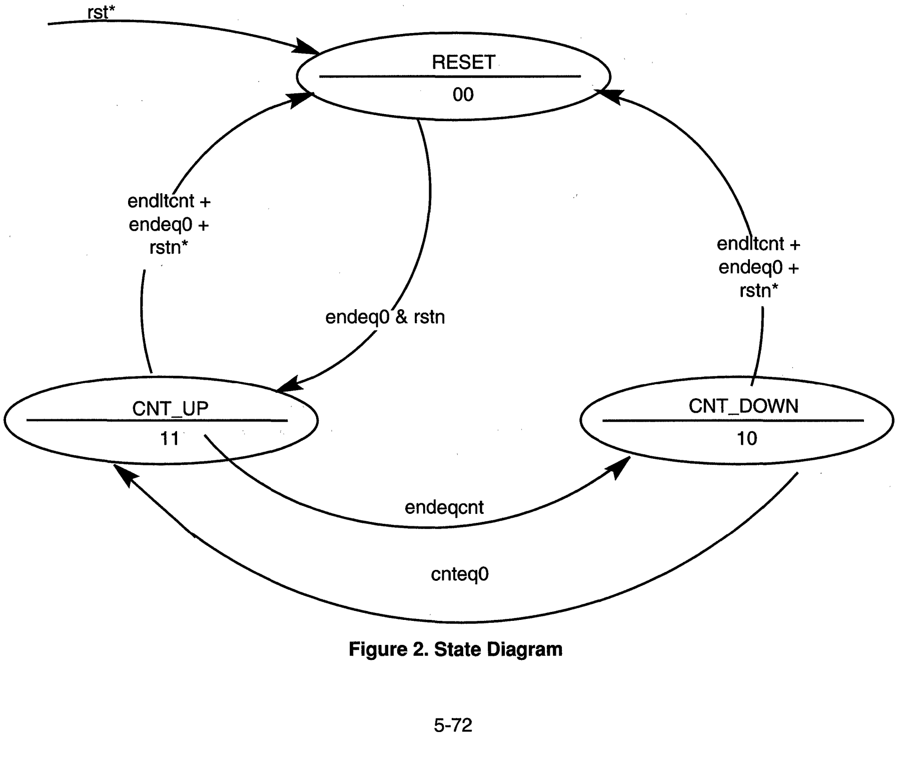


**Figure 2. State Diagram**

## Design and I/O Declarations

The basic structure of both Abel-HDL and VHDL source files allow for one or more design units to be defined within it. Each design unit is a complete logic description. Multiple design units may be combined hierarchically in a single top-level source file which binds them together. In the example, we have used a single design unit for simplicity. The first section of code chosen for comparison contains the design declaration and device I/O declarations. In this example, the pin numbers have been fixed. This is optional in either language. The declaration of the target device is also optional in the source code. The targeted device need not be declared until it's time to compile and fit the logic.

*Figure 3* contains the Abel-HDL code that declares the module name (line 01), the device (line 03), and the input and output pin numbers and types (lines 04–09). Line 45 is the end statement that completes the design module.

The corresponding VHDL code is shown in *Figure 4*. VHDL requires a slightly different structure. A design unit consists of an Entity section and an Architecture section. By themselves,

```
01: module clk_gena;

02: declarations
03: device 'p335';

    "Inputs
04:  clock, ld_cnt, rstn                 pin 3,1,7;
05:  incnt3, incnt2, incnt1, incnt0      pin 6,5,4,2;

    "Outputs
06:  endeqcnt, endltcnt                  pin 25,23 istype 'com';
07:  endeq0, cnteq0                      pin 24,27 istype 'com';
08:  clk_out                             pin 17 istype 'reg_d';
09:  count3, count2, count1, count0      pin 19,15,28,26;
.
.
.
45: end clk_gena;
```

**Figure 3. Abel-HDL Design and I/O Declarations**

```
01: entity clk_genv is
02: port(clock, ld_cnt, rstn      :in bit;
03:     incnt                      :in bit_vector(3 downto 0);
04:     count                      :buffer bit_vector(3 downto 0);
05:     endeqcnt, endltcnt, clk_out :buffer bit;
06:     endeq0, cnteq0             :buffer bit);

07: attribute part_name of clk_genv : entity is "c335";
08: attribute pin_numbers of clk_genv : entity is
09:  "clock:3 ld_cnt:1 rstn:7 clk_out:17 "
10:& "incnt(3):6 incnt(2):5 incnt(1):4 incnt(0):2 "
11:& "count(3):19 count(2):15 count(1):28 count(0):26 "
12:& "endeqcnt:25 endltcnt:23 endeq0:24 cnteq0:27";
13: end clk_genv;

14: use work.int_math.all;
15: use work.rtlpkg.all;

16: architecture behave of clk_genv is
.
.
.
22: begin
.
.
.
60: end behave;
```

**Figure 4. VHDL Design and I/O Declarations**

```
10:  rst_ctr                              node istype 'reg_d';
11:  endcnt3,endcnt2,endcnt1,endcnt0      node;
12:  incnt     = [incnt3,incnt2,incnt1,incnt0];
13:  count    = [count3,count2,count1,count0];
14:  endcnt   = [endcnt3,endcnt2,endcnt1,endcnt0];
15:  outputs = [count,endeqcnt,endltcnt,endeq0,cnteq0,clk_out];
16:  cnt_state = [rst_ctr,clk_out];
```

**Figure 5. Abel-HDL Signal Declarations**

```
17:  signal endcnt : bit_vector(3 downto 0);
18:  signal cnt_state : bit_vector(0 to 1);
```

**Figure 6. VHDL Signal Declarations**

each of these is considered a separate design component. The Entity section defines the component name (line 01). The port statement (lines 02–06) declares the I/O of the entity. For each signal, a mode (in, out, buffer) defines the direction of the pin (buffer signifies output with feedback). The signal type is also defined here. The type of a signal defines size and possible values which that signal can take on. Type bit defines a one-bit signal that can have the values of "0" and "1". Type bit_vector (0 to 3) declares that the signal is a 4-bit vector, each bit of which can take on the values of "0" or "1". Line 07 declares the target device and is optional. Lines 08-12 declares the fixed pin assignments and is also optional. Note that this line could be written as a single line terminated with a ";". The "&" in this context signifies a continuation from the previous line. Line 13 is the end statement that terminates the *clk_genv* entity.

Lines 14 and 15 are statements that call out other libraries and packages making them visible to this design. These libraries and packages may be predefined in the VHDL language or may be user defined. They may contain components, functions, procedures, declarations, etc., which may then be used by the current design. For instance, *work.int_math.all* enables all functions contained in the package *int_math* (integer math), which is found in the work library. These functions describe the operation of the "+" and "–" operators used in the up and down counter logic descriptions. The package *rtlpkg* contains the definition of the Global Synchronous Set (gss) statement used on line 27.

The second part of a complete design unit is the Architecture section. In this section, we find the description of the behavior of the black box defined in the Entity section. Associated with the Architecture statement are begin and end statements. Line 16 declares the Architecture name, behave, for the following statements which describe the functionality of the Entity *clk_genv*. The Entity and Architecture sections are separated because VHDL allows multiple architectures to be defined for a given entity. Only one architecture can be associated with an entity in a given design. This feature allows multiple versions of an architecture to be saved in a library. The Configuration statement is used to select a specific architecture (see Reference 3).

## Internal Signal Declarations

In both Abel-HDL and VHDL, internal signals (nodes) may be defined. These signals do not connect directly to an input or an output pin and may result from buried logic or may simply represent a wire to transfer data. Both languages are similar in that the signal name and type are declared.

As can be seen in *Figure 5*, lines 10 and 11 of the Abel-HDL code declare the signals RST_CTR and ENDCNT(3...0). Each is defined as type node, as opposed to type pin, which would mean that the signal would be connected to an I/O pin. An optional signal attribute may be used with the keyword istype to define the signal's characteristics more explicitly. Lines 12–16 show the groupings of signals into sets. Defining sets allows a group of signals to be referenced by one name. Any operation performed on the set name will be performed on each member of the set.

VHDL allows signal names that represent, among others, bit vectors such as the one shown in line 17 in *Figure 6*. Here ENDCNT is equivalent to [ENDCNT(3), ENDCNT(2), ENDCNT(1), ENDCNT(0)]. As seen earlier in the entity declaration, INCNT has been defined as a port (I/O pins) and is a 4-bit vector similar to ENDCNT. Individual signals cannot be declared and grouped into sets as with Abel-HDL. Rather, groups are declared initially as bit vectors. The individual members of the set can then be operated upon separately or as a group (line 58 of the VHDL source code shows cnt_state(1) being assigned to the output CLK_OUT).

## State-Machine State Definitions

The section where state register assignments are declared is very similar for Abel-HDL and VHDL. Both languages require assignment of a constant value to a name which gets compared to the current value of the state bits (cnt_state) in the actual state machine implementation (IF-THEN-ELSE, CASE-WHEN). Shown in this document is one method of designing a state machine. Both Abel-HDL and VHDL allow a variety of ways in which to create a state machine. For large state machines, a more compact implementation might be with a Truth-Table in which inputs and outputs are described in a tabular form. This method is more compact but some may find it less "readable" than other methods. Both languages also support Mealy, Moore, and one-hot (one register per state) state machine implementations. *Figure 7* shows the Abel-HDL state assignment code.

```
17:  reset      = [0,0];
18:  cnt_up     = [1,1];
19:  cnt_down   = [1,0];
```

**Figure 7. Abel-HDL State Machine Definition**

```
19:  constant reset : bit_vector(0 to 1) := "00";
20:  constant cnt_up : bit_vector(0 to 1) := "11";
21:  constant cnt_down : bit_vector(0 to 1) := "10";
```

**Figure 8. VHDL State Machine Definition**

```
20: equations
21:     endeqcnt = ((endcnt.fb - 1) == count.fb);
22:     endltcnt = (endcnt.fb < count.fb);
23:     endeq0 = (endcnt.fb == 0);
24:  cnteq0 = (count.fb == 1);
25:     outputs.sp = !rstn;
26:     count.clk = clock;
```

**Figure 9. Abel-HDL Combinatorial Logic Equations**

```
23:  endeqcnt <= '1' when (count = (endcnt-1)) else '0';
24:  endltcnt <= '1' when (endcnt < count) else '0';
25:  endeq0   <= '1' when (endcnt = "0000") else '0';
26:  cnteq0   <= '1' when (count = "0001") else '0';
27:  gss <= NOT(rstn);
```

**Figure 10. VHDL Combinatorial Logic Equations**

*Figure 8* shows the VHDL code needed to make state assignments. Note here the increased verbosity relative to Abel-HDL. This, again, is due to the fact that VHDL is a more general-purpose language and that statements must be more explicit. Here, a constant is defined to be a certain type (bit_vector) and then is assigned a initial value using the ":=" operator. Note that VHDL's usage of the ":=" operator is different than its meaning as a registered signal assignment operator in Abel-HDL.

## Combinatorial Logic Equations

Both Abel-HDL and VHDL allow combinatorial Boolean) logic equations. As *Figures 9* and *10* show, the syntax is quite similar. Combinatorial statements in Abel-HDL are signified by a "=" operator. Use of the *istype reg* attribute in the signal declaration section and/or use of the appropriate explicit signal extensions (.c, .q, .d, etc...) and the ":=" operator signifies registered logic. In VHDL, the syntax for both a combinatorial and registered signal assignment is the same, "<=". The difference being determined by where in the code the statement appears. It is treated as a registered signal only if the signal assignment statement occurs inside of a clocked process. (See References 1 and 3 for full explanation of processes.)

Since the CY7C335 used in this example, like many other devices, contains special features, such as global or individual resets, presets, or OEs, there needs to be a means to expressly access them. Lines 25 and 27 of the Abel-HDL and VHDL, respectively, show how to access the available global synchronous preset of the device. In the Abel-HDL code we have defined a set to be all of the output signals (see Line 14). By simply using the .sp extension, !rstn is assigned to the global synchronous preset signal. Note that it is not necessary to define a set. Since in this device the preset is global, by assigning !rstn to the preset of one output register, all are automatically connected. Line 26 assigns the signal clock to the counter registers.

In VHDL, since extensions are not allowed in signal assignments, special functions are created and placed in a standard package which access these specific device features. In this case, the gss (global synchronous set) function is found in the package *rtlpkg*. When using a function (or other statement) found in a package, a use statement must be added (Line 15) so that the contents of the package are visible to this design. This requirement may seem cumbersome on the surface. It, in fact, represents one of the most powerful advantages of using VHDL. It gives the ability to save and reuse commonly used statements in standard or user defined packages which can then be accessed by any design. These packages can in turn be placed in libraries which can be organized by function, project, etc.

## Input Register Logic Definition

The CY7C335 was chosen for this example because it can be used to illustrate a variety of features that may be found in other programmable logic devices. In this section, we are making use of the registered inputs. The inputs are defined as INCNT(3:0). Once registered, these signals become ENDCNT(3:0) and are assigned to internal nodes in the device.

In the Abel_HDL code shown in *Figure 11*, Line 27 assigns the signal LD_CNT to be the clock for the ENDCNT registers. Line 28 uses the registered assignment operator, ":=", to create ENDCNT from INCNT.

In VHDL, to create registered logic, a process must be used. This highlights a key concept in VHDL, the notion of concurrent vs. sequential statements. All concurrent statements are continuously and simultaneously evaluated, creating combi-

```
27:   endcnt.clk = ld_cnt;
28:   endcnt := incnt;
```

**Figure 11. Abel-HDL Input Register Definition**

natorial logic. Sequential statements, as the name implies, are evaluated in order. The IF-THEN-ELSE construct is a classic example of a sequential statement. In VHDL, only statements found within a process are sequential. The processes themselves are concurrent and are continuously evaluated at the same time as all other statements between the begin and end of an Architecture section. A process is awakened (i.e., evaluated) when a change occurs in the value of a signal that is sensitive to that process. Sensitive signals are defined by the use of a sensitivity list or a wait statement at the beginning of the process. In our example we have used the wait statement to awaken a process when the clock signal for the associated logic sees a rising edge (Lines 28–31 of *Figure 12*). Here, the *in_reg* process is evaluated when a rising edge occurs on LD_CNT. Inside, the process statements are evaluated sequentially. In this process there is a single statement which, when evaluated, causes the value of INCNT to be transferred to ENDCNT. This effectively creates a register clocked by LD_CNT.

## State-Machine Description

The description of the simple three-state state machine (see *Figure 2*) is shown next for both Abel-HDL and VHDL in *Figures 13* and *14*, respectively. In general, both languages allow a variety of state machine definition methods. Included are truth tables, IF-THEN-ELSE statements, and CASE-WHEN statements. Both implementations require a state machine name declaration, clock declaration, and state descriptions.

In the Abel-HDL code of *Figure 13*, line 29 declares the signal clock to be the clock source for the state registers. Line 30 declares the following state descriptions to be for the state machine cnt_state. Lines 31–44 are the descriptions for each of the three states. Within each state description can be found signal assignments and IF-THEN-ELSE statements defining the conditional next-state assignments.

```
28: in_reg : process begin
29:     wait until (ld_cnt = '1');
30:         endcnt <= incnt;
31: end process;
```

**Figure 12. VHDL Input Register Definition**

Similar to Abel-HDL, the VHDL code of *Figure 14* contains a clock declaration on line 33 (the wait until statement implies clock is the register clock in this process), and a state machine declaration on line 34 (the case statement defines cnt_state as the state machine under evaluation). Lines 35–57 contain the individual state descriptions. Lines 32 and 58 declare the beginning and end of the process called counter. This explicit declaration of the beginning and end of processes is necessary because of the VHDL distinction between concurrent statements and sequential (within a process) statements. Note the addition of the “when others” statement. This is added to insure the state machine can recover from an invalid (undefined) state. Lastly, line 59 assigns the value of the state bit cnt_state(1) to the output signal CLK_OUT. Had this statement been placed inside the process it would have been treated as a sequential statement and, therefore, would be registered. This would have caused a registered, or pipelined, delay to be added to CLK_OUT.

## Summary

In summary, as design languages, Abel-HDL and IEEE-VHDL are really quite similar in complexity. Many experienced Abel-HDL users may perceive VHDL to be unnecessarily complicated. This may be true if one is limited to the smaller playing field covered by Abel-HDL. VHDL, on the other hand, covers a broader set of applications, such as full system-level description and simulation. The extra verbosity is minimal when compared to the extra functionality provided. For instance, VHDL allows true source code simulation, an easy migration path to ASICs (standard, portable language), and design with different types such as integers, enumerated types, records, etc. It also is truly device independent. For instance, falling edge clocks and XORs can be described behaviorally in VHDL whereas in Abel-HDL, the target device must be declared and specific fuses programmed to make use of these special features.

```
29:  cnt_state.clk = clock;
30:  state_diagram cnt_state
31:       state reset:
32:                 count := 0000;
33:            if (endeq0) then reset;
34:            else cnt_up;

35:       state cnt_up:
36:                 count := (count.fb + 1);
37:            if (endltcnt # endeq0) then reset;
38:            else if (endeqcnt) then cnt_down;
39:            else cnt_up;

40:       state cnt_down:
41:                 count := (count.fb - 1);
42:            if (endltcnt # endeq0) then reset;
43:            else if (cnteq0) then cnt_up;
44:            else cnt_down;
```

**Figure 13. Abel-HDL State Machine Equations**

```
counter : process begin
 wait until (clock = '1');
      case cnt_state is
      when reset =>
        count <= "0000";
      if (endeq0 = '0') then cnt_state <= cnt_up;
      else cnt_state <= reset;
      end if;
      when cnt_up =>
      count <= count +1;
      if (endltcnt='1' OR endeq0='1') then
      cnt_state <= reset;
      elsif (endeqcnt = '1') then cnt_state <= cnt_down;
      else cnt_state <= cnt_up;
      end if;
      when cnt_down =>
      count <= count - 1;
      if (endltcnt='1' OR endeq0='1') then
      cnt_state <= reset;
      elsif (cnteq0 = '1') then cnt_state <= cnt_up;
      else cnt_state <= cnt_down;
      end if;
      when others =>
      count <= "0000";
      cnt_state <= reset;
      end case;
end process;
 clk_out <= cnt_state(1);
```

**Figure 14. VHDL State Machine Equations**

## References

1. Cypress Semiconductor, Warp2® *User's Manual*, 1993.
2. Data I/O, *Abel Design Software User Manual*, 1990.
3. S. Mazor and P. Langstraat, *A Guide To VHDL*, Kluwer Academic Publishers, Norwell, MA, 1992.
4. Douglas L. Perry, VHDL Second Edition, McGraw-Hill Series, Computer Engineering, 1994.

## Appendix A. VHDL Design File for Prog. Clock Generator

```
entity clk_genv is
 port(clock, ld_cnt, rstn     :in bit;
      incnt             :in bit_vector(3 downto 0);
      count             :buffer bit_vector(3 downto 0);
      endeqcnt, endltcnt, clk_out  :buffer bit;
      endeq0, cnteq0     :buffer bit);
attribute part_name of clk_genv : entity is ″c335";
attribute pin_numbers of clk_genv : entity is
   ″clock:3 ld_cnt:1 rstn:7 clk_out:17 ″
 & ″incnt(3):6 incnt(2):5 incnt(1):4 incnt(0):2 ″
 & ″count(3):19 count(2):15 count(1):28 count(0):26 ″
 & ″endeqcnt:25 endltcnt:23 endeq0:24 cnteq0:27";
end clk_genv;

use work.int_math.all;
use work.rtlpkg.all;

architecture behave of clk_genv is

 signal endcnt : bit_vector(3 downto 0);
 signal cnt_state : bit_vector(0 to 1);
 constant reset : bit_vector(0 to 1) := ″00";
 constant cnt_up : bit_vector(0 to 1) := ″11";
 constant cnt_down : bit_vector(0 to 1) := ″10";

begin

 endeqcnt <= ’1’ when (count = (endcnt-1)) else ’0’;
 endltcnt <= ’1’ when (endcnt < count) else ’0’;
 endeq0   <= ’1’ when (endcnt = ″0000") else ’0’;
 cnteq0   <= ’1’ when (count = ″0001") else ’0’;
 gss <= NOT(rstn);

in_reg : process begin
    wait until (ld_cnt = ’1’);
       endcnt <= incnt;
end process;

counter : process begin
    wait until (clock = ’1’);
    case cnt_state is
    when reset =>
        count <= ″0000";
      if (endeq0 = ’0’) then cnt_state <= cnt_up;
      else cnt_state <= reset;
      end if;

    when cnt_up =>
      count <= count +1;
      if (endltcnt=’1’ OR endeq0=’1’) then
      cnt_state <= reset;
      elsif (endeqcnt = ’1’) then cnt_state <= cnt_down;
      else cnt_state <= cnt_up;
      end if;
```

## Appendix A. VHDL Design File for Prog. Clock Generator (continued)

```
 when cnt_down =>
      count <= count - 1;
      if (endltcnt='1' OR endeq0='1') then
      cnt_state <= reset;
      elsif (cnteq0 = '1') then cnt_state <= cnt_up;
      else cnt_state <= cnt_down;
      end if;

   when others =>
      count <= "0000";
      cnt_state <= reset;

    end case;
end process;
 clk_out <= cnt_state(1);
end behave;
```

## Appendix B. Abel-HDL Design File for Prog. Clock Generator

```
module clk_gena;

declarations
device 'p335';

"Inputs
 clock, ld_cnt, rstn         pin 3,1,7;
 incnt3, incnt2, incnt1, incnt     pin 6,5,4,2;

"Outputs
 endeqcnt, endltcnt        pin 25,23 istype 'com';
 endeq0, cnteq0          pin 24,27 istype 'com';
 clk_out            pin 17 istype 'reg_d';
 count3, count2, count1, count0    pin 19,15,28,26;

 rst_ctr            node istype 'reg_d';
 endcnt3,endcnt2,endcnt1,endcnt    node;

 incnt    = [incnt3,incnt2,incnt1,incnt0];
 count    = [count3,count2,count1,count0];
 endcnt   = [endcnt3,endcnt2,endcnt1,endcnt0];
 outputs = [count,endeqcnt,endltcnt,endeq0,cnteq0,clk_out];

 cnt_state = [rst_ctr,clk_out];
 reset     = [0,0];
 cnt_up    = [1,1];
 cnt_down  = [1,0];

equations

 endeqcnt = ((endcnt.fb - 1) == count.fb);
 endltcnt = (endcnt.fb < count.fb);
 endeq0 = (endcnt.fb == 0);
 cnteq0 = (count.fb == 1);
 outputs.sp = !rstn;
 count.clk = clock;
 endcnt.clk = ld_cnt;
 endcnt := incnt;

 cnt_state.clk = clock;
 state_diagram cnt_state
      state reset:
                count := 0000;
           if (endeq0) then reset;
           else cnt_up;
```

## Appendix B. Abel-HDL Design File for Prog. Clock Generator (continued)

```
      state cnt_up:
                count := (count.fb + 1);
           if (endltcnt # endeq0) then reset;
           else if (endeqcnt) then cnt_down;
           else cnt_up;

      state cnt_down:
                count := (count.fb - 1);
           if (endltcnt # endeq0) then reset;
           else if (cnteq0) then cnt_up;
           else cnt_down;
end clk_gena;
```

*Warp2* is aregistered trademark of Cypress Semiconductor Corporation.
Abel is a trademark of Data I/O Corporation.

CYPRESS

# State Machine Design Considerations and Methodologies

The use of state machines provides a systematic way to design complex sequential logic circuits—an increasingly popular approach since the advent of PLD (Programmable Logic Device) circuitry. This application note describes the many options encountered during the state machine design cycle. By exhaustively walking through the PLD-based design example presented here, you can weigh the merits of several design approaches.

## Definitions of Commonly Used Terms

*External input vector*—External signals (stimulus) applied to the state machine.

*System outputs*—Signals generated by the state machine that are explicitly designed for availability to the external system (hardware outside of the state machine). Registered system outputs can also be fed back into the state machine as part of the State Vector, which is then used in the decode of the state machine's next state.

*State registers*—Registers used exclusively for determining the next state of the machine (feedback).

*State outputs*—Outputs of the state registers that are available to the external system. (They are typically available to the external machine for debug or due to the lack of buried registers.)

*State vector or machine state*—The registered feedback information defining the present state of the machine and required to determine the next state of the machine.

*State path*—The transitional condition that must be met for the state machine to progress from one state to another. The state path typically consists of one or more product terms generated from external inputs, although other state paths are possible.

*Total input vector*—The combination of the external input vector and the state vector. The total input vector is decoded to generate the next state of the machine.

## State Machine Entry Methods

There are many ways of describing a state machine, each with distinct advantages and disadvantages. Three popular description methods are state diagrams, state tables, and high-level languages (HLLs). The state diagram provides an easily observable flow description of the state machine. Because the ability to view the flow of states provides distinct documentation advantages, state diagrams will be used throughout this application note to describe the example state machine.

Upon completing a state diagram, you can easily convert the diagram's visual information into the other types of state machine description or directly into Boolean equations. Several available software programs accept their own forms of state table, HLL, and/or Boolean entry. You can enter all these formats easily via your favorite text editor. The software then translates the inputs into suitable forms (usually a JEDEC map) for hardware implementation.

Another method of describing a state machine, the state table, offers perhaps the most concise description. Its major advantage over the other entry methods is the availability of state table reduction methods (see Reference 1). When applied to your state table definition, a reduction program generates a minimal model for the function. The software used for state machine synthesis throughout this application note uses the state table method of entry. The program is called LOG/iC™ from Isdata Corporation.

Finally, high-level language (HLL) state machine entry is probably the most popular form of state machine design. HLLs typically offer C-language-like instructions (e.g., case, if-then-else, etc.) to describe the machine.

## A Sample State Machine

The sample state machine is a clock generator for a pipelined (three system execution stages), bit-slice-based, central processing unit (CPU). Each of the three system execution stages contains two clocks for a total of six system clocks for every instruction execution. With pipelining enabled, each instruction takes an average of two clock periods. Further, external hardware unaffected by CPU wait and stop states (e.g., cache memory) needs both polarities of an additional free-running clock.

To minimize clock edge skew, the state machine provides both versions of the clock. To put the timing of this application into perspective, executing each pipeline stage in an 80-ns period (or 12.5 MHz) requires the state machine to run at 25 MHz. This speed is well within the range of the available PALs, EPLDs and PROMs that can be used to implement the state machine.

Each of the pipeline's three execution stages has a specific function. Briefly, the first stage of the pipeline accesses the Writable Control Store (WCS) RAM. The Arithmetic Logic Unit (ALU) execution occurs during the second stage of the pipeline. Finally, the third pipeline stage clocks status and memory address registers. The function(s) performed during each of the three stages are described in greater detail in the State Machine Output Definition section of this application note.

If this design only generates a simple set of pipelined clocks, why not use shift registers and miscellaneous glue logic instead of a state machine? There are two reasons to consider a state machine. First, it is usually desirable to minimize the number of chips required; the state machine in PLD form might need external glue logic, but significantly less than the shift register solution.

The second reason for considering a state machine is that this application requires more then just a simple set of pipeline clocks. The function of the clock signals is to provide control of the CPU in multiple modes of operation. The desired modes of operation follow.

### PIPELINED RUN Mode

In this mode, the CPU simultaneously performs the instructions in all three stages of the pipeline. For example, while instruction n does an ALU operation, instruction n+1 accesses WCS, and instruction n–1 clocks ALU status.

### NONPIPELINED RUN Mode

NONPIPELINED RUN mode performs all three stages of instruction execution without overlap. The time to complete one nonpipelined instruction equals the average of three pipelined instructions.

### CPU STOP

The system must have a way to perform an orderly stop of CPU execution from both of the above run modes. This stop might be the result of several possible conditions, including a utility stop from a system control unit, a single step, a breakpoint, or a response to external hardware (e.g., a logic analyzer). The free-running clocks continue to run during the CPU STOP mode and remain running at all times, except during a reset condition.

### CPU WAIT

In CPU WAIT mode, an external condition causes a delay in an instruction's execution. The instruction pauses until the external condition is removed. One application for the CPU WAIT mode is to handle a cache miss. When a cache miss occurs, the CPU remains in the CPU WAIT mode until the cache completes its memory transfer.

### SINGLE STEP

The ability to execute one instruction at a time is needed to debug the CPU. You can easily implement SINGLE STEP external to the clock state machine by pulsing the RUN signal. SINGLE STEP mode is described further in the State Machine Input Definition section of this application note.

### INTERRUPT

A variety of system conditions can interrupt the CPU out of its normal execution sequence and immediately start the execution of the interrupt handler. The influence of the INTERRUPT mode on the system clocks will be discussed in greater detail later in this application note.

### REPEAT INSTRUCTION

The REPEAT INSTRUCTION mode is a CPU debug feature. It is a good idea to implement this mode external to the clock state machine. By dubbing the clock to the instruction register and the interrupt line to the clock state machine, the CPU continually executes the instruction in the instruction register.

## Synchronous vs. Asynchronous Machine

At this point in the state machine design, an appropriate type of state machine must be chosen to match the application. Two major types are the asynchronous and the synchronous implementations. The asynchronous machine changes state when one or more of its inputs changes from a previously stable input state. After a state change, the outputs of the state machine settle, while the machine stabilizes once again. A basic example of an asynchronous state machine would be a simple SR latch built from two NAND gates (*Figure 1*). For the clocking application considered in this application note, the asynchronous state machine implementation would be a poor choice, due to the instability of the system outputs.

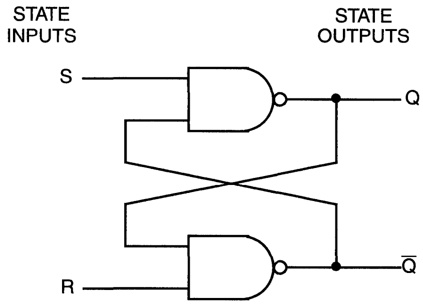


**Figure 1. SR Latch, Asynchronous State Machine Example**

The synchronous state machine offers a better choice. A synchronous state machine block diagram appears in *Figure 2.* Generally, a synchronous state machine samples the total input vector at specific periods to determine the machine's next state. When designing synchronous state machines, it is important to avoid state register metastability. External inputs to the machine must be synchronized to guarantee stable state register inputs, and the feedback time plus data set-up time to the state register clock must be less then or equal to the state clock period.

The modern theory of synchronous state machines was pioneered by Mealy and Moore (see Reference 1). Mealy and Moore machines differ slightly from each other in the way they control the system outputs. During a specific machine state, a Mealy machine allows the input conditions to alter the system outputs (the outputs depend on the "total" input state). In contrast, a Moore machine system outputs depend only on the present machine state. Thus, the system outputs remain stable until the next time period, when the Moore machine samples the total input vector to determine the next state. If all design conditions are met (external inputs are stable prior to the next state clock), the Moore machine provides glitch-free system outputs-a desirable characteristic for the CPU system clock. The design described here is therefore implemented as a Moore machine.

## Clock Generator Output Definition

As explained earlier, each of the three system execution stages contains two clocks for a total of six system clocks for every instruction execution. The naming convention for these clocks is

CLK_xy

where x = 1, 2, or 3, representing the first, second, or third stage of the instruction execution and y = A or B, representing the first or second half of the execution stage.

Following this convention, the state machine's two free-running clocks are named CLK_A and CLK_B. These clocks run at half the state clock frequency and 180 degrees out of phase. The free-running clocks occur at the same time as their respective CLK_xA and CLK_xB clocks.

The major clock functions for this application are:

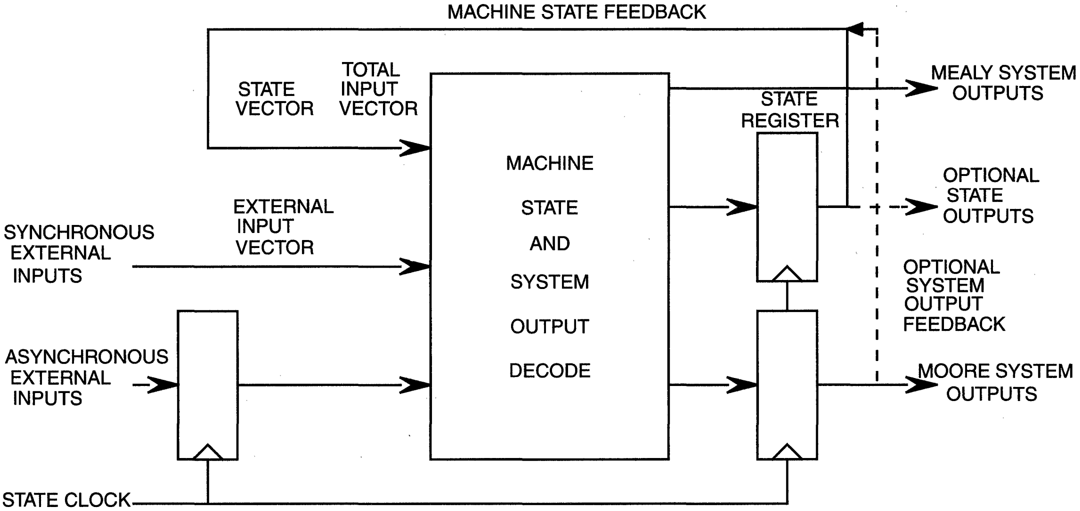


**Figure 2. Synchronous State Machine Block Diagram**

*CLK_1B:* The leading edge of this clock updates the instruction register.

*CLK_2A:* This clock's leading edge marks the start of ALU execution. The information on the ALU input bus clocks into the appropriate input registers at this time. The instruction cycle is considered recoverable up through and including CLK_2A (i.e., the status of the machine from the previous instruction has not been altered).

*CLK_2B:* Used to control the second half of the ALU execution stage, this clock initiates a write to RAM, triggers counters, gates ALU output into its latch, and clocks the ALU output information into any of the distributed destination registers.

*CLK_3A:* On this clock the memory address register can be updated. The ALU output bus status and ALU status is also clocked into the CPU status register.

## Clock Generator Inputs

A set of inputs (external stimulus to the state machine) controls the state machine. The clock state machine described here has eight external inputs, including the state machine clock. These inputs are:

*STATECLK:* The state machine clock.

*RESET:* An asynchronous or synchronous reset input that can be connected directly to the state registers' preset or clear or to all clocked register inputs (D or T input). If connected to the preset or clear, RESET need not be synchronized. In this case, RESET forces the state machine into the machine's initial state, regardless of the present state. RESET can result from any combination of the following sources:

- Power up circuit (system reset)
- System controller software decodes system reset
- System controller software decodes module reset
- CPU software decodes module reset

*RUN:* This signal controls the start and stop sequence of the CPU clocks. In PIPELINE RUN mode, the start sequence generates the proper clock progression to fill up the pipeline registers, and the stop sequence empties the pipeline. RUN is externally manipulated to implement the single step and breakpoint functions.

*NPL:* Used to select NONPIPELINED RUN vs. PIPELINED RUN modes, this signal must be set to the selected mode prior to activating the RUN signal. Setting NPL = 1 selects NONPIPELINED RUN mode, and NPL = 0 selects PIPELINED RUN mode. The single step function operates properly in NONPIPELINED RUN mode only.

*INTR:* This signal indicates an external interrupt. When INTR is received, and IEN (interrupt enable, described below) is active, the CPU executes its interrupt handler. An interrupt inhibits the instruction register update clock (CLK_1B) and the ALU update clock (CLK_2B). CLK_1A for the interrupt instruction executes on the next cycle. The interrupt condition has priority over a wait condition and therefore starts generating clocks to permit execution of the interrupt instructions.

*IEN:* This interrupt enable signal qualifies INTR. IEN is likely to be a bit in the instruction word, allowing the user to define sections of uninterruptable code.

*WAIT:* The wait condition is initiated when both WAIT and WEN (wait enable, described below) are active. The CPU remains in the wait condition until WAIT goes inactive.

*WEN:* This wait enable signal qualifies WAIT for entrance into the wait condition. Like IEN, WEN is usually a bit in the instruction word, allowing the user to define sections of wait-sensitive code.

## State Machine Partitioning

When architecting a state machine, it is generally a good practice to break up large machines into workable blocks, with each of the smaller machines containing states that require common inputs and generate common outputs. The example clock state machine is small enough to be designed as a single state machine, although it would be trivial to design logic to generate the free-running clocks as a separate machine from the rest of the clock state machine. Equations for the free-running clocks are:

$\text{CLK\_A} := \overline{\text{RESET}} \bullet \overline{\text{CLK\_A}}$

$\text{CLK\_B} := \overline{\text{RESET}} \bullet \text{CLK\_A}$

where ":=" indicates a registered output.

By examining these output equations, you can see that the free-running clocks have only two dependencies in common with the remaining portion of the clock state machine, i.e., RESET and STATECLK. The free-running clocks are required as inputs to the other state machine to synchronize the additional system outputs, however.

The example presented here implements the free-running clocks and the other system outputs within the same state definition. The resulting output equations can be verified against the equations for the free-running clocks alone.

### The Initial Machine State

Regardless of the preferred state machine entry method, attacking the problem starts with defining the initial state of the machine. This initial state (INIT in the example) must be consistent with the power-on condition and/or an external input used to initialize the machine (RESET).

The state of the machine can be decoded from the present values of the system outputs, state registers, or a combination of the two. (The advantages and disadvantages of the state definition options will be discussed in greater detail later in this application note.) The initial machine state is generally, but not always, a decode of all 0s or all 1s. In the example design, INIT is the decode of all 0s.

### Naming the States

With the exception of INIT, each state in the example design is named to indicate the active system clocks occurring during that state. For example, during state A, only CLK_A is active. Similarly, state 123B has only CLK_1B, CLK_2B, CLK_3B, and CLK_B active. Additionally, an "N" suffix designates a nonpipelined state and a "W" suffix designates a wait condition state; this convention differentiates between states with identical active system outputs.

### CPU Inactive States

The RESET input causes the state machine to enter the INIT state from any state in the machine. From the INIT state, the machine unconditionally starts to generate the free-running clocks. As shown in *Figure 3*, a line pointing from the INIT state to the A state, with a path equation equal to 1, indicates an unconditional branch. The state machine progression continues from the A state unconditionally into the B state. In the B state a multi-branch condition exists. If the RUN input remains inactive, then the A and B states continue to toggle, generating only the free-running clocks. Hence the INIT, A, and B states are referred to as "CPU inactive states."

### Nonpipelined States

If the NPL input is active while the RUN input becomes active, the state machine operates in NONPIPELINED RUN mode and follows the model portrayed in *Figure 4*.

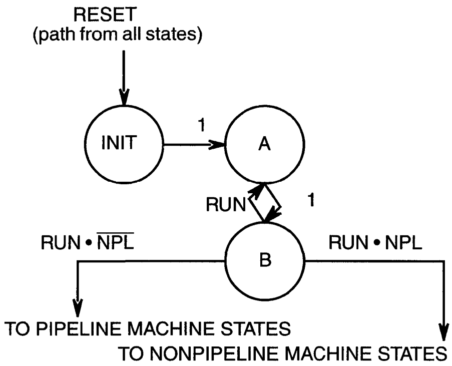


**Figure 3. CPU Inactive States**

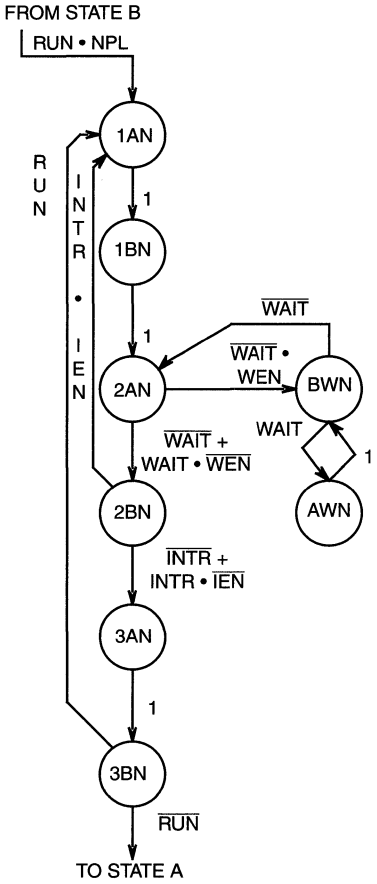


**Figure 4. Non-Pipelined States**

### Pipelined States

If the NPL input is inactive when the RUN input goes active, thus indicating PIPELINED RUN mode, the state machine operates as depicted in *Figure 5.*

### Unique States

When the RUN input goes active, the next state executed is either the 1A or the 1AN state, depending upon the value of the NPL input (refer to *Figures 4* and *5*). Notice that the active system outputs in these two states are identical. Why generate two identical states—when an additional state register might be required to differentiate between the states? (This assumes you use the system outputs to decode the machine's states.) The redundant states are not a problem because the additional state register needed to differentiate between the states is not an issue. There are two reasons for this. First, if you eliminate the redundant states, the state machine would require at least one additional state register anyway to differentiate between the B and the BW or BWN states, which would be needed without 1A and 1AN. (Separation of states BW and BWN from state B is required for correct functionality.) Second, adding another state only increases the number of state registers if the new total number of states exceeds an additional binary boundary (2, 4, 8, 16,...). This is not a problem here.

You might also choose to widen your state machine (increase the number of state registers) to reduce the number of product terms to the state or system output registers. This decision should take into account the desired circuit implementation (PLDs, PROMS, discrete hardware, etc.) and is often an

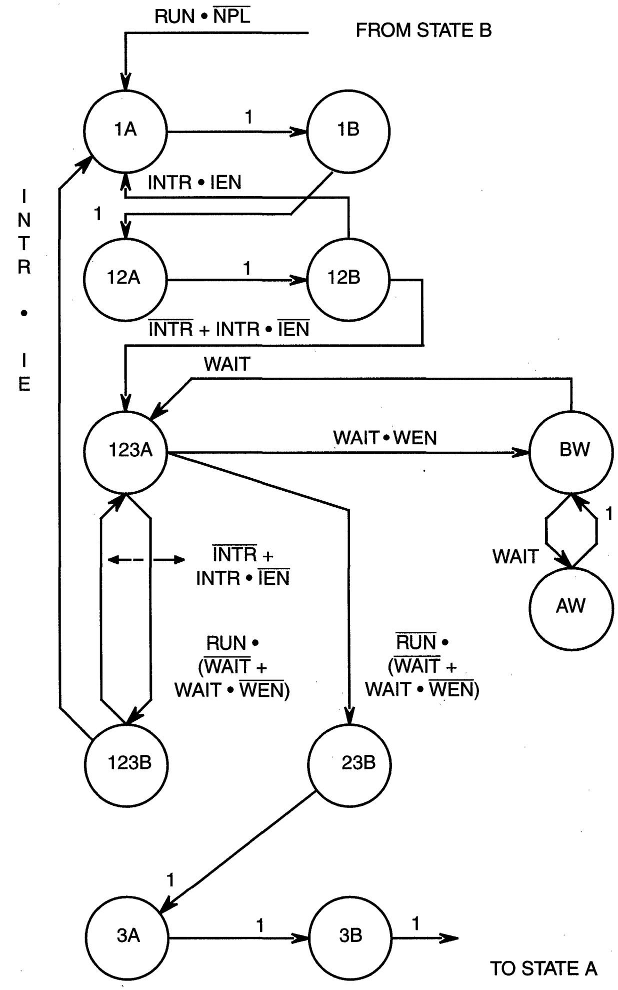


**Figure 5. Pipelined States**

iterative process. In general, you can initially architect the state machine in the manner that is the easiest for you to understand, then make additional changes or small adjustments later if they become necessary.

## State Description Verification

Now that all the pieces of the state machine are functionally defined (refer to *Figure 6* for the completed state diagram), consider methods for verifying the validity of the design. Some software you can use to describe and implement state machines would already offer verification at this point in a design. For other methods, read on!

One way to verify a state machine design is to recognize a rule of thumb: Out of every state, there should be a state path to another state for every possible combination of relevant external inputs. For example, there are two paths out of state 123B, with INTR and IEN as the relevant external inputs:

Path 1 = INTR • IEN

Path 2 = $\overline{\text{INTR}}$ + INTR • $\overline{\text{IEN}}$

If there are no known restrictions on the external inputs, a simple method of verifying the above rule of thumb is to generate an equation where all of the paths out of a state are ORed together as follows:

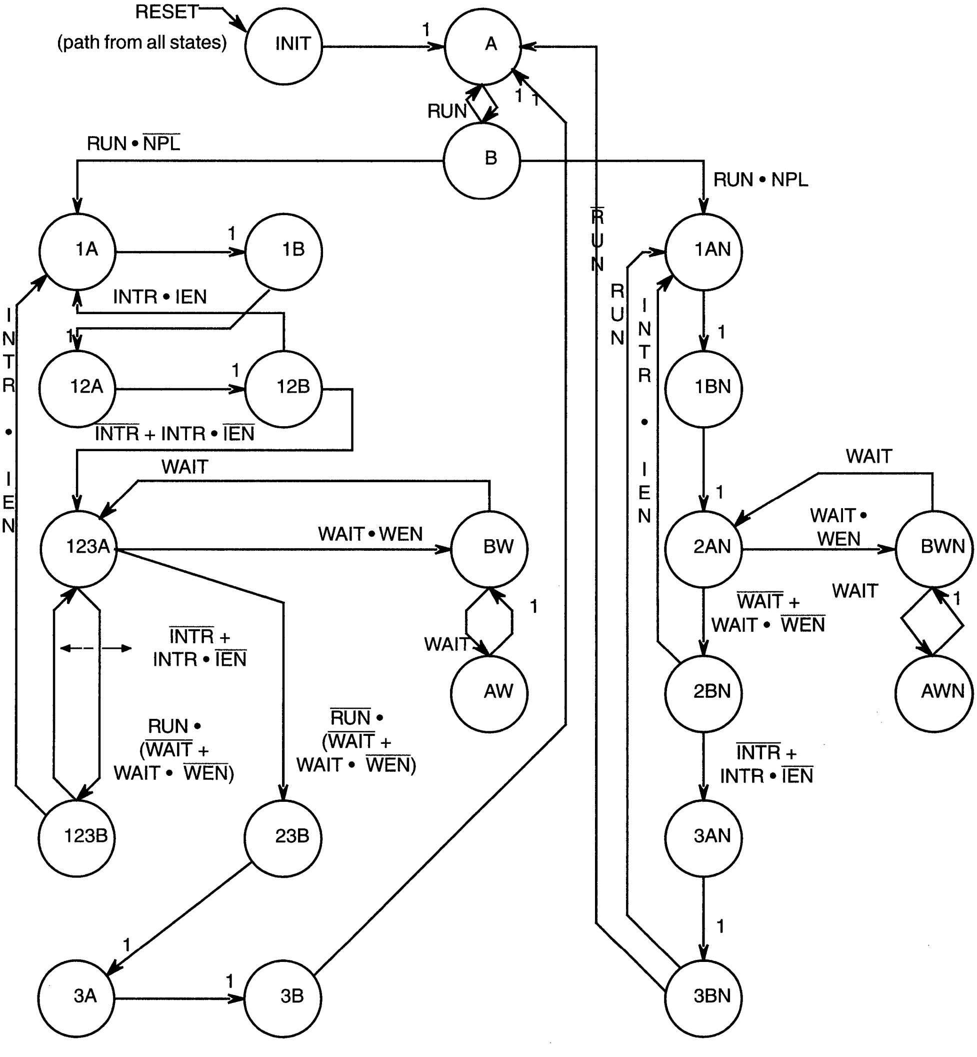

RESET
(path from all states)
INIT
1
A
RUN
1
1
RUN • $\overline{\text{NPL}}$
B
RUN • NPL
1A
1
1B
INTR • IEN
1
12A
1
12B
$\overline{\text{INTR}}$ + INTR • $\overline{\text{IEN}}$
INTR • IEN
WAIT
123A
WAIT • WEN
BW
1
WAIT
AW
$\overline{\text{INTR}}$ + INTR • $\overline{\text{IEN}}$
RUN • ($\overline{\text{WAIT}}$ + WAIT • $\overline{\text{WEN}}$)
$\overline{\text{RUN}}$ • ($\overline{\text{WAIT}}$ + WAIT • $\overline{\text{WEN}}$)
123B
23B
1
3A
1
3B
$\overline{\text{RUN}}$
RUN
INTR • IEN
1AN
1
1BN
1
WAIT
2AN
WAIT • WEN
BWN
WAIT
1
$\overline{\text{WAIT}}$ + WAIT • $\overline{\text{WEN}}$
2BN
AWN
$\overline{\text{INTR}}$ + INTR • $\overline{\text{IEN}}$
3AN
1
3BN

**Figure 6. CPU Clock State Machine**

$$\begin{aligned}
&\text{OUT\_STATE\_123B} && = \text{Path 1 + Path 2;}\\
&\text{OUT\_STATE\_123B} && = (\text{INTR} \bullet \text{IEN})\\
& && + \overline{\text{INTR}}\\
& && + (\text{INTR} \bullet \overline{\text{IEN}});\\
= \ &\text{OUT\_STATE\_123B} && = 1
\end{aligned}$$

If the equation's terms equal 1 after Boolean reduction, then every state path out of the state is accounted for. The main advantage to this verification method is that you can easily do it using readily available Boolean reduction software.

If there are known restrictions to the external inputs, you can use this information to reduce the complexity of the machine. If it is impossible for the INTR • $\overline{\text{IEN}}$ condition to occur externally, for example, then you can leave this condition out of the Path 2 equation. In that case, the reduction of the OUT_STATE_123B equation yields a non-1 result.

Because the method of verification just described does not detect redundant path equations, it is useful to revise the original rule of thumb to: Out of every state, there should be one and only one state path to another state for every possible combination of relevant external inputs.

This revised condition is not as easily verified as the original statement. The easiest way to verify the more restrictive case is to simulate the state machine. To do this, you must generate a test vector for every possible external input that is relevant to each state simulated. Automatic test vector generation programs are available that produce every possible combination. After running the vectors against the design, you must visually inspect the output to verify that the machine never enters an illegal state.

## System and State Register Output Generation

The model defining the clock state machine is complete, but there are still quite a few important decisions to be made regarding the final circuit implementation. Some of the major alternatives for final implementation are:

- System output vs. exclusive state register state decode
- D flip-flop vs. T flip-flop implementation
- PLD vs. PROM implementation

To gain some insight into these choices, consider how the output or feedback equations are assembled. Take, for example, the generation of CLK_3A using a D flip-flop (FF) implementation. By referring to *Figure 6*, you can find all the states in which CLK_3A is active. These are 123A, 3A, and 3AN. The CLK_3A output is generated by ORing the state decodes that, when ANDed with their respective state paths, advance the state machine into the three states listed above. Specifically:

$$\begin{aligned}
&\text{CLK\_3A :=} &&\\
&(\text{Decode of 12B})\bullet(\overline{\text{INTR}}+\text{INTR}\bullet\overline{\text{IEN}}) && ;\text{-123A}\\
&+(\text{Decode of BW})\bullet(\overline{\text{WAIT}}) && ;\text{-123A}\\
&+(\text{Decode of 23B})\bullet(1) && ;\text{-3A}\\
&+(\text{Decode of 2BN})\bullet(\overline{\text{INTR}}+\text{INTR}\bullet\overline{\text{IEN}}) && ;\text{-3AN}
\end{aligned}$$

When you define the state decodes, the CLK_3A equations are completely specified in terms of the state machine inputs (state path), state registers, and/or system outputs (state decode). Typically, you then multiply the equation out to form a sum of products. This format provides for easy implementation in a PLD, which has a sum-of-products architecture, and also provides a useful foundation for further equation reduction.

### State Decode

As discussed earlier, the next state of the machine can be decoded from the present values of the system outputs, the state registers, or a combination of the two. The choice typically comes down to weighing the maximum number of product terms verses the maximum number of flip-flops available in an implementation. For a Moore machine with registered system outputs, using the system outputs to uniquely define the states uses the smallest number of flip-flops to define the state machine. However, it is often necessary to add one or more state registers to uniquely define the states.

State assignment for this state decoding method is quite simple, but also rigidly defined, allowing limited flexibility when assigning the additional state registers. After reduction, the feedback and output equations of this "narrow" state machine might contain too many product terms to be implemented in a specific PLD, although product term complexity is never a problem with a PROM implementation.

### Exclusive State Registers

Another consideration in state machine design is that you might be able to distribute the number of product terms more evenly among the equations implementing the state machine by using state registers exclusively to decode the states. Because the state decodes in the state registers can be selected to assist in Boolean reduction, proper state assignment enables the more complex equations to fit into a specific implementation.

This type of decode is useful in a PLD implementation, where there is a shortage of product terms for a specific state flip-flop, but extra flip-flops are available. Adding an extra state register can simplify the decode logic enough to fit the design in a single PLD.

The total number of exclusive state registers required to implement a state machine varies from a minimum of LOG(2)X (rounded up to the nearest integer) to a maximum of X, where X is the total number of states in the machine. You can iteratively change this number, along with the state assignment, to obtain a suitable solution.

The state assignment itself is a non-trivial issue, with almost limitless possibilities and no known method of obtaining the optimal solution. There are, however, some guidelines that can be used to obtain workable solutions:

1. Two or more states that potentially enter the same state with identical path equations should be adjacent (their binary codes differ in exactly one position). As an example, refer to *Figure 5*. States 12B and 123B both proceed into state 1A if the path condition INTR • IEN is true. When generating the CLK_1A equation, two of the terms of the equation look like this:

$$\begin{aligned}
&\text{CLK\_1A :=} &&\\
&\quad(\text{Decode of 12B}) \bullet (\text{INTR} \bullet \text{IEN}) && ;\text{-1A}\\
&+ \ (\text{Decode of 123B}) \bullet (\text{INTR} \bullet \text{IEN}) && ;\text{-1A}\\
&\quad\cdot \quad \cdot \quad \cdot &&
\end{aligned}$$

If the decode of 12B and 123B differ in exactly one position, then Boolean reduction (which uses the A•B + $\overline{\text{A}}$•B = B relationship) converts the two product terms into one smaller product term.

2. Two or more states that might proceed into different states with identical path equations, and an identical active out-

put, should be adjacent. This situation occurs in the previous CLK_3A equation, shown again here:

$$\begin{array}{ll}
\texttt{CLK\_3A :=} & \\
\texttt{(Decode of 12B)}\bullet\texttt{(}\overline{\texttt{INTR}}\texttt{+INTR}\bullet\overline{\texttt{IEN}}\texttt{)} & \texttt{;-123A} \\
\texttt{+(Decode of BW)}\bullet\texttt{(}\overline{\texttt{WAIT}}\texttt{)} & \texttt{;-123A} \\
\texttt{+(Decode of 23B)}\bullet\texttt{(1)} & \texttt{;-3A} \\
\texttt{+(Decode of 2BN)}\bullet\texttt{(}\overline{\texttt{INTR}}\texttt{+INTR}\bullet\overline{\texttt{IEN}}\texttt{);} & \texttt{-3AN}
\end{array}$$

Note that if states 12B and 2BN are adjacent, then you can reduce the CLK_3A equation to three product terms.

## Clock Generator Implementation

As mentioned earlier, there are many ways to implement state machines. The following sections discuss some of the pros and cons associated with some of the more common state machine implementations.

### D Flip-Flop Implementation

There are more products available that support a D flip-flop solution than any other implementation. Therefore, it is usually the most cost-effective solution for a state machine.

*Table 1* lists the number of product terms per output obtained by compiling the clock generator state machine definition with the LOG/iC software, using D flip-flops. The compiler input file appears in Appendix A. Optimizing the design (*Table 2*) significantly reduces the number of product terms needed.

**Table 1. Non-optimized Results for Clock Generator: D Flip-Flop Implementation**

| Log/IC Optimization Summary (FACT) | | | | |
|---|---|---|---|---|
| **CPU Time Quota per Function: 100 sec** | | | | |
| **Function** | **INV** | **P-Terms** | **CPU-Time** | **Flags** |
| CLK_1A.D | No | 12 | <1 | N |
| | Yes | 27 | <1 | N |
| CLK_1B.D | No | 5 | <1 | N |
| | Yes | 34 | 1 | N |
| CLK_2A.D | No | 8 | <1 | N |
| | Yes | 31 | <1 | N |
| CLK_2B.D | No | 7 | <1 | N |
| | Yes | 32 | <1 | N |
| CLK_3A.D | No | 8 | <1 | N |
| | Yes | 31 | <1 | N |
| CLK_3B.D | No | 6 | <1 | N |
| | Yes | 33 | <1 | N |
| CLK_A.D | No | | | NT |
| | Yes | | | NT |
| QQ1.D | No | 6 | <1 | N |
| | Yes | 5 | <1 | N |
| QQ2.D | No | 10 | <1 | N |
| | Yes | 9 | <1 | N |

N: No Optimization
T: Trivial Function
FACT Minimization: 11 sec

**Table 2. Optimized Results for Clock Generator: D Flip-Flop Implementation**

| Log/IC Optimization Summary (FACT) | | | | |
|---|---|---|---|---|
| **CPU Time Quota per Function: 100 sec** | | | | |
| **Function** | **INV** | **P-Terms** | **CPU-T ime** | **Flags** |
| CLK_1A.D | No | 6 | 1 | |
| | Yes | 11 | 2 | |
| CLK_1B.D | No | 3 | 1 | |
| | Yes | 4 | <1 | |
| CLK_2A.D | No | 4 | 1 | |
| | Yes | 7 | <1 | |
| CLK_2B.D | No | 3 | 1 | |
| | Yes | 4 | <1 | |
| CLK_3A.D | No | 4 | 1 | |
| | Yes | 9 | 1 | |
| CLK_3B.D | No | 3 | <1 | |
| | Yes | 3 | 1 | |
| CLK_A.D | No | 1 | <1 | |
| | Yes | 2 | <1 | |
| CLK_B.D | No | 1 | 1 | |
| | Yes | 2 | <1 | |
| QQ1.D | No | 3 | <1 | |
| | Yes | 3 | 1 | |
| QQ2.D | No | 6 | 16 | |
| | Yes | 6 | 2 | |

FACT Minimization: 29 sec

### T Flip-Flop Implementation

Even though D flip-flop solutions are more widely available, there are times when the logic needed for this implementation is prohibitively complex. Under these circumstances, a T flip-flop implementation might be more cost effective, because using T flip-flops reduces the logic significantly.

The best example of this situation is a simple synchronous binary counter. While the most significant bit (MSB) of an N-bit counter in a D flip-flop implementation requires N product terms, the T flip-flop solution requires only one product term. Note that the Cypress family of CY7C33x devices offers you a configurable T- or D-type implementation if you place an XOR gate prior to the D flip-flop; route the AND/OR array to one of the XOR's inputs and the flip-flop's Q output (via an additional product term) to the other XOR input.

It isn't clear from simple observation, however, whether the T flip-flop implementation is beneficial for the clock generator state machine. One way to clarify this question is to change three command lines in the state machine description shown in Appendix A and recompile to produce a T flip-flop implementation. *Table 3* contains the product term results using T flip-flops. A quick study of the results reveals that the optimized version using D flip-flops (*Table 2*) requires fewer product terms than the T flip-flop version.

**Table 3. Optimized Results for Clock Generator: T Flip-Flop Implementation**

| LOG/iC Optimization Summary (FACT) | | | | |
|---|---|---|---|---|
| CPU Time Quota per Function: 100 sec | | | | |
| **Function** | **INV** | **P-Terms** | **CPU-Time** | **Flags** |
| CLK_1A.T | No | 6 | <1 | |
| | Yes | 7 | 1 | |
| CLK_1B.T | No | 4 | 1 | |
| | Yes | 3 | 1 | |
| CLK_2A.T | No | 5 | 1 | |
| | Yes | 4 | <1 | |
| CLK_2B.T | No | 4 | 1 | |
| | Yes | 3 | <1 | |
| CLK_3A.T | No | 5 | <1 | |
| | Yes | 6 | 2 | |
| CLK_3B.T | No | 4 | <1 | |
| | Yes | 2 | <1 | |
| CLK_A.T | No | | | C |
| | Yes | | | C |
| CLK_B.T | No | 2 | 1 | |
| | Yes | 1 | <1 | |
| QQ1.T | No | 3 | <1 | |
| | Yes | 5 | 1 | |
| QQ2.T | No | 6 | <1 | |
| | Yes | 11 | 2 | |

### PLD Implementation

With the LOG/iC PLD Database option, the software assists in selecting a PLD, and it shows that the non-optimized version of the clock state machine fits in a PALCE22V10 without further reduction. If the equations are reduced using Boolean reduction, however, a lower-cost solution is available. The results shown in *Table 3* indicate that the less expensive PLDC20G10 would work. Appendix A shows the listing for the 20G10 LOG/iC implementation. Waveforms for the completed design appear in Appendix B. You can verify the CLK_A and CLK_B equation results against the equations generated in the State Machine Partitioning section of this application note.

### PROM Implementation

You can obtain very high speed solutions by implementing state machines using PROMs. A PROM uses a look-up table to decode the machine's next state, as opposed to the AND/OR array in a PLD. The main advantage of using a look-up table to decode the next state is that every combination of the inputs can be decoded. Thus, you can create an extremely complex machine, without equation reductions.

The look-up table's drawback is that the PROM's depth grows exponentially (2N, where N = # of inputs to the look-up table) with every additional input to the look-up table. To determine the depth required, notice that the present total input vector provides the inputs to the look-up table. The clock generator state machine has seven external inputs, six system outputs, and two state outputs, which indicates a feasible implementation using the CY7C277 (32K x 8) registered PROM.

Using a registered PROM such as the CY7C277 to implement the machine also helps to reduce the parts count, because the PROM implements both the state and system output registers. LOG/iC offers support for implementing state machines in PROMs, and only a few minor changes to the state machine description shown in Appendix A are required. *PROM replaces the *PAL command, some simple statements indicating the CY7C277 architecture (INPUTS = 15 AND OUTPUTS = 8) replaces the TYPE = statement, and PROGFORMAT = INTEL–HEX.

## Reference

1. Donald D. Givone, Introduction to Switching Circuit Theory (New York: McGraw-Hill, Inc., 1970)

## Appendix A. LOG/iC PLD Source Code: Clock State Machine

```
LOG/iC-PAL             Rel 3.2/2-2328-1721/00034 #32-5955 90/03/15  23:49:45
----------------------------------------------------------------------------
LOG/iC - COPYRIGHT (C) 1985,1988 BY ISDATA GMBH, 7500 KARLSRUHE WEST-GERMANY
Cypress Semiconductor                                 LICENCE FOR IBM-PC/XT/AT
Data Set: OD20G10.DCB
  1      1: *IDENTIFICATION
  2      2: PIPELINED CLOCKING SYSTEM OD20G10
  3      3: ERIC B.  ROSS
  4      4: CYPRESS SEMICONDUCTOR
  5      5: NAMING CONVENTION
  6      6:  OD      = SYSTEM OUTPUTS ARE DFLOPS AND ARE USED FOR STATE DEF
  7      7:  20G10   = PALC20G10 IMPLEMENTATION
  8      8: *PAL
  9      9:  TYPE=PALC20G10
 10 I   10:
 11     11: *X-NAMES
 12 I   12: ;---------------------------------------------------------------------
 13 I   13: ;INPUT DEFINITIONS :
 14 I   14: ; RUN     = START & STOP EXECUTION OF OUTPUT CLOCKS (NORMAL, SINGLE
 15 I   15: ;           STEP, & BREAK PT.  EXECUTION
 16 I   16: ; NPL     = PIPELINED VS NON-PIPELINED MODE OF EXECUTION
 17 I   17: ; INTR    = EXTERNAL INTERRUPT CONDITION (TLB MISS, PARITY ERROR,...)
 18 I   18: ; IEN     = INTERRUPT ENABLE
 19 I   19: ; WAIT    = WAIT ENABLE (CACHE MISS)
 20 I   20: ; WEN     = WAIT ENABLE
 21 I   21: ;---------------------------------------------------------------------
 22 I   22: ;
 23     23:  RUN, NPL, INTR, IEN, WAIT, WEN, RESET;
 24 I   24:
 25     25: *Z-NAMES
 26 I   26: ;---------------------------------------------------------------------
 27 I   27: ;OUTPUT DEFINITIONS :
 28 I   28: ;
 29 I   29: ; 3 CLOCK STAGES 1, 2, 3
 30 I   30: ; 2 CLOCKS PER STATE A, B
 31 I   31: ;    CLK_XX WHERE XX = 1A,1B,2A,2B,3A,3B
 32 I   32: ;
 33 I   33: ; 2 FREE RUNNING CLOCKS
 34 I   34: ;    CLK_A, CLK_B
 35 I   35: ;
 36 I   36: ; ADDITIONAL REGISTERS FOR STATE DEFINITION
 37 I   37: ;    QQ1, QQ2
 38 I   38: ;---------------------------------------------------------------------
 39 I   39: ;
 40     40: CLK_1A, CLK_1B, CLK_2A, CLK_2B, CLK_3A, CLK_3B, CLK_A, CLK_B, QQ1, QQ2;
 41 I   41:
 42     42: *Z-VALUES
 43 I   43:
 44 I   44: ;                         ADDITIONAL OUTPUTS
 45 I   45: ;      SYSTEM OUTPUTS     FOR STATE DEFINITION
 46 I   46: ;      _______________    ____________________
 47 I   47: ;
 48 I   48: ;      C C C C C C C C    Q Q
 49 I   49: ;      L L L L L L L L    Q Q
 50 I   50: ;      K K K K K K K K    1 2
 51 I   51: ;      1 1 2 2 3 3 A B
 52 I   52: ;      A B A B A B
 53 I   53:
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
 54     54:  S1  = 0 0 0 0 0 0 0 0   - -           ; INIT  COMMON STATES
 55     55:  S2  = 0 0 0 0 0 0 1 0   0 -           ; SA       - INACTIVE
 56     56:  S3  = 0 0 0 0 0 0 0 1   0 -           ; SB         MODE STATES
 57 I   57:
 58     58:  S4  = 1 0 0 0 0 0 1 0   - 0           ; S1A    PIPELINE STATES
 59     59:  S5  = 0 1 0 0 0 0 0 1   - 0           ; S1B
 60     60:  S6  = 1 0 1 0 0 0 1 0   - -           ; S12A
 61     61:  S7  = 0 1 0 1 0 0 0 1   - -           ; S12B
 62     62:  S8  = 1 0 1 0 1 0 1 0   - -           ; S123A
 63     63:  S9  = 0 1 0 1 0 1 0 1   - -           ; S123B
 64     64:  S10 = 0 0 0 1 0 1 0 1   - -           ; S23B 65     65:  S11 = 0 0 0 0 1 0 1 0   - 0
 66     66:  S12 = 0 0 0 0 0 1 0 1   - 0           ; S3B
 67     67:  S13 = 0 0 0 0 0 0 1 0   1 0           ; SAW
 68     68:  S14 = 0 0 0 0 0 0 0 1   1 0           ; SBW
 69 I   69:
 70     70:  S15 = 1 0 0 0 0 0 1 0   - 1           ; S1AN   NON-PIPELINE
 71     71:  S16 = 0 1 0 0 0 0 0 1   - 1           ; S1BN
 72     72:  S17 = 0 0 1 0 0 0 1 0   - -           ; S2AN
 73     73:  S18 = 0 0 0 1 0 0 0 1   - -           ; S2BN
 74     74:  S19 = 0 0 0 0 1 0 1 0   - 1           ; S3AN
 75     75:  S20 = 0 0 0 0 0 1 0 1   - 1           ; S3BN
 76     76:  S21 = 0 0 0 0 0 0 1 0   1 1           ; SAWN
 77     77:  S22 = 0 0 0 0 0 0 0 1   1 1           ; SBWN
 78 I   78:
 79     79: *STRING
 80     80:  INIT    = 1                           ;  COMMON STATES
 81     81:  SA      = 2                           ;   -INACTIVE MODE
 82     82:  SB      = 3                           ;    STATES
 83 I   83:
 84     84:  S1A     = 4                           ;  PIPELINE STATES
 85     85:  S1B     = 5                           ;
 86     86:  S12A    = 6                           ;
 87     87:  S12B    = 7                           ;
 88     88:  S123A   = 8                           ;
 89     89:  S123B   = 9                           ;
 90     90:  S23B    = 10                          ;
 91     91:  S3A     = 11                          ;
 92     92:  S3B     = 12                          ;
 93     93:  SAW     = 13                          ;
 94     94:  SBW     = 14                          ;
 95 I   95:
 96     96:  S1AN    = 15                          ;  NON-PIPELINE
 97     97:  S1BN    = 16                          ;
 98     98:  S2AN    = 17                          ;
 99     99:  S2BN    = 18                          ;
100    100:  S3AN    = 19                          ;
101    101:  S3BN    = 20                          ;
102    102:  SAWN    = 21                          ;
103    103:  SBWN    = 22                          ;
104    104:  LASTSTATE = 22;
105 I  105:
106    106: *FLOW-TABLE
107 I  107: ;
108 I  108: ;-------------------------------------------------------------------
109 I  109: ;RESET STATE
110 I  110: ;ALL STATES MUST RESET TO THE INITIAL STATE (ALL OUTPUTS REGISTERS 0)
UPON
111 I  111: ;AN ACTIVE RESET INPUT.  SINCE THE 20G10 HAS NO GLOBAL OR INDIVIDUAL
112 I  112: ;RESETS TO THE OUTPUT REGISTERS, RESET TO INITIAL STATE MUST BE EMBEDDED
113 I  113: ;INTO THE STATE MACHINE
114 I  114: ;
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
115    115: RELEVENT = RESET                           ;
116    116: S[1..'LASTSTATE'], X 1      , F 'INIT'  ;ALL STATE > INIT UPON RESET
117    138: RELEVENT = RESET = 0                       ;
118 I  139: ;
119 I  140: ;-------------------------------------------------------------------
120 I  141: ;INACTIVE MODE STATES
121    142: RELEVANT = RUN, NPL                        ;
122    143: S 'INIT'    , X - -         , F 'SA'    ;INITIAL STATE AFTER RESET
123 I  144:
124    145: S 'SA'      , X - -         , F 'SB'    ;INACTIVE MODE STATE, ONLY
125 I  146:
126    147: S 'SB'      , X 0 -         , F 'SA'    ;FREE RUN CLKS A & B ARE ACTIVE
127    148:               X 1 0         , F 'S1A'   ; PIPELINE VS.
128    149:               X 1 1         , F 'S1AN'  ; NON-PIPELINE DECISION
129 I  150:
130 I  151: ;-------------------------------------------------------------------
131 I  152: ;PIPELINE MODE STATES
132 I  153:
133    154: RELEVANT = INTR, IEN                       ;*PRIMING THE PIPELINE*
134    155: S 'S1A'     , X - -         , F 'S1B'   ;
135 I  156:
136    157: S 'S1B'     , X - -         , F 'S12A'  ;
137 I  158:
138    159: S 'S12A'    , X - -         , F 'S12B'  ;
139 I  160:
140    161: S 'S12B'    , X 1 1         , F 'S1A'   ;   INTERRUPT CONDITION ? YES
141    162:               X 1 0         , F 'S123A' ;                         NO
142    163:               X 0 -         , F 'S123A' ;                         NO
143 I  164:
144    165: RELEVANT =  RUN, INTR, IEN, WAIT, WEN; *FULL PIPELINE*
145    166: S 'S123A'   , X - - - 1 1 , F 'SBW'     ;   WAIT CONDITION
146    167:               X 0 - - 0 - , F 'S23B'    ;   /RUN COND., EMPTY PIPELINE
147    168:               X 0 - - 1 0 , F 'S23B'    ;   /RUN COND., EMPTY PIPELINE
148    169:               X 1 - - 0 - , F 'S123B'   ;   RUN CONDITION
149    170:               X 1 - - 1 0 , F 'S123B'   ;   RUN CONDITION
150 I  171:
151    172: S 'S123B'   , X - 1 1 - - , F 'S1A'     ;   INTERUPT CONDITION
152    173:               X - 0 - - - , F 'S123A'   ;   RUN CONDITION
153    174:               X - 1 0 - - , F 'S123A'   ;   RUN CONDITION
154 I  175:
155    176: RELEVANT = RUN                             ; *EMPTY PIPELINE*
156    177: S 'S23B'    , X -           , F 'S3A'   ;
157 I  178:
158    179: S 'S3A'     , X -           , F 'S3B'   ;
159 I  180:
160    181: S 'S3B'     , X -           , F 'SA'    ;   BACK TO INACTIVE STATE
161 I  182:
162    183: RELEVANT = WAIT                            ; *PIPELINE WAIT STATES*
163    184: S 'SBW'     , X 1           , F 'SAW'   ;   WAIT
164    185:               X 0           , F 'S123A' ;   /WAIT
165 I  186:
166    187: S 'SAW'     , X -           , F 'SBW'   ;
167 I  188:
168 I  189: ;-------------------------------------------------------------------
169 I  190: ;NON-PIPELINE MODE STATES
170 I  191:
171    192: S 'S1AN'    , X -           , F 'S1BN'  ;
172 I  193:
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
173     194: S 'S1BN'    , X -            , F 'S2AN'  ;
174 I 195:
175     196: RELEVANT = WAIT, WEN                     ;
176     197: S 'S2AN'    , X 1 1        , F 'SBWN'    ; WAIT CONDITION
177     198:               X 0 -        , F 'S2BN'    ; /WAIT CONDITION
178     199:               X 1 0        , F 'S2BN'    ; /WAIT CONDITION
179 I 200:180    201: RELEVANT = INTR, IEN                      ;
181     202: S 'S2BN'    , X 1 1        , F 'S1AN'    ; INTERRUPT CONDITION
182     203:               X 0 -        , F 'S3AN'    ; /INTERRUPT CONDITION
183     204:               X 1 0        , F 'S3AN'    ; /INTERRUPT CONDITION
184 I 205:
185     206: RELEVANT = RUN                           ;
186     207: S 'S3AN'    , X -            , F 'S3BN'  ;
187 I 208:
188     209: S 'S3BN'    , X 1            , F 'S1AN'  ;
189     210:               X 0            , F 'SA'    ; BACK TO INACTIVE STATE
190 I 211:
191     212: RELEVANT = WAIT                          ;*NON-PIPELINED WAIT STATES*
192     213: S 'SBWN'    , X 1            , F 'SAWN'  ;  REMAIN IN WAIT
193     214:               X 0            , F 'S2AN'  ;  END OF WAIT CONDITION
194 I 215:
195     216: S 'SAWN'    , X -            , F 'SBWN'  ;  REMAIN IN WAIT
196 I 217:
197     218: *STATE-ASSIGNMENT
198     219:  Z-VALUES
199 I 220:
200 I 221:
201     222: *PIN
202     223:  STATECLK = 1, RUN = 2, NPL = 3, INTR = 4, IEN = 5, WAIT = 6, WEN = 7,
203     223:  RESET = 8, CLK_1A = 14, CLK_1B = 15, CLK_2A = 16, CLK_2B = 17,
204     223:  CLK_3A = 18, CLK_3B = 19, CLK_A = 20, CLK_B = 21, QQ1 = 22, QQ2 = 23;
205 I 224:
206     225: *RUN-CONTROL
207     226:  LISTING= LONG,SYMBOL-TABLE,EQUATIONS,PINOUT;
208     227:  PROGFORMAT= L-EQUATIONS
209     228:  OPTIMIAZATION= P-TERMS;
210     229: *END
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

| LOG/IC SYMBOL TABLE | | | | |
|---|---|---|---|---|
| SYMBOL | TYPE | REG | LEVEL | PIN/NODE |
| GND | LOCAL | - | HIGH | |
| VCC | LOCAL | - | HIGH | |
| RUN | X-VARIABLE | - | HIGH | 2 |
| NPL | X-VARIABLE | - | HIGH | 3 |
| INTR | X-VARIABLE | - | HIGH | 4 |
| IEN | X-VARIABLE | - | HIGH | 5 |
| WAIT | X-VARIABLE | - | HIGH | 6 |
| WEN | X-VARIABLE | - | HIGH | 7 |
| RESET | X-VARIABLE | - | HIGH | 8 |
| CLK_1A | X-VARIABLE | - | HIGH | 14 |
| CLK_1B | X-VARIABLE | - | HIGH | 15 |
| CLK_2A | X-VARIABLE | - | HIGH | 16 |
| CLK_2B | X-VARIABLE | - | HIGH | 17 |
| CLK_3A | X-VARIABLE | - | HIGH | 18 |
| CLK_3B | X-VARIABLE | - | HIGH | 19 |
| CLK_A | X-VARIABLE | - | HIGH | 20 |
| CLK_B | X-VARIABLE | - | HIGH | 21 |
| QQ1 | X-VARIABLE | - | HIGH | 22 |
| QQ2 | X-VARIABLE | - | HIGH | 23 |
| CLK_1A.D | Z-VARIABLE | DFF | HIGH | 14 |
| CLK_1B.D | Z-VARIABLE | DFF | HIGH | 15 |
| CLK_2A.D | Z-VARIABLE | DFF | HIGH | 16 |
| CLK_2B.D | Z-VARIABLE | DFF | HIGH | 17 |
| CLK_3A.D | Z-VARIABLE | DFF | HIGH | 18 |
| CLK_3B.D | Z-VARIABLE | DFF | HIGH | 19 |
| CLK_A.D | Z-VARIABLE | DFF | HIGH | 20 |
| CLK_B.D | Z-VARIABLE | DFF | HIGH | 21 |
| QQ1.D | Z-VARIABLE | DFF | HIGH | 22 |
| QQ2.D | Z-VARIABLE | DFF | HIGH | 23 |

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
EXPANDED FUNCTION TABLE (INCLUDING LOCAL VARIABLES):
----------------------------------------------------
                             :  CCCC CC
                             :  LLLL LLCC
               CCC CCC       :  KKKK KKLL
              RLLL LLLC C    :  ____ __KK QQ
         I W  EKKK KKKL L    :  1122 33__ QQ
    GVRN NIAW S___ ___K KQQ  :  ABAB ABAB 12
    NCUP TEIE E112 233_ _QQ  :  ....  ....  ..
    DCNL RNTN TABA BABA B12  :  DDDD DDDD DD
    ----------------------------------------
    ---- ---- 1000 0000 0--  :  0000 0000 --;     1/   116
    ---- ---- 0000 0000 0--  :  0000 0010 0-;     2/   143
    ---- ---- 1000 0001 00-  :  0000 0000 --;     3/   117
    ---- ---- 0000 0001 00-  :  0000 0001 0-;     4/   145
    ---- ---- 1000 0000 10-  :  0000 0000 --;     5/   118
    --0- ---- 0000 0000 10-  :  0000 0010 0-;     6/   147
    --10 ---- 0000 0000 10-  :  1000 0010 -0;     7/   148
    --11 ---- 0000 0000 10-  :  1000 0010 -1;     8/   149
    ---- ---- 1100 0001 0-0  :  0000 0000 --;     9/   119
    ---- ---- 0100 0001 0-0  :  0100 0001 -0;    10/   155
    ---- ---- 1010 0000 1-0  :  0000 0000 --;    11/   120
    ---- ---- 0010 0000 1-0  :  1010 0010 --;    12/   157
    ---- ---- 1101 0001 0--  :  0000 0000 --;    13/   121
    ---- ---- 0101 0001 0--  :  0101 0001 --;    14/   159
    ---- ---- 1010 1000 1--  :  0000 0000 --;    15/   122
    ---- 11-- 0010 1000 1--  :  1000 0010 -0;    16/   161
    ---- 10-- 0010 1000 1--  :  1010 1010 --;    17/   162
    ---- 0--- 0010 1000 1--  :  1010 1010 --;    18/   163
    ---- ---- 1101 0101 0--  :  0000 0000 --;    19/   123
    ---- --11 0101 0101 0--  :  0000 0001 10;    20/   166
    --0- --0- 0101 0101 0--  :  0001 0101 --;    21/   167
    --0- --10 0101 0101 0--  :  0001 0101 --;    22/   168
    --1- --0- 0101 0101 0--  :  0101 0101 --;    23/   169
    --1- --10 0101 0101 0--  :  0101 0101 --;    24/   170
    ---- ---- 1010 1010 1--  :  0000 0000 --;    25/   124
    ---- 11-- 0010 1010 1--  :  1000 0010 -0;    26/   172
    ---- 0--- 0010 1010 1--  :  1010 1010 --;    27/   173
    ---- 10-- 0010 1010 1--  :  1010 1010 --;    28/   174
    ---- ---- 1000 1010 1--  :  0000 0000 --;    29/   125
    ---- ---- 0000 1010 1--  :  0000 1010 -0;    30/   177
    ---- ---- 1000 0101 0-0  :  0000 0000 --;    31/   126
    ---- ---- 0000 0101 0-0  :  0000 0101 -0;    32/   179
    ---- ---- 1000 0010 1-0  :  0000 0000 --;    33/   127
    ---- ---- 0000 0010 1-0  :  0000 0010 0-;    34/   181
    ---- ---- 1000 0001 010  :  0000 0000 --;    35/   128
    ---- ---- 0000 0001 010  :  0000 0001 10;    36/   187
    ---- ---- 1000 0000 110  :  0000 0000 --;    37/   129
    ---- --1- 0000 0000 110  :  0000 0010 10;    38/   184
    ---- --0- 0000 0000 110  :  1010 1010 --;    39/   185
    ---- ---- 1100 0001 0-1  :  0000 0000 --;    40/   130
    ---- ---- 0100 0001 0-1  :  0100 0001 -1;    41/   192
    ---- ---- 1010 0000 1-1  :  0000 0000 --;    42/   131
    ---- ---- 0010 0000 1-1  :  0010 0010 --;    43/   194
    ---- ---- 1001 0001 0--  :  0000 0000 --;    44/   132
    ---- --11 0001 0001 0--  :  0000 0001 11;    45/   197
    ---- --0- 0001 0001 0--  :  0001 0001 --;    46/   198
    ---- --10 0001 0001 0--  :  0001 0001 --;    47/   199
    ---- ---- 1000 1000 1--  :  0000 0000 --;    48/   133
    ---- 11-- 0000 1000 1--  :  1000 0010 -1;    49/   202
    ---- 0--- 0000 1000 1--  :  0000 1010 -1;    50/   203
    ---- 10-- 0000 1000 1--  :  0000 1010 -1;    51/   204
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
  ---- ---- 1000 0101 0-1  :  0000 0000 --;    52/  134
  ---- ---- 0000 0101 0-1  :  0000 0101 -1;    53/  207
  ---- ---- 1000 0010 1-1  :  0000 0000 --;    54/  135
  --1- ---- 0000 0010 1-1  :  1000 0010 -1;    55/  209
  --0- ---- 0000 0010 1-1  :  0000 0010 0-;    56/  210
  ---- ---- 1000 0001 011  :  0000 0000 --;    57/  136
  ---- ---- 0000 0001 011  :  0000 0001 11;    58/  216
  ---- ---- 1000 0000 111  :  0000 0000 --;    59/  137
  ---- --1- 0000 0000 111  :  0000 0010 11;    60/  213
  ---- --0- 0000 0000 111  :  0010 0010 --;    61/  214
  REST                     :  ---- ---- --;    62
  ----------------------------------------
  1234 5678 9012 3456 789     1234 5678 90


 STATE ASSIGNMENT:
-----------------
  CCCC CC
  LLLL LLCC
  KKKK KKLL
  ____ __KK QQ
  1122 33__ QQ
  ABAB ABAB 12
  ------------
  0000 0000 --; 1
  0000 0010 0-; 2
  0000 0001 0-; 3
  1000 0010 -0; 4
  0100 0001 -0; 5
  1010 0010 --; 6
  0101 0001 --; 7
  1010 1010 --; 8
  0101 0101 --; 9
  0001 0101 --; 10
  0000 1010 -0; 11
  0000 0101 -0; 12
  0000 0010 10; 13
  0000 0001 10; 14
  1000 0010 -1; 15
  0100 0001 -1; 16
  0010 0010 --; 17
  0001 0001 --; 18
  0000 1010 -1; 19
  0000 0101 -1; 20
  0000 0010 11; 21
  0000 0001 11; 22
EXPANDED FUNCTION TABLE (LOCAL VARIABLES REMOVED):
--------------------------------------------------
                          :  CCCC CC
                          :  LLLL LLCC
         C CCCC C         :  KKKK KKLL
        RL LLLL LCC       :  ____ __KK QQ
   I  W EK KKKK KLL       :  1122 33__ QQ
  RNNI AWS_ ____ _KKQ Q   :  ABAB ABAB 12
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
    UPTE IEE1 1223 3__Q Q  :  ....  ....  ..
    NLRN TNTA BABA BAB1 2  :  DDDD DDDD DD
    -----------------------------------------
    ---- --10 0000 000- -  :  0000 0000 --;     1/  116
    ---- --00 0000 000- -  :  0000 0010 0-;     2/  143
    ---- --10 0000 0100 -  :  0000 0000 --;     3/  117
    ---- --00 0000 0100 -  :  0000 0001 0-;     4/  145
    ---- --10 0000 0010 -  :  0000 0000 --;     5/  118
    0--- --00 0000 0010 -  :  0000 0010 0-;     6/  147
    10-- --00 0000 0010 -  :  1000 0010 -0;     7/  148
    11-- --00 0000 0010 -  :  1000 0010 -1;     8/  149
    ---- --11 0000 010- 0  :  0000 0000 --;     9/  119
    ---- --01 0000 010- 0  :  0100 0001 -0;    10/  155
    ---- --10 1000 001- 0  :  0000 0000 --;    11/  120
    ---- --00 1000 001- 0  :  1010 0010 --;    12/  157
    ---- --11 0100 010- -  :  0000 0000 --;    13/  121
    ---- --01 0100 010- -  :  0101 0001 --;    14/  159
    ---- --10 1010 001- -  :  0000 0000 --;    15/  122
    --11 --00 1010 001- -  :  1000 0010 -0;    16/  161
    --10 --00 1010 001- -  :  1010 1010 --;    17/  162
    --0- --00 1010 001- -  :  1010 1010 --;    18/  163
    ---- --11 0101 010- -  :  0000 0000 --;    19/  123
    ---- 1101 0101 010- -  :  0000 0001 10;    20/  166
    0--- 0-01 0101 010- -  :  0001 0101 --;    21/  167
    0--- 1001 0101 010- -  :  0001 0101 --;    22/  168
    1--- 0-01 0101 010- -  :  0101 0101 --;    23/  169
    1--- 1001 0101 010- -  :  0101 0101 --;    24/  170
    ---- --10 1010 101- -  :  0000 0000 --;    25/  124
    --11 --00 1010 101- -  :  1000 0010 -0;    26/  172
    --0- --00 1010 101- -  :  1010 1010 --;    27/  173
    --10 --00 1010 101- -  :  1010 1010 --;    28/  174
    ---- --10 0010 101- -  :  0000 0000 --;    29/  125
    ---- --00 0010 101- -  :  0000 1010 -0;    30/  177
    ---- --10 0001 010- 0  :  0000 0000 --;    31/  126
    ---- --00 0001 010- 0  :  0000 0101 -0;    32/  179
    ---- --10 0000 101- 0  :  0000 0000 --;    33/  127
    ---- --00 0000 101- 0  :  0000 0010 0-;    34/  181
    ---- --10 0000 0101 0  :  0000 0000 --;    35/  128
    ---- --00 0000 0101 0  :  0000 0001 10;    36/  187
    ---- --10 0000 0011 0  :  0000 0000 --;    37/  129
    ---- 1-00 0000 0011 0  :  0000 0010 10;    38/  184
    ---- 0-00 0000 0011 0  :  1010 1010 --;    39/  185
    ---- --11 0000 010- 1  :  0000 0000 --;    40/  130
EXPANDED FUNCTION TABLE (LOCAL VARIABLES REMOVED)- continued :
    ---- --01 0000 010- 1  :  0100 0001 -1;    41/  192
    ---- --10 1000 001- 1  :  0000 0000 --;    42/  131
    ---- --00 1000 001- 1  :  0010 0010 --;    43/  194
    ---- --10 0100 010- -  :  0000 0000 --;    44/  132
    ---- 1100 0100 010- -  :  0000 0001 11;    45/  197
    ---- 0-00 0100 010- -  :  0001 0001 --;    46/  198
    ---- 1000 0100 010- -  :  0001 0001 --;    47/  199
    ---- --10 0010 001- -  :  0000 0000 --;    48/  133
    --11 --00 0010 001- -  :  1000 0010 -1;    49/  202
    --0- --00 0010 001- -  :  0000 1010 -1;    50/  203
    --10 --00 0010 001- -  :  0000 1010 -1;    51/  204
    ---- --10 0001 010- 1  :  0000 0000 --;    52/  134
    ---- --00 0001 010- 1  :  0000 0101 -1;    53/  207
    ---- --10 0000 101- 1  :  0000 0000 --;    54/  135
    1--- --00 0000 101- 1  :  1000 0010 -1;    55/  209
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
      0--- --00 0000 101- 1  :  0000 0010 0-;    56/  210
      ---- --10 0000 0101 1  :  0000 0000 --;    57/  136
      ---- --00 0000 0101 1  :  0000 0001 11;    58/  216
      ---- --10 0000 0011 1  :  0000 0000 --;    59/  137
      ---- 1-00 0000 0011 1  :  0000 0010 11;    60/  213
      ---- 0-00 0000 0011 1  :  0010 0010 --;    61/  214
      REST                   :  ---- ---- --;    62
      ---------------------------------------
      1234 5678 9012 3456 7     1234 5678 90
  PIPELINED CLOCKING SYSTEM OD20G10
  CYPRESS SEMICONDUCTOR
  90/03/15  23:49:45
  ****************************************************
  ***  NET DESCRIPTION TABLE FOR AND/OR STRUCTURE  ***
  ****************************************************
                       :  CCCC CC
                       :  LLLL LLCC
        C CCCC C       :  KKKK KKLL
       RL LLLL LCC     :  ____ __KK QQ
     I  W EK KKKK KLL     :  1122 33__ QQ
RNNI AWS_ ____ _KKQ Q  :  ABAB ABAB 12
UPTE IEE1 1223 3__Q Q  :  ....  ....  ..
NLRN TNTA BABA BAB1 2  :  DDDD DDDD DD
--------------------------------------
                     INV  ....  ....  ..
                     REG  DDDD DDDD DD
--------------------------------------
---- 0-0- --0- 0-11 0  :  A...  ....  ..  ;   1
1--- --0- --0- 1--- 1  :  A...  ....  ..  ;   2
---- --0- 1--- ---- 0  :  A...  ....  ..  ;   3
---- --0- 1-1- ---- -  :  A...  ....  ..  ;   4
--11 --0- --1- 0--- -  :  A...  ....  ..  ;   5
1--- --0- 0-0- 0-10 -  :  A...  ....  ..  ;   6
---- --01 ---0 ---- -  :  .A..  ....  ..  ;   7
1--- -001 ---- ---- -  :  .A..  ....  ..  ;   8
1--- 0-01 ---- ---- -  :  .A..  ....  ..  ;   9
---0 --0- 1--- ---- -  :  ..A.  ....  ..  ;  10
--0- --0- 1--- ---- -  :  ..A.  ....  ..  ;  11
---- 0-0- --0- 0-11 -  :  ..A.  ....  ..  ;  12
---- --0- 1-0- ---- -  :  ..A.  ....  ..  ;  13
---- 0-0- -1-- ---- -  :  ...A ....  ..  ;  14
---- -00- -1-- ---- -  :  ...A ....  ..  ;  15
---- --01 -1-0 ---- -  :  ...A ....  ..  ;  16
---0 --0- --1- ---- -  :  ....  A...  ..  ;  17
--0- --0- --1- ---- -  :  ....  A...  ..  ;  18
---- --0- 0-1- 1--- -  :  ....  A...  ..  ;  19
---- 0-0- 0-0- 0-11 0  :  ....  A...  ..  ;  20
---- 0-0- ---1 ---- -  :  ....  .A..  ..  ;  21
---- -00- ---1 ---- -  :  ....  .A..  ..  ;  22
---- --0- -0-1 ---- -  :  ....  .A..  ..  ;  23
---- --0- ---- -0-- -  :  ....  ..A.  ..  ;  24
---- --0- ---- -1-- -  :  ....  ...A ..  ;  25
---- ---- ---- -1-1 -  :  ....  ....  A.  ;  26
---- ---- ---- 0-11 -  :  ....  ....  A.  ;  27
---- ---- -1-- ---- -  :  ....  ....  A.  ;  28
---- ---- --0- 1--- -  :  ....  ....  .A ;  29
---- ---- 0-1- 0--- -  :  ....  ....  .A ;  30
---- ---0 -1-- ---- -  :  ....  ....  .A ;  31
---- ---- -0-- -1-- 1  :  ....  ....  .A ;  32
-1-- ---0 0--0 0--0 -  :  ....  ....  .A ;  33
---- ---- 00-- 0--1 1  :  ....  ....  .A ;  34
--------------------------------------
```

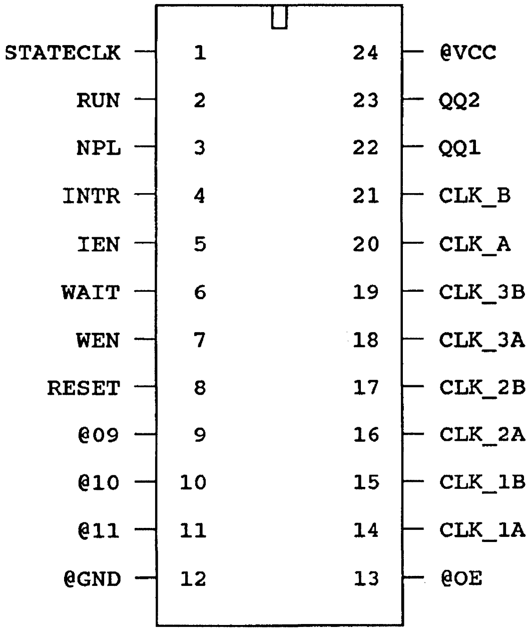
## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
1234 5678 9012 3456 7  :  1234 5678 90
   PIPELINED CLOCKING SYSTEM OD20G10
   CYPRESS SEMICONDUCTOR
   90/03/15  23:49:45
   ****************************************************
   ***              BOOLEAN EQUATIONS               ***
   ****************************************************


  CLK_1A.D     :=
           /WAIT   & /RESET  & /CLK_2B       & /CLK_3B       &  CLK_B
                   &  QQ1    & /QQ2
         +  RUN    & /RESET  & /CLK_2B       &  CLK_3B       &  QQ2
         + /RESET  &  CLK_1B       & /QQ2
         + /RESET  &  CLK_1B       &  CLK_2B
         +  INTR   &  IEN    & /RESET  &  CLK_2B       & /CLK_3B         + RUN     & /RESET  &
/CLK_1B       & /CLK_2B       & /CLK_3B
                   &  CLK_B  & /QQ1    ;
 CLK_1B.D     :=
            /RESET &  CLK_1A       & /CLK_3A
        +  RUN    & /WEN    & /RESET  &  CLK_1A
         +  RUN    & /WAIT   & /RESET  &  CLK_1A       ;
 CLK_2A.D     :=
           /IEN    & /RESET  &  CLK_1B
         + /INTR   & /RESET  &  CLK_1B
         + /WAIT   & /RESET  & /CLK_2B       & /CLK_3B       &  CLK_B
                   &  QQ1
         + /RESET  &  CLK_1B       & /CLK_2B       ;
 CLK_2B.D     :=
           /WAIT   & /RESET  &  CLK_2A
         + /WEN    & /RESET  &  CLK_2A
         + /RESET  &  CLK_1A       &  CLK_2A       & /CLK_3A       ;
 CLK_3A.D     :=
           /IEN    & /RESET  &  CLK_2B
         + /INTR   & /RESET  &  CLK_2B
         + /RESET  & /CLK_1B       &  CLK_2B       &  CLK_3B
         + /WAIT   & /RESET  & /CLK_1B       & /CLK_2B       & /CLK_3B
                   &  CLK_B  &  QQ1    & /QQ2    ;
 CLK_3B.D     :=
           /WAIT   & /RESET  &  CLK_3A
         + /WEN    & /RESET  &  CLK_3A
         + /RESET  & /CLK_2A       &  CLK_3A       ;
 CLK_A.D      :=
           /RESET  & /CLK_A  ; CLK_B.D      :=
           /RESET  &  CLK_A  ;
 QQ1.D  :=  CLK_A  &  QQ1
         + /CLK_3B       &  CLK_B  &  QQ1
         +  CLK_2A       ;
 QQ2.D  := /CLK_2B       &  CLK_3B
         + /CLK_1B       &  CLK_2B       & /CLK_3B
         + /CLK_1A       &  CLK_2A
         + /CLK_2A       &  CLK_A  &  QQ2
         +  NPL    & /CLK_1A       & /CLK_1B       & /CLK_3A
                   & /CLK_3B       & /QQ1
         + /CLK_1B       & /CLK_2A       & /CLK_3B       &  QQ1    &  QQ2    ;
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
 PIPELINED CLOCKING SYSTEM OD20G10
CYPRESS SEMICONDUCTOR
90/03/15  23:49:45
```

PLDC20G10

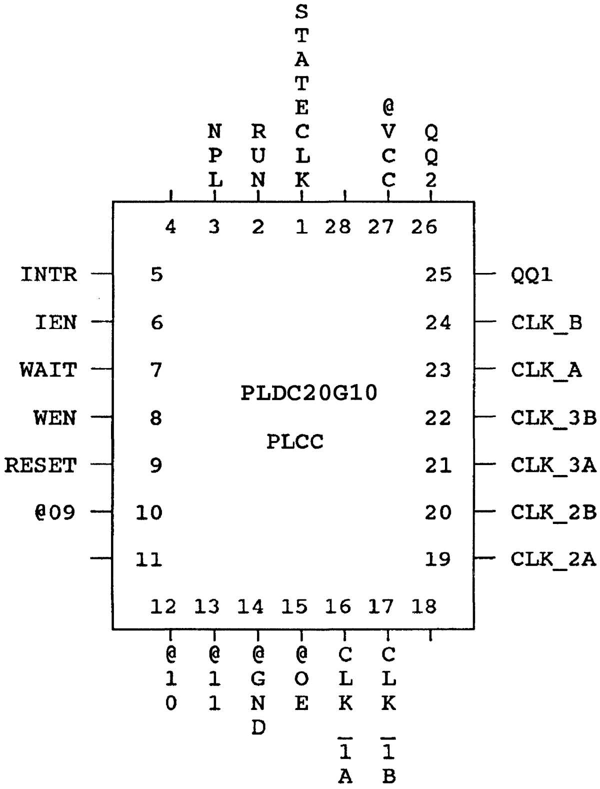


## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

```
PIPELINED CLOCKING SYSTEM OD20G10
CYPRESS SEMICONDUCTOR
90/03/15  23:49:45


                            S
                            T
                            A
                            T
                I           E   @
                N   N   R   C   V   Q   Q
                T   P   U   L   C   Q   Q
                R   L   N   K   C   2   1
             ___|___|___|___|___|___|___|___
            |                               |
            |   4   3   2   1  28  27  26   |
            |                               |
          --|  5                        25  |-- CLK_B
            |                               |
    IEN   --|  6                        24  |-- CLK_A
            |                               |
    WAIT  --|  7                        23  |-- CLK_3B
            |         PLDC20G10             |
          --|  8                        22  |-- CLK_3A
            |            LCC                |
    WEN   --|  9                        21  |-- CLK_2B
            |                               |
    RESET --| 10                        20  |-- CLK_2A
            |                               |
          --| 11                        19  |--
            |                               |
            |  12  13  14  15  16  17  18   |
            |___|___|___|___|___|___|___|___|
                |   |   |   |   |   |   |
                @   @   @   @   @   C   C
                0   1   1   G   O   L   L
                9   0   1   N   E   K   K
                            D
                                    _   _
                                    1   1
                                    A   B
```

## Appendix A. LOG/iC PLD Source Code: Clock State Machine (continued)

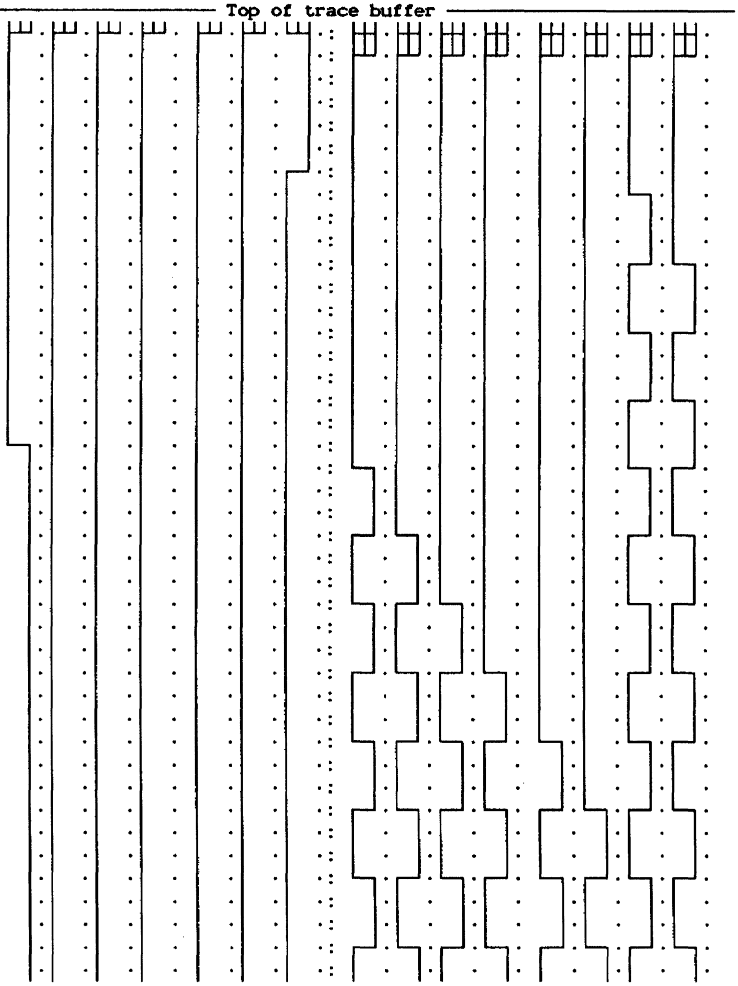
```
PIPELINED CLOCKING SYSTEM OD20G10
CYPRESS SEMICONDUCTOR
90/03/15  23:49:45
```



```
-----------------------------------------------------
LOG/iC  -  PAL            CPU TIME USED:      45 SEC
```

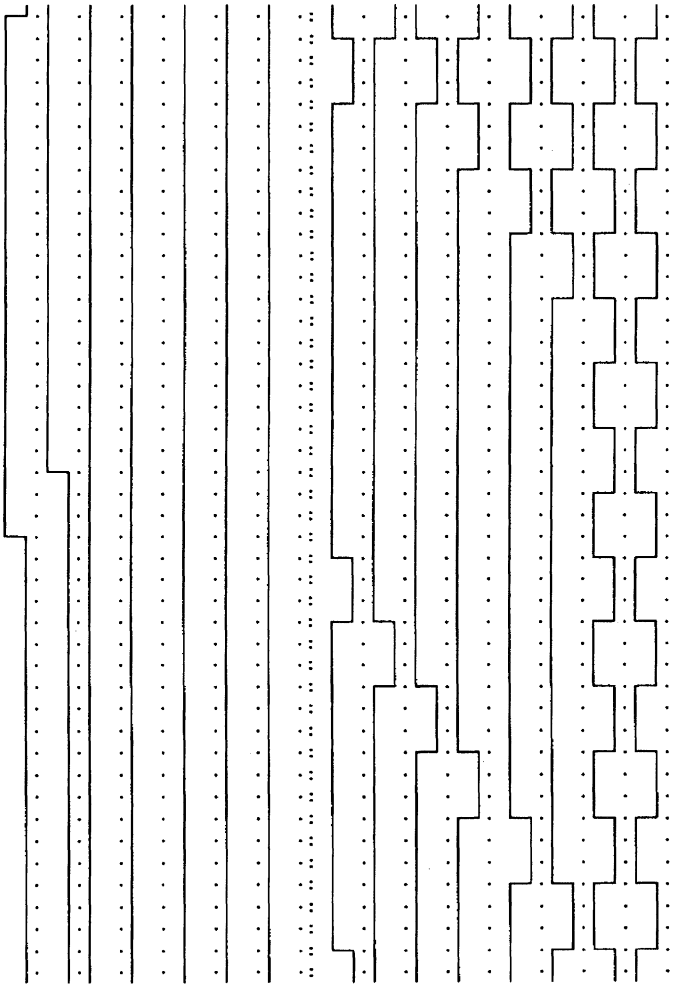
## Appendix B. LOG/iC Simulation: Clock State Machine

```
PIPELINED CLOCKING SYSTEM OD20G10  3/7/90
                                                       C   C   C   C    C   C
  E   S                                          R     L   L   L   L    L   L   C   C
  v   t                I          W              E     K   K   K   K    K   K   L   L
  e   a       R   N    N   I      A   W          S     _   _   _   _    _   _   K   K
  n   t       U   P    T   E      I   E          E     1   1   2   2    3   3   _   _
  t   e       N   L    R   N      T   N          T     A   B   A   B    A   B   A   B
--------------------------------------------------------------------------------------
  #   #      0-1 0-1 0-1 0-1    0-1 0-1 0-1 : 0-1 0-1 0-1 0-1    0-1 0-1 0-1 0-1   0
--------------------------------------------------------------------------------------
----------------------------------- Top of trace buffer ------------------------------
  1   1 IU :
  1   1 IC :
  1   1 IU :
  2   1 IU :
  2   1 IC :
  2   1 IU :
  3   1 IU :
  3   1 IC :
  3   2 IU :
  4   2 IU :
  4   2 IC :
  4   3 IU :
  5   3 IU :
  5   3 IC :
  5   2 IU :
  6   2 IU :
  6   2 IC :
  6   3 IU :
  7   3 IU :
  7   3 IC :
  7   4 IU :
  8   4 IU :
  8   4 IC :
  8   5 IU :
  9   5 IU :
  9   5 IC :
  9   6 IU :
 10   6 IU :
 10   6 IC :
 10   7 IU :
 11   7 IU :
 11   7 IC :
 11   8 IU :
 12   8 IU :
 12   8 IC :
 12   9 IU :
 13   9 IU :
 13   9 IC :
 13   8 IU :
 14   8 IU :
 14   8 IC :
 14   9 IU :
```

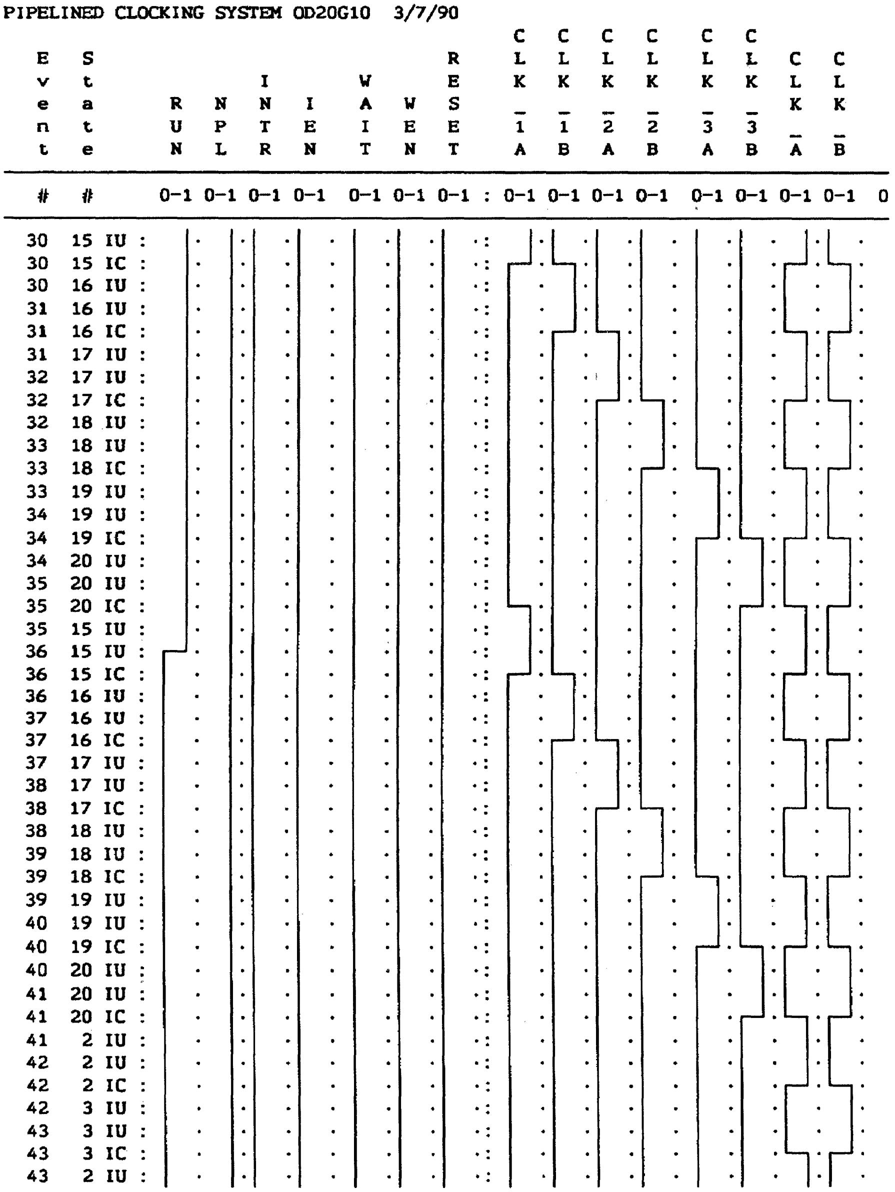


## Appendix B. LOG/iC Simulation: Clock State Machine (continued)

```
PIPELINED CLOCKING SYSTEM OD20G10  3/7/90
                                                   C   C   C   C    C   C
 E   S                                     R       L   L   L   L    L   L   C   C
 v   t             I           W           E       K   K   K   K    K   K   L   L
 e   a       R   N   N   I     A   W       S       _   _   _   _    _   _   K   K
 n   t       U   P   T   E     I   E       E       1   1   2   2    3   3   _   _
 t   e       N   L   R   N     T   N       T       A   B   A   B    A   B   A   B
-------------------------------------------------------------------------------------
 #   #      0-1 0-1 0-1 0-1   0-1 0-1 0-1  :  0-1 0-1 0-1 0-1  0-1 0-1 0-1 0-1  0
-------------------------------------------------------------------------------------
15   9 IU :
15   9 IC :
15   8 IU :
16   8 IU :
16   8 IC :
16  10 IU :
17  10 IU :
17  10 IC :
17  11 IU :
18  11 IU :
18  11 IC :
18  12 IU :
19  12 IU :
19  12 IC :
19   2 IU :
20   2 IU :
20   2 IC :
20   3 IU :
21   3 IU :
21   3 IC :
21   2 IU :
22   2 IU :
22   2 IC :
22   3 IU :
23   3 IU :
23   3 IC :
23  15 IU :
24  15 IU :
24  15 IC :
24  16 IU :
25  16 IU :
25  16 IC :
25  17 IU :
26  17 IU :
26  17 IC :
26  18 IU :
27  18 IU :
27  18 IC :
27  19 IU :
28  19 IU :
28  19 IC :
28  20 IU :
29  20 IU :
29  20 IC :
29  15 IU :
```



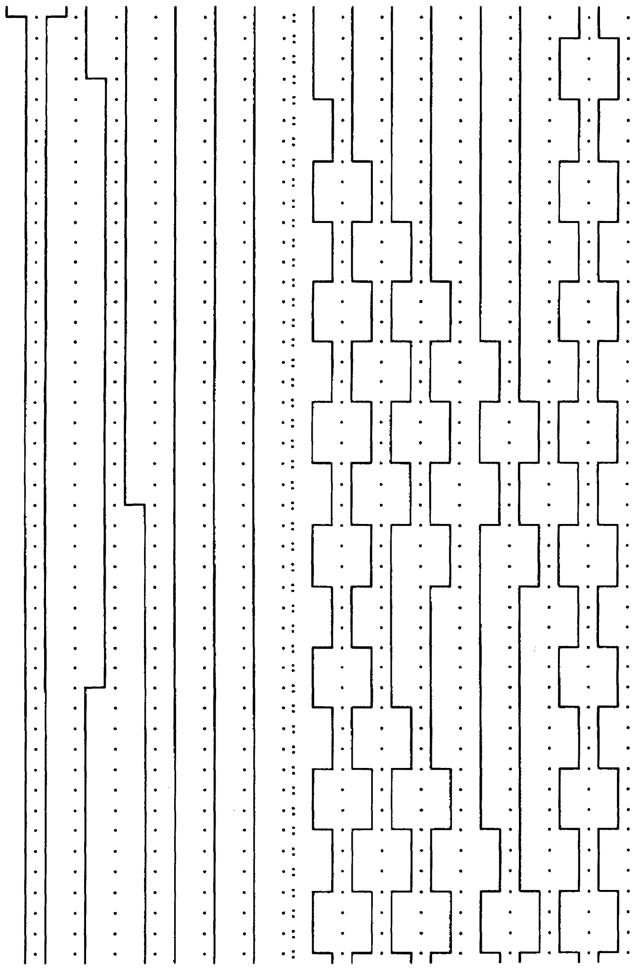
## Appendix B. LOG/iC Simulation: Clock State Machine (continued)



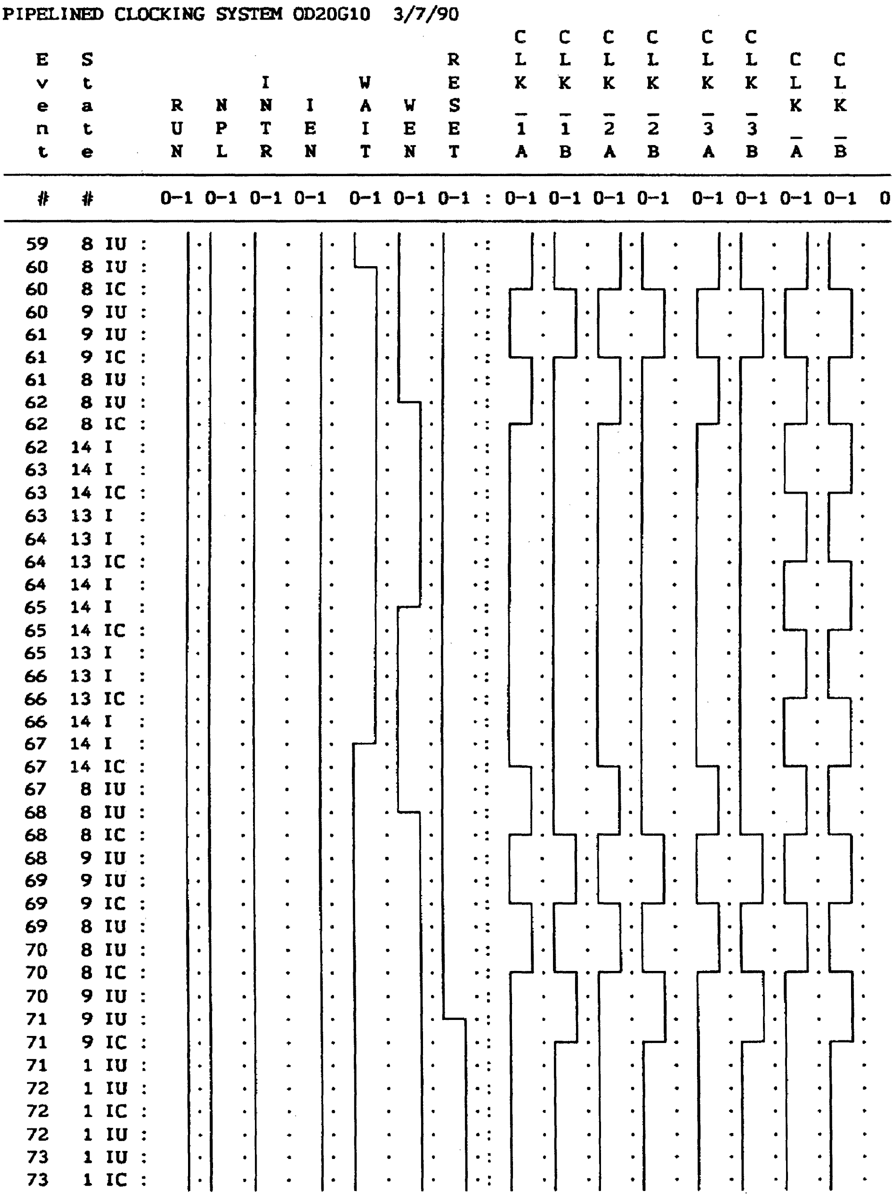
```
PIPELINED CLOCKING SYSTEM OD20G10   3/7/90
                                                    C   C   C   C    C   C
  E   S                                   R         L   L   L   L    L   L   C   C
  v   t               I         W         E         K   K   K   K    K   K   L   L
  e   a       R   N   N   I     A   W     S         _   _   _   _    _   _   K   K
  n   t       U   P   T   E     I   E     E         1   1   2   2    3   3   _   _
  t   e       N   L   R   N     T   N     T         A   B   A   B    A   B   A   B

  #   #      0-1 0-1 0-1 0-1   0-1 0-1 0-1  :  0-1 0-1 0-1 0-1  0-1 0-1 0-1 0-1  0

 30  15 IU :
 30  15 IC :
 30  16 IU :
 31  16 IU :
 31  16 IC :
 31  17 IU :
 32  17 IU :
 32  17 IC :
 32  18 IU :
 33  18 IU :
 33  18 IC :
 33  19 IU :
 34  19 IU :
 34  19 IC :
 34  20 IU :
 35  20 IU :
 35  20 IC :
 35  15 IU :
 36  15 IU :
 36  15 IC :
 36  16 IU :
 37  16 IU :
 37  16 IC :
 37  17 IU :
 38  17 IU :
 38  17 IC :
 38  18 IU :
 39  18 IU :
 39  18 IC :
 39  19 IU :
 40  19 IU :
 40  19 IC :
 40  20 IU :
 41  20 IU :
 41  20 IC :
 41   2 IU :
 42   2 IU :
 42   2 IC :
 42   3 IU :
 43   3 IU :
 43   3 IC :
 43   2 IU :
```

## Appendix B. LOG/iC Simulation: Clock State Machine (continued)

```
PIPELINED CLOCKING SYSTEM OD20G10  3/7/90
                                                  C   C   C   C    C   C
 E   S                                  R         L   L   L   L    L   L   C   C
 v   t               I          W       E         K   K   K   K    K   K   L   L
 e   a      R    N   N    I     A   W   S         _   _   _   _    _   _   K   K
 n   t      U    P   T    E     I   E   E         1   1   2   2    3   3   _   _
 t   e      N    L   R    N     T   N   T         A   B   A   B    A   B   A   B
 ------------------------------------------------------------------------------
 #   #     0-1 0-1 0-1 0-1   0-1 0-1 0-1 :  0-1 0-1 0-1 0-1  0-1 0-1 0-1 0-1   0
 ------------------------------------------------------------------------------
44   2 IU :
44   2 IC :
44   3 IU :
45   3 IU :
45   3 IC :
45   4 IU :
46   4 IU :
46   4 IC :
46   5 IU :
47   5 IU :
47   5 IC :
47   6 IU :
48   6 IU :
48   6 IC :
48   7 IU :
49   7 IU :
49   7 IC :
49   8 IU :
50   8 IU :
50   8 IC :
50   9 IU :
51   9 IU :
51   9 IC :
51   8 IU :
52   8 IU :
52   8 IC :
52   9 IU :
53   9 IU :
53   9 IC :
53   4 IU :
54   4 IU :
54   4 IC :
54   5 IU :
55   5 IU :
55   5 IC :
55   6 IU :
56   6 IU :
56   6 IC :
56   7 IU :
57   7 IU :
57   7 IC :
57   8 IU :
58   8 IU :
58   8 IC :
58   9 IU :
59   9 IU :
59   9 IC :
```

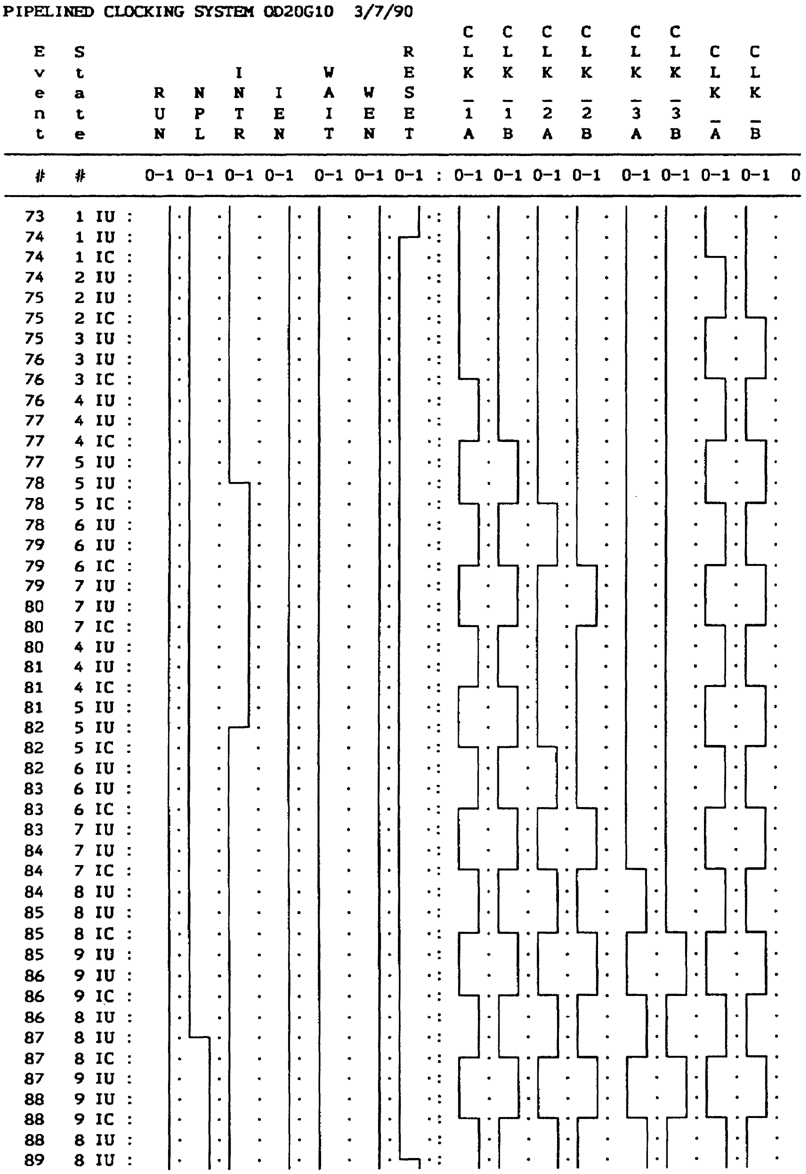


## Appendix B. LOG/iC Simulation: Clock State Machine (continued)



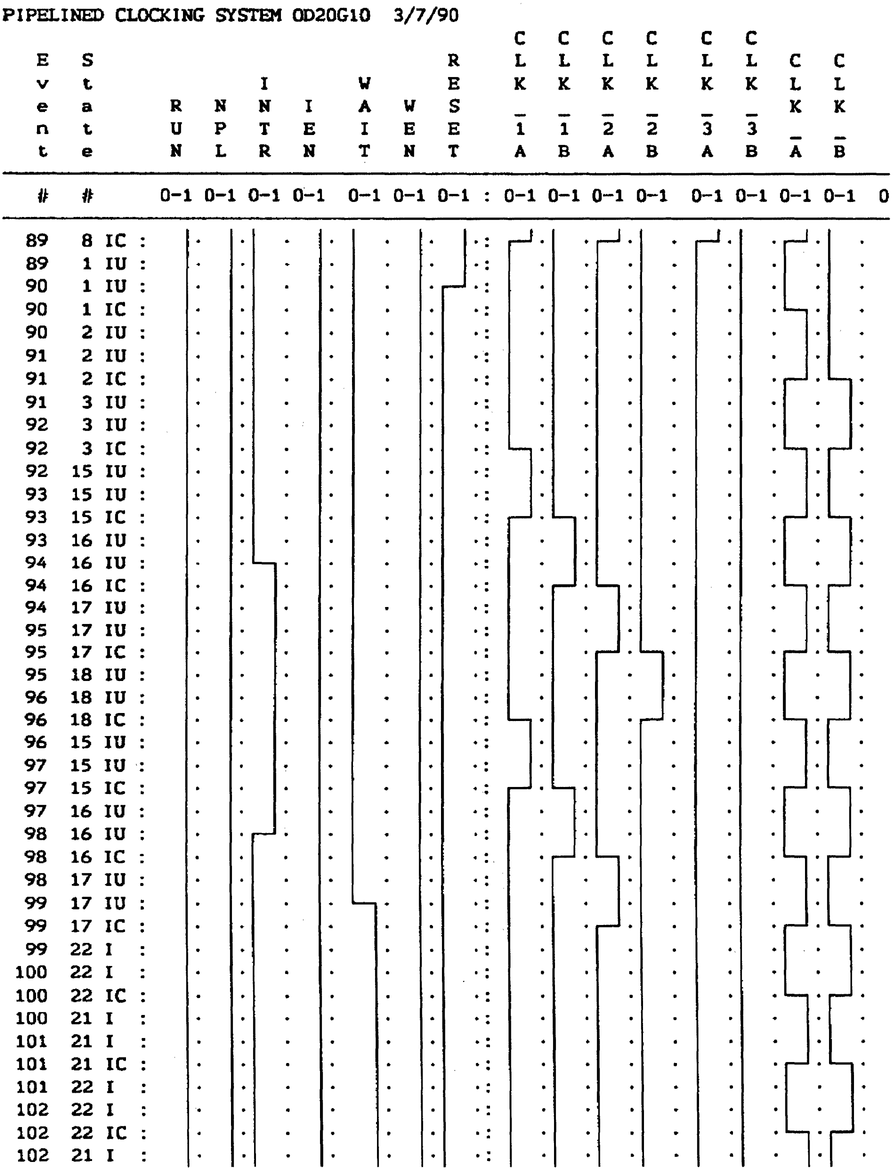
```
PIPELINED CLOCKING SYSTEM OD20G10   3/7/90
                                                   C   C   C   C     C   C
 E   S                                   R         L   L   L   L     L   L   C   C
 v   t              I        W           E         K   K   K   K     K   K   L   L
 e   a      R   N   N   I    A   W       S         _   _   _   _     _   _   K   K
 n   t      U   P   T   E    I   E       E         1   1   2   2     3   3   _   _
 t   e      N   L   R   N    T   N       T         A   B   A   B     A   B   A   B

 #   #     0-1 0-1 0-1 0-1  0-1 0-1 0-1 : 0-1 0-1 0-1 0-1   0-1 0-1 0-1 0-1   0

 59   8 IU :
 60   8 IU :
 60   8 IC :
 60   9 IU :
 61   9 IU :
 61   9 IC :
 61   8 IU :
 62   8 IU :
 62   8 IC :
 62  14 I  :
 63  14 I  :
 63  14 IC :
 63  13 I  :
 64  13 I  :
 64  13 IC :
 64  14 I  :
 65  14 I  :
 65  14 IC :
 65  13 I  :
 66  13 I  :
 66  13 IC :
 66  14 I  :
 67  14 I  :
 67  14 IC :
 67   8 IU :
 68   8 IU :
 68   8 IC :
 68   9 IU :
 69   9 IU :
 69   9 IC :
 69   8 IU :
 70   8 IU :
 70   8 IC :
 70   9 IU :
 71   9 IU :
 71   9 IC :
 71   1 IU :
 72   1 IU :
 72   1 IC :
 72   1 IU :
 73   1 IU :
 73   1 IC :
```

## Appendix B. LOG/iC Simulation: Clock State Machine (continued)



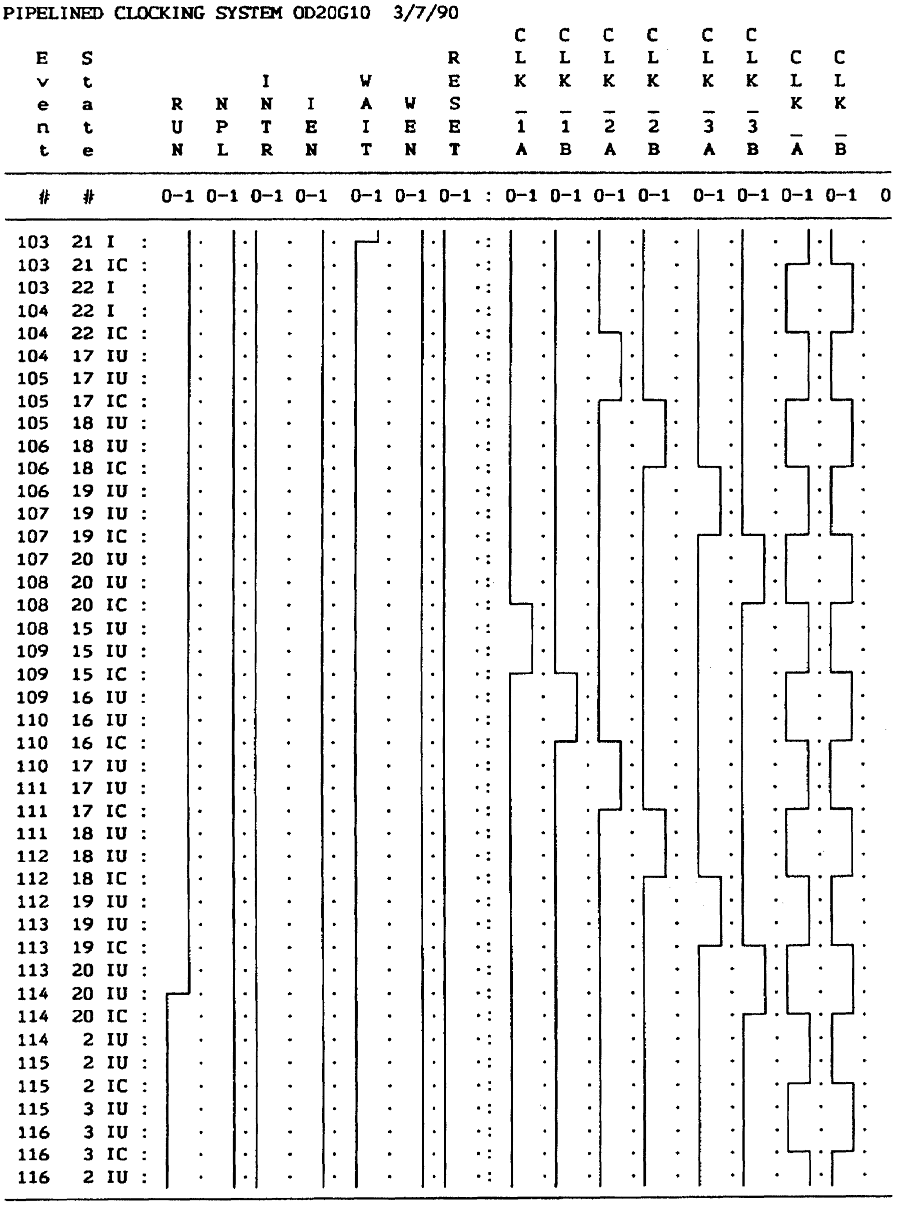
```
PIPELINED CLOCKING SYSTEM CD20G10  3/7/90
                                                  C   C   C   C    C   C
  E  S                                      R     L   L   L   L    L   L   C   C
  v  t             I        W               E     K   K   K   K    K   K   L   L
  e  a       R  N  N  I     A   W           S     _   _   _   _    _   _   K   K
  n  t       U  P  T  E     I   E           E     1   1   2   2    3   3   _   _
  t  e       N  L  R  N     T   N           T     A   B   A   B    A   B   A   B
 ------------------------------------------------------------------------------
  #  #      0-1 0-1 0-1 0-1  0-1 0-1 0-1 : 0-1 0-1 0-1 0-1  0-1 0-1 0-1 0-1   0
 ------------------------------------------------------------------------------
 73  1 IU :
 74  1 IU :
 74  1 IC :
 74  2 IU :
 75  2 IU :
 75  2 IC :
 75  3 IU :
 76  3 IU :
 76  3 IC :
 76  4 IU :
 77  4 IU :
 77  4 IC :
 77  5 IU :
 78  5 IU :
 78  5 IC :
 78  6 IU :
 79  6 IU :
 79  6 IC :
 79  7 IU :
 80  7 IU :
 80  7 IC :
 80  4 IU :
 81  4 IU :
 81  4 IC :
 81  5 IU :
 82  5 IU :
 82  5 IC :
 82  6 IU :
 83  6 IU :
 83  6 IC :
 83  7 IU :
 84  7 IU :
 84  7 IC :
 84  8 IU :
 85  8 IU :
 85  8 IC :
 85  9 IU :
 86  9 IU :
 86  9 IC :
 86  8 IU :
 87  8 IU :
 87  8 IC :
 87  9 IU :
 88  9 IU :
 88  9 IC :
 88  8 IU :
 89  8 IU :
```

## Appendix B. LOG/iC Simulation: Clock State Machine (continued)



```
PIPELINED CLOCKING SYSTEM OD20G10  3/7/90
                                                      C   C   C   C     C   C
  E   S                                     R         L   L   L   L     L   L   C   C
  v   t               I         W           E         K   K   K   K     K   K   L   L
  e   a       R   N   N   I     A   W       S         _   _   _   _     _   _   K   K
  n   t       U   P   T   E     I   E       E         1   1   2   2     3   3   _   _
  t   e       N   L   R   N     T   N       T         A   B   A   B     A   B   A   B

  #   #      0-1 0-1 0-1 0-1   0-1 0-1 0-1   :  0-1 0-1 0-1 0-1   0-1 0-1 0-1 0-1  0

 89   8 IC :
 89   1 IU :
 90   1 IU :
 90   1 IC :
 90   2 IU :
 91   2 IU :
 91   2 IC :
 91   3 IU :
 92   3 IU :
 92   3 IC :
 92  15 IU :
 93  15 IU :
 93  15 IC :
 93  16 IU :
 94  16 IU :
 94  16 IC :
 94  17 IU :
 95  17 IU :
 95  17 IC :
 95  18 IU :
 96  18 IU :
 96  18 IC :
 96  15 IU :
 97  15 IU :
 97  15 IC :
 97  16 IU :
 98  16 IU :
 98  16 IC :
 98  17 IU :
 99  17 IU :
 99  17 IC :
 99  22 I  :
100  22 I  :
100  22 IC :
100  21 I  :
101  21 I  :
101  21 IC :
101  22 I  :
102  22 I  :
102  22 IC :
102  21 I  :
```

## Appendix B. LOG/iC Simulation: Clock State Machine (continued)



```
PIPELINED CLOCKING SYSTEM OD20G10   3/7/90
                                                      C   C   C   C    C   C
 E   S                                     R          L   L   L   L    L   L   C   C
 v   t              I          W           E          K   K   K   K    K   K   L   L
 e   a      R   N   N   I      A   W       S          _   _   _   _    _   _   K   K
 n   t      U   P   T   E      I   E       E          1   1   2   2    3   3   _   _
 t   e      N   L   R   N      T   N       T          A   B   A   B    A   B   A   B

 #   #     0-1 0-1 0-1 0-1    0-1 0-1 0-1 :  0-1 0-1 0-1 0-1  0-1 0-1 0-1 0-1   0

103  21 I  :
103  21 IC :
103  22 I  :
104  22 I  :
104  22 IC :
104  17 IU :
105  17 IU :
105  17 IC :
105  18 IU :
106  18 IU :
106  18 IC :
106  19 IU :
107  19 IU :
107  19 IC :
107  20 IU :
108  20 IU :
108  20 IC :
108  15 IU :
109  15 IU :
109  15 IC :
109  16 IU :
110  16 IU :
110  16 IC :
110  17 IU :
111  17 IU :
111  17 IC :
111  18 IU :
112  18 IU :
112  18 IC :
112  19 IU :
113  19 IU :
113  19 IC :
113  20 IU :
114  20 IU :
114  20 IC :
114   2 IU :
115   2 IU :
115   2 IC :
115   3 IU :
116   3 IU :
116   3 IC :
116   2 IU :
```

LOG/iC is a trademark of Isdata Corporation.

# Efficient Arithmetic Designs Targeting FLASH370i™ CPLDs

## Introduction

The design of fast and efficient arithmetic elements is imperative because of its applications in the many areas of science and engineering. It is important for designers to be aware of the choices available to them in selecting an efficient algorithm for their application. Even the seemingly simple arithmetic operations turn out to be more complex than one expects, when attempting to implement them. There is a lot of literature available in the field, but very little provides the level of detail required to go all the way from a concept to a final implementation.

This application note is intended to help designers create efficient arithmetic designs targeting a FLASH370i™ complex programmable logic device (CPLD). The designer has many alternatives in choosing between arithmetic implementations for a given design. The decision on the final choice is typically based on issues like resource availability, speed of operation, and modularity. Creating designs in view of the target device's architecture will definitely yield better results than implementing a generic design on the same device. The discussion in this application note addresses arithmetic algorithms, design methodologies, and implementations tailored to the features and resources offered in the FLASH370i family of CPLDs. These specialized arithmetic designs achieve a balanced tradeoff between speed/area requirements for a given application. In this application note the user is offered a wide variety of algorithms and implementations to choose from. This variety provides the designer with the flexibility to choose the model best suited for the target application. This choice is absolutely necessary, since design requirements and constraints vary from application to application.

The discussion assumes that the designer has a good feel for the features and resources available in the FLASH370i family of CPLDs. The implementation details and design tradeoffs in building adders, subtracters, equality and magnitude comparators are addressed in this application note. This application note includes many VHDL (VHSIC Hardware Design Language) examples to illustrate the working and implementation of the algorithms presented. Block diagrams are also presented wherever necessary to help the designer understand the design better.

All algorithms in this application note are described within the same framework, so that the similarities between different algorithms become evident and consequently, the basic principle behind these algorithms can be easily identified. This application note is also intended to create a solid foundation from which designers can pick up ideas and concepts and create their own algorithms/implementations.

The VHDL code presented in this application note are intentionally presented in a simple style. The intent of this application note is to allow a designer to visualize and implement arithmetic models efficiently and not to explain how to code them. All VHDL keywords are presented in italics. This application note also assumes that the reader has a good grasp of the fundamentals of VHDL. Some of the LPM (library of parameterized modules) elements for CPLDs provided in the *Warp*™ software are built using the concepts and final implementations discussed here. This provides the user with an excellent opportunity to choose the best algorithm and implementation tailored to the target application.

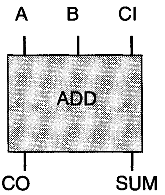
## Adders

The addition of two operands is the most frequent operation in almost any arithmetic unit. The two-operand adder is commonly used in performing additions and subtractions. It is also used when executing complex arithmetic functions like multiplication and division.

### ADD: 1-Bit Full Adder

The basic component used in adding two operands is called a *Full Adder.* The full adder element will be henceforth referred to as the 'ADD' component. The block diagram and functionality of ADD is shown in *Figure 1.* A and B are the two

ADD:     1-Bit Full Adder (1 Pass)



(Basic building block)

Functionality: SUM = A *XOR* B *XOR* CI

CO = (A *AND* B) or (A *AND* CI) or (B *AND* CI)

**Figure 1. Block Diagram and Functionality of a Full Adder**

operands to be added and CI is the Carry-in to the component. SUM and CO are the Sum and Carry-out from the component.

The VHDL code describing the functionality of the ADD component is shown here. This design takes one pass through the Logic (AND-OR) array to fit into a FLASH370i CPLD. The ADD component instantiated in the VHDL code shown has exactly the same functionality shown in *Figure 1.*

```
-- This VHDL code implements a full adder component called ADD
-- within a package called MATHPKG
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
PACKAGE mathpkg IS
  COMPONENT add
    PORT (CI: IN STD_LOGIC;
      A, B: INSTD_LOGIC;
      SUM: OUT STD_LOGIC;
      CO: OUT STD_LOGIC);
  END COMPONENT;
END mathpkg;
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY add IS
  PORT (CI: IN STD_LOGIC;
    A, B: IN STD_LOGIC;
    SUM: OUT STD_LOGIC;
    CO: OUT STD_LOGIC);
END add;
ARCHITECTURE archadd OF add IS
BEGIN
  SUM <= A XOR B XOR CI;
  CO <= (A and B) or (A and CI) or (B and CI);
END archadd;
```

### RADD12: 12-Bit Ripple Carry Adder

An *n*-bit two-operand ripple carry adder can be built using *n* ADD components. All the 2*n* input bits are available to the adder at the same time. However the carries have to propagate from the LSB position to the MSB. In other words, we need to wait until the carries ripple through *n* ADD components to claim that the SUM outputs are correct. Because of this rippling effect, the adder is referred to as the *Ripple Carry* Adder. This is the simplest form of adding any two operands. It uses the least amount of area compared to all other implementations but, on the negative side, is the slowest implementation. This is typically the implementation provided with a synthesis tool when it recognizes the '+' operator in a VHDL code. The block diagram of a 12-bit Ripple Carry Adder (RADD12) is shown in *Figure 2.*

The VHDL code describing the functionality of the RADD12 component is shown here. This design takes 12 passes through the logic array to fit into a FLASH370i CPLD. The outputs of the LSB ADD component are produced in the first pass. The outputs of the succeeding ADD components are produced with every alternate pass through the logic array. Each pass through the logic array has a time penalty associated with it. It is recommended that the reader understand the timing issues associated with the FLASH370i CPLD (refer to the "CY7C37x Timing Parameters" application note).

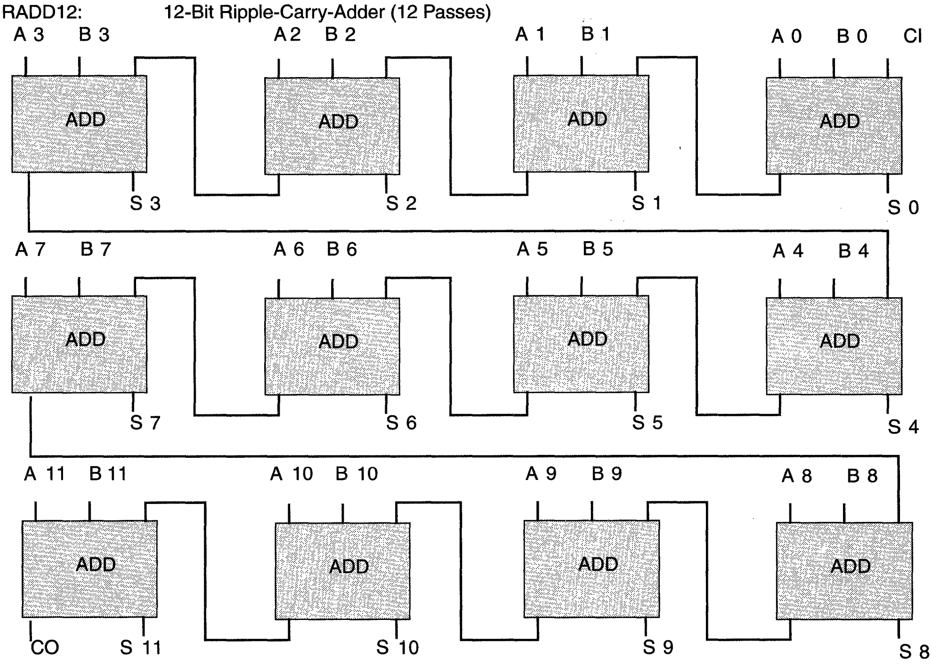


**Figure 2. Block Diagram of a 12-Bit Ripple Carry Adder**

```
--This VHDL code describes the implementation of a generic
--12 bit ripple carry adder.

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE WORK.MATHPKG.ALL;

ENTITY rippleadd12 IS
  PORT  (CI: IN STD_LOGIC;
    A11, A10, A9, A8, A7, A6, A5, A4, A3, A2, A1, A0 : IN STD_LOGIC;
    B11, B10, B9, B8, B7, B6, B5, B4, B3, B2, B1, B0 : IN STD_LOGIC;
    SUM11, SUM10, SUM9, SUM8, SUM7, SUM6, SUM5, SUM4,
    SUM3, SUM2, SUM1, SUM0 : OUT STD_LOGIC;
    CO: OUT STD_LOGIC);
END rippleadd12;

ARCHITECTURE archripple12add OF rippleadd12 IS
SIGNAL C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11 : STD_LOGIC;

attribute synthesis_off of C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11 : signal is true;

BEGIN

  i1:  add PORT MAP(CI,A0,B0,SUM0,C1);
  i2:  add PORT MAP(C1,A1,B1,SUM1,C2);
  i3:  add PORT MAP(C2,A2,B2,SUM2,C3);
  i4:  add PORT MAP(C3,A3,B3,SUM3,C4);
  i5:  add PORT MAP(C4,A4,B4,SUM4,C5);
  i6:  add PORT MAP(C5,A5,B5,SUM5,C6);
  i7:  add PORT MAP(C6,A6,B6,SUM6,C7);
  i8:  add PORT MAP(C7,A7,B7,SUM7,C8);
  i9:  add PORT MAP(C8,A8,B8,SUM8,C9);
  i10: add PORT MAP(C9,A9,B9,SUM9,C10);
  i11: add PORT MAP(C10,A10,B10,SUM10,C11);
  i12: add PORT MAP(C11,A11,B11,SUM11,CO);

END archripple12add;
```

The need and use for the 'Synthesis_off' attribute used in the VHDL code will be discussed a little later.

### ADD2WC: 2-Bit Adder with Carry-Out

The concept of the ADD component can be extended to create a 2-bit adder which takes in two 2-bit operands with a carry-in and produces a 2-bit SUM and a carry-out as outputs. This component will be referred to as the ADD2WC (2-bit adder with a carry-out). This also takes just one pass through the logic array to yield results. The block diagram of ADD2WC is shown in *Figure 3*. A0, A1 and B0, B1 are the two operands to be added and CI is the Carry-in to the component. S0, S1 and CO are the Sums and Carry-outs from the component.

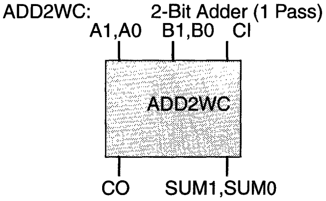


**Figure 3. A 2-Bit Full Adder with a Carry-Out**

The VHDL code describing the functionality of the ADD2WC component is shown here. This design takes one pass through the logic array to fit into a FLASH370i CPLD.

```
--VHDL code describing a 2-bit adder with carry-out.

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

PACKAGE add2wc_pkg IS
  COMPONENT add2wc PORT(
    CI : IN STD_LOGIC;
    A1,A0: IN STD_LOGIC;
    B1,B0: IN STD_LOGIC;
    SUM1,SUM0 : OUT STD_LOGIC;
    CO: OUT STD_LOGIC);
  END COMPONENT;
END add2wc_pkg;

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY add2wc IS
  PORT (CI : IN STD_LOGIC;
        A1,A0: IN STD_LOGIC;
        B1,B0: IN STD_LOGIC;
        SUM1,SUM0 : OUT STD_LOGIC;
        CO: OUT STD_LOGIC);
END add2wc;

ARCHITECTURE archadd2wc OF add2wc IS

BEGIN

SUM0 <= A0 XOR B0 XOR CI;
SUM1 <= A1 XOR B1 XOR ((A0 AND B0) OR (A0 AND CI) OR (B0 AND CI));
CO   <= (A0 AND B0 AND B1)
  OR (A0 AND B0 AND A1)
  OR (CI AND B0 AND B1)
  OR (CI AND B0 AND A1)
  OR (CI AND A0 AND B1)
  OR (CI AND A0 AND A1)
  OR (A1 AND B1);

END archadd2wc;
```

The concept of ADD2WC can be extended to describe the ADD2NC component. The ADD2NC component is a cut-down version of the ADD2WC component, and does *not* have a carry-out. The VHDL code and block diagram for the ADD2NC component is easy to extrapolate and is not shown here.

### R2ADD12: 12-Bit Ripple Carry Adder using the ADD2WC as a Basic Block

A 12-bit adder using the ADD2WC component is shown here. This adder takes 6 passes to produce all results, as opposed to the 12 passes needed for the 12-bit adder using the ADD component. The outputs of the LSB ADD2WC component are produced in the first pass. The outputs of the succeeding ADD2WC components are produced with every alternate pass through the logic array. The number of macrocells used by this scheme is less than RADD12, but the product term count is higher. A comparison of different schemes is presented later. The block diagram of R2ADD12 is shown in *Figure 4.* The VHDL code describing the functionality is also attached.

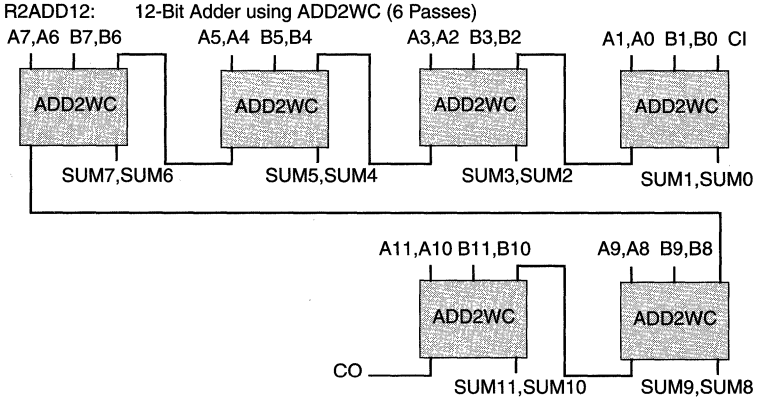


**Figure 4. Block Diagram of a 12-Bit Ripple Carry Adder Using 2-Bit Adders**

```
--A 12-bit Ripple carry adder built using the ADD2WC element as a basic building block

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE WORK.ADD2WC.ALL;

ENTITY add12 IS
  PORT (CI : IN STD_LOGIC;
       A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0: IN STD_LOGIC;
       B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0: IN STD_LOGIC;
       SUM11,SUM10,SUM9,SUM8,SUM7,SUM6,SUM5,SUM4,
       SUM3,SUM2,SUM1,SUM0 : OUT STD_LOGIC;
       CO: OUT STD_LOGIC);
END add12;

ARCHITECTURE archadd12 OF add12 IS

  SIGNAL C2, C4, C6, C8, C10 : STD_LOGIC;

  attribute synthesis_off of C2, C4, C6, C8, C10 : signal is true;

BEGIN

  i1:  add2wc PORT MAP(CI,A1,A0,B1,B0,SUM1,SUM0,C2);
  i2:  add2wc PORT MAP(C2,A3,A2,B3,B2,SUM3,SUM2,C4);
  i3:  add2wc PORT MAP(C4,A5,A4,B5,B4,SUM5,SUM4,C6);
  i4:  add2wc PORT MAP(C6,A7,A6,B7,B6,SUM7,SUM6,C8);
  i5:  add2wc PORT MAP(C8,A9,A8,B9,B8,SUM9,SUM8,C10);
  i6:  add2wc PORT MAP(C10,A11,A10,B11,B10,SUM11,SUM10,CO);

END archadd12;
```

### ADD3WC: The 3-Bit Ripple Carry Adder

There is yet another way we could implement an *n*-bit ripple carry adder targeting the FLASH370i CPLDs. We can implement the *n*-bit adder using the 3-bit group adder (ADD3WC) as opposed to a 2-bit group adder (ADD2WC). The problem with a 3-bit group adder is the sum-splitting of the functionality of the MSB Sum bit (SUM2). This takes more than 16 product terms (PTs) and takes 2 passes through the logic array to produce the result. All other results, including the carry-out, take less than 16 PTs and take just one pass to produce results. To control sum-splitting the functionality of SUM2, the intermediate carry C2 is created and assigned to a node. C2 is then used to create the functionality of SUM2. Note that the functionality of CO takes less than 16 PTs and is generated at the first pass, so the carry rippling is faster. This makes this component a faster building block. This scheme still takes two passes to create the functionality of SUM2, but without getting sum-split. The resource utilization of a 12-bit adder using the 3-bit group adder is presented later. The block diagram of the ADD3WC component is shown in *Figure 5*.

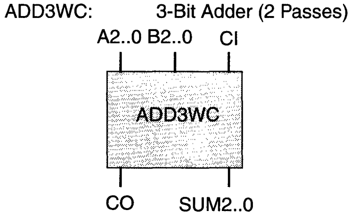


**Figure 5. A 3-Bit Full Adder with a Carry-Out**

```
-- 3-Bit Adder with Carry-out
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

PACKAGE add3wc_pkg IS
  COMPONENT add3wC
    PORT (CI : IN BIT;
        A2,A1,A0: IN BIT;
        B2,B1,B0: IN BIT;
        SUM2,SUM1,SUM0 : OUT BIT;
        CO: OUT BIT);
  END COMPONENT;
END add3wc_pkg;
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY add3wc IS
  PORT (CI : IN STD_LOGIC;
    A2,A1,A0: IN STD_LOGIC;
    B2,B1,B0: IN STD_LOGIC;
    SUM2,SUM1,SUM0 : OUT STD_LOGIC;
    CO: OUT STD_LOGIC);
END add3wc;
ARCHITECTURE archadd3wc OF add3wc IS

SIGNAL C2: STD_LOGIC;

  attribute synthesis_off of C2: signal is true;
BEGIN
  SUM0 <= A0 XOR B0 XOR CI;
  SUM1 <= A1 XOR B1 XOR ((A0 AND B0) or (A0 AND CI) or (B0 AND CI));
  SUM2 <= A2 XOR B2 XOR C2;
  C2   <= (A0 AND B0 AND B1)
    OR (A0 AND B0 AND A1)
    OR (CI AND B0 AND B1)
    OR (CI AND B0 AND A1)
    OR (CI AND A0 AND B1)
    OR (CI AND A0 AND A1)
    OR (A1 AND B1);
  CO <= (A2 AND B2) OR ((A1 AND B1) AND (A2 OR B2))
    OR ((A0 AND B0) AND (A1 OR B1) AND (A2 OR B2))
    OR (CI AND (A0 OR B0) AND (A1 OR B1) AND (A2 OR B2));
END archadd3wc;
```

### Function and Use of the *Synthesis_off* Attribute

The Synthesis_off attribute causes a signal to be made into a factoring point for logic equations and keeps the signal from being minimized out during optimization.

The attribute is useful for the following reasons:

1. It gives the user control over which equations or sub-expressions need to be factored into a node.
2. It helps in cutting down on compile time for designs that have a lot of 'signal redirection' (signals getting inverted/reassigned to other signals). This attribute provides the Logic optimizer a better control over the optimization process by reducing the number of signals it needs to deal with.
3. It provides better results for designs where a signal with a large functionality is being used by many other signals. If left alone, the fitter would collapse all the internal signals (which is desirable in many cases) and may drive the design's resource requirements beyond the available limits.

By using the Synthesis_off attribute, the user can assign the commonly-used signal to a node and bring down the resource utilization.

A side effect of using the Synthesis_off attribute is that the design will now take an extra pass through the array to achieve the same functionality. The extra pass may be required anyway, if more than 16 PTs are required.

This attribute is only recommended for use on combinatorial signals. Registered signals are assigned to a node by natural factoring and the Synthesis_off attribute on these signals is redundant.

This attribute can be associated with signals declared both in VHDL and schematics. The 'BUF' component can also be used in schematics and VHDL to achieve the same results as the Synthesis_off attribute. Please refer to the *Warp* Synthesis manual for more details.

### Carry-Lookahead Principle

The predominant delay in adders is due to carry propagation. The carry-lookahead principle aims at minimizing this delay. The sum and carry equations for each bit position in an adder is given by:

$$\mathbf{S_i = A_i \text{ xor } B_i \text{ xor } C_i}$$
$$\mathbf{C_{i+1} = (A_i \text{ and } B_i) \text{ or } (A_i \text{ and } C_i) \text{ or } (B_i \text{ and } C_i)}$$

A carry is *generated* whenever $A_i$ and $B_i$ are both '1' and a carry is *propagated* whenever either $A_i$ or $B_i$ are '1'.

Generate term: **($G_i = A_i$ and $B_i$)**
Propagate term: **($P_i = A_i$ or $B_i$)**

Note: $P_i$ can be ($A_i$ xor $B_i$), but 'OR' is easier to implement than an 'XOR' in CPLDs.

Rewriting the equation for $C_{i+1}$, we get

$$C_{i+1} = G_i \text{ or } (P_i \text{ and } C_i)$$

Writing the equations for a 4-bit carry-lookahead adder:

$$\mathbf{C_1 = G_0 \text{ or } (P_0 \text{ and } C_0)}$$
$$\mathbf{C_2 = G_1 \text{ or } (P_1 \text{ and } C_1)}$$
$$\mathbf{C_3 = G_2 \text{ or } (P_2 \text{ and } C_2)}$$
$$\mathbf{C_4 = G_3 \text{ or } (P_3 \text{ and } C_3)}$$

where **$G_i = (A_i$ and $B_i)$** and **$P_i = (A_i$ or $B_i)$**. The values of $G_i$ and $P_i$ can be generated in a single pass through the PIM array. The carry-in to any of the bit positions can be computed in a second pass through the array, based upon the values of the various **$G_i$s and $P_i$s** generated in the first pass.

The generalized carry-lookahead equation to compute the different carry-in signals is shown here:

$C_{i+1}$ = $G_i$ or ($P_i$ and $G_{i-1}$) or ($P_i$ and $P_{i-1}$ and $G_{i-1}$) or ... or ($P_i$ and $P_{i-1}$ and ... and $P_0$ and $C_0$)

We can further speed up the addition by providing a carry-lookahead over groups in addition to the internal lookahead within the group. We define a *group-generated* carry *E* and a *group-propagated* carry *R*, for a group of size 4 as follows: *E* = '1' if a carry-out (of the group) is generated internally and *R* = '1' if a carry-in (to the group) is propagated internally to produce a carry-out (of the group). The boolean equations for these carries are:

$$\mathbf{E = G_3 \text{ or } (P_3 \text{ and } G_2) \text{ or } (P_3 \text{ and } P_2 \text{ and } G_1) \text{ or } (P_3 \text{ and } P_2 \text{ and } P_1 \text{ and } G_0)}$$
$$\mathbf{R = (P_3 \text{ and } P_2 \text{ and } P_1 \text{ and } P_0)}$$

The group-generated and group-propagated carries for several groups can now be used to generate group carry-ins in a manner similar to single-bit carry-ins.

The selection of the group size plays an important role in obtaining the best possible implementation for a carry-lookahead adder in a CPLD. Some of the different possible implementations for a 12-bit carry-lookahead adder are shown in *Figure 6*.

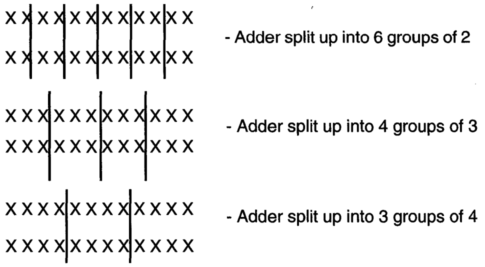

- Adder split up into 6 groups of 2
- Adder split up into 4 groups of 3
- Adder split up into 3 groups of 4

**Figure 6. Some Possible Implementations for 12-Bit Carry-Lookahead Adder**

The number of passes each of these implementations take and the number of product terms (PTs) and macrocells (MCs) used vary for each scheme (see *Table 1* in the "Comparison of Resource Utilization for Different Schemes in Building a 12-Bit Adder" section). Each scheme has its own advantage over the other. The user needs to judiciously choose between the different schemes based on the application, bit-size, and the CPLD chosen and its architectural constraints. The number of passes taken through the logic is a direct representation of the total time taken for producing final results. Each extra pass results in a time penalty. The rule to follow is, "The smaller the number of passes through the logic array, the faster your application runs." The implementation of a 12-bit carry-lookahead adder with different group-sizes is presented next.

### FC2ADD12: 12-Bit Full Carry-Lookahead Adder Using a Group-Size of 2 Bits

The FLASH370i CPLD can access up to 16 PTs for each macrocell. The functionality of any signal that has more than 16

PTs is sum-split to fit it into multiple MCs. The number of PTs utilized for signals that sum-split is large and is an undesirable option. With the 2-bit group-size implementation we can accommodate the entire functionality of a 32-bit full carry-lookahead adder without any of the signals getting sum-split. The scheme takes a maximum of three passes through the logic array for all adder sizes up to 32 bits to generate outputs. The various values of Es and Rs, SUM1, SUM0, and C2 are generated in the first pass. All the other intermediate carries are generated in the second pass and the various SUM results are generated in the third pass. A key point to note is that the value of CO is produced in the second pass, even though the various SUM outputs are generated in the third pass only. This makes the component cascadable and modular. Refer to *Table 1* for details on the resource utilization of different 12-bit adder implementations. The FC2ADD12 is built using the ADD2WC and ADD2NC as basic building blocks. The block diagram of a FC2ADD12 is shown in *Figure 7.* The VHDL code for the design is also presented.

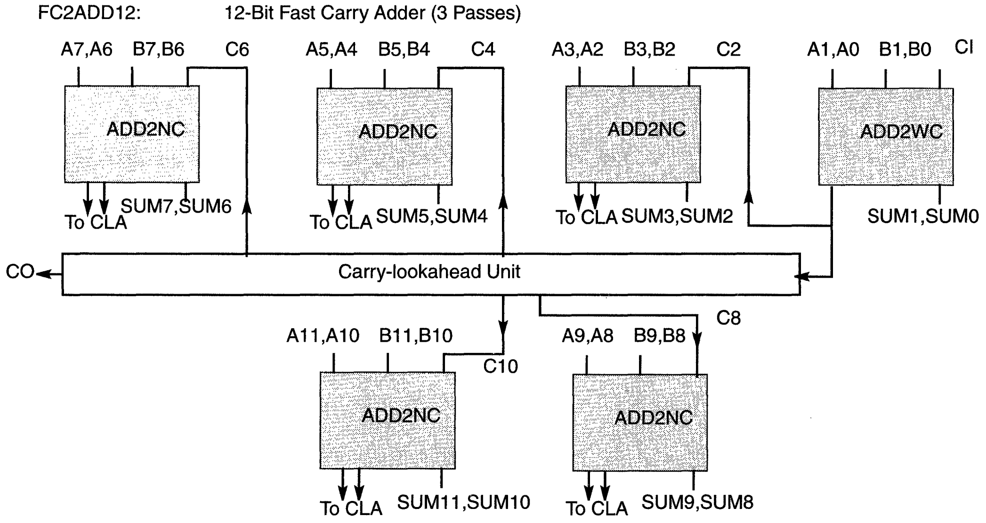


**Figure 7. 12-Bit Full Carry-Lookahead Adder Using ADD2WC and ADD2NC**

```
--A 12-bit Full carry-lookahead adder built using the ADD2WC and ADD2NC
--elements

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE WORK.ADD2WC.ALL;
USE WORK.ADD2NC.ALL;

ENTITY fc2add12 IS
  PORT (CI : IN STD_LOGIC;
       A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0: IN STD_LOGIC;
       B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0: IN STD_LOGIC;
       SUM11,SUM10,SUM9,SUM8,SUM7,SUM6,SUM5,SUM4,
       SUM3,SUM2,SUM1,SUM0 : OUT STD_LOGIC;
       CO: OUT STD_LOGIC);
END fc2add12;

ARCHITECTURE archfc2add12 OF fc2add12 IS

  SIGNAL C2, C4, C6, C8, C10 : STD_LOGIC;
  SIGNAL E1,E2,E3,E4,E5  : STD_LOGIC;
  SIGNAL R1,R2,R3,R4,R5  : STD_LOGIC;
  attribute synthesis_off of E1,E2,E3,E4,E5 : signal is true;
  attribute synthesis_off of R1,R2,R3,R4,R5 : signal is true;
  attribute synthesis_off of C2, C4, C6, C8, C10 : signal is true;
```

```
BEGIN

  i1:  add2wc PORT MAP(CI,A1,A0,B1,B0,SUM1,SUM0,C2);
  i2:  add2nc PORT MAP(C2,A3,A2,B3,B2,SUM3,SUM2);
  i3:  add2nc PORT MAP(C4,A5,A4,B5,B4,SUM5,SUM4);
  i4:  add2nc PORT MAP(C6,A7,A6,B7,B6,SUM7,SUM6);
  i5:  add2nc PORT MAP(C8,A9,A8,B9,B8,SUM9,SUM8);
  i6:  add2nc PORT MAP(C10,A11,A10,B11,B10,SUM11,SUM10);

  E1 <= (A3 AND B3) OR ((A3 OR B3) AND (A2 AND B2));
  R1 <= (A3 OR B3) AND (A2 OR B2);

  C4 <= E1 OR (C2 AND R1);

  E2 <= (A5 AND B5) OR ((A5 OR B5) AND (A4 AND B4));
  R2 <= (A5 OR B5) AND (A4 OR B4);

  C6 <= E2 OR ((E1 OR (C2 AND R1)) AND R2);

  E3 <= (A7 AND B7) OR ((A7 OR B7) AND (A6 AND B6));
  R3 <= (A7 OR B7) AND (A6 OR B6);

  C8 <= E3 OR ((E2 OR ((E1 OR (C2 AND R1)) AND R2)) AND R3);

  E4 <= (A9 AND B9) OR ((A9 OR B9) AND (A8 AND B8));
  R4 <= (A9 OR B9) AND (A8 OR B8);

  C10 <= E4 OR ((E3 OR ((E2 OR ((E1 OR (C2 AND R1)) AND R2)) AND R3)) AND
        R4);

  E5 <= (A11 AND B11) OR ((A11 OR B11) AND (A10 AND B10));
  R5 <= (A11 OR B11) AND (A10 OR B10);

  CO <= E5 OR ((E4 OR ((E3 OR ((E2 OR ((E1 OR (C2 AND R1)) AND R2)) AND
       R3)) AND R4)) AND R5);

END archfc2add12;
```

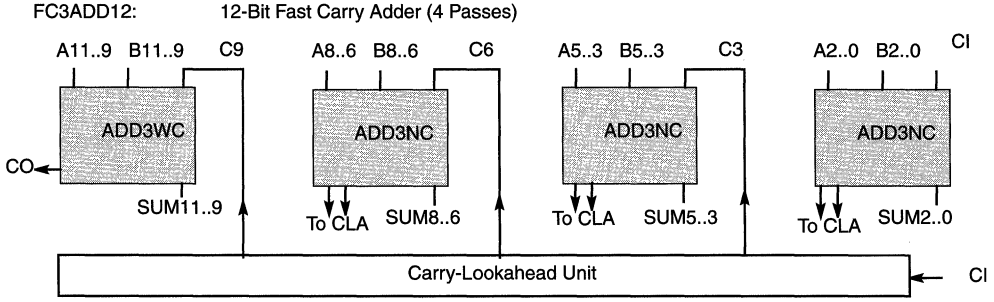


**Figure 8. 12-Bit Full Carry-Lookahead Adder using ADD3WC and ADD3NC**

### FC3ADD12: 12-Bit Full Carry-Lookahead Adder using a Group-Size of 3 Bits

This is very similar to the FC2ADD12, differing in the group-size of the adder used as the basic building block. The basic building blocks in this scheme are the ADD3WC and the ADD3NC components. The VHDL code attached and the block diagram in *Figure 8* illustrate the design. This scheme takes four passes through the logic array to yield all the results. The Es and the Rs are generated in the first pass. The intermediate carries C3, C6, and C9 are generated in the second pass. The carries internal to the group are generated in the third pass and the final SUM outputs in the fourth pass. As a different approach, the CO is generated by the MSB ADD3WC as opposed to the Carry-lookahead unit. This results in CO being generated in the third pass as opposed to the second pass. The VHDL code clearly indicates the manner in which the model is built.

For some bit-sizes, given that the 3-bit group-size is odd-numbered, the designer will have to choose a non-modular structure in building the adder. For example, a 32-bit adder cannot be built using just ADD3NCs and can be built using 10 ADD3NCs and one ADD2NC. The designer needs to choose the final implementation based on the constraints of the application.

```
--12-Bit Fast carry-Lookahead adder with 3-bit groups
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE WORK.ADD3WC.ALL;
USE WORK.ADD3NC.ALL;
ENTITY fc3add12 IS
  PORT (
    A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0 : IN STD_LOGIC;
    B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0 : IN STD_LOGIC;
    CI : IN STD_LOGIC;
    CO : OUT STD_LOGIC;
    SUM11,SUM10,SUM9,SUM8,SUM7,SUM6,SUM5,SUM4,SUM3,
    SUM2,SUM1,SUM0 : OUT STD_LOGIC);
END fc3add12;
ARCHITECTURE fc3add12arch OF fc3add12 IS
SIGNAL  E1,E2,E3 : STD_LOGIC;
SIGNAL  R1,R2,R3  : STD_LOGIC;
SIGNAL  C3,C6,C9 : STD_LOGIC;
attribute synthesis_off of C3,C6,C9 : signal is true;
attribute synthesis_off of E1,E2,E3 : signal is true;
attribute synthesis_off of R1,R2,R3 : signal is true;

BEGIN

i1: add3nc PORT MAP(CI,A2,A1,A0,B2,B1,B0,SUM2,SUM1,SUM0);
i2: add3nc PORT MAP(C3,A5,A4,A3,B5,B4,B3,SUM5,SUM4,SUM3);

i3: add3nc PORT MAP(C6,A8,A7,A6,B8,B7,B6,SUM8,SUM7,SUM6);
i4: add3wc PORT MAP(C9,A11,A10,A9,B11,B10,B9,SUM11,SUM10,SUM9,CO);
E1 <=  (A2 AND B2)
  OR ((A1 AND B1) AND (A2 OR B2))
  OR ((A0 AND B0) AND (A1 OR B1) AND (A2 OR B2));

R1 <=  (A2 OR B2) AND (A1 OR B1) AND (A0 AND B0);

C3 <= E1 OR (R1 AND CI);

E2 <=  (A5 AND B5)
  OR ((A4 AND B4) AND (A5 OR B5))
  OR ((A3 AND B3) AND (A4 OR B4) AND (A5 OR B5));

R2 <=  (A5 OR B5) AND (A4 OR B4) AND (A3 AND B3);

C6 <= E2 OR (E1 AND R2) OR (R2 AND R1 AND CI);

E3 <=  (A8 AND B8)
  OR ((A7 AND B7) AND (A8 OR B8))
  OR ((A6 AND B6) AND (A7 OR B7) AND (A8 OR B8));

R3 <=  (A8 OR B8) AND (A7 OR B7) AND (A6 AND B6);

C9 <= E3 OR (E2 AND R3) OR (E1 AND R3 AND R2) OR (R3 AND R2 AND R1 AND CI);
END fc3add12arch;
```

### FC4ADD12: 12-Bit Full Carry-Lookahead Adder using a Group-Size of 4 Bits

This is very similar to the FC2ADD12 and, again, differs in the group-size of the adder used as the basic building block. The basic building block in this scheme is the ADD4NC component. The ADD4NC component is built using a combination of ADD2WC and ADD2NC in the same order. This component is replicated to create the adder of the desired size. In the very last stage, two ADD2WCs are used instead of an ADD2WC and an ADD2NC. The VHDL code attached and the block diagram in *Figure 9* illustrate the design's functionality. This scheme takes four passes through the logic array to yield results. The various Es and Rs are generated in the first pass, the values of C4 and C8 in the second pass, the outputs from all the ADD2WCs in the third pass, and the outputs from ADD2NC in the fourth pass. Note that the value of CO is generated in the second pass. This scheme uses fewer MCs and more PTs than the previously mentioned schemes. The resource utilization of this model is shown in *Table 1.*

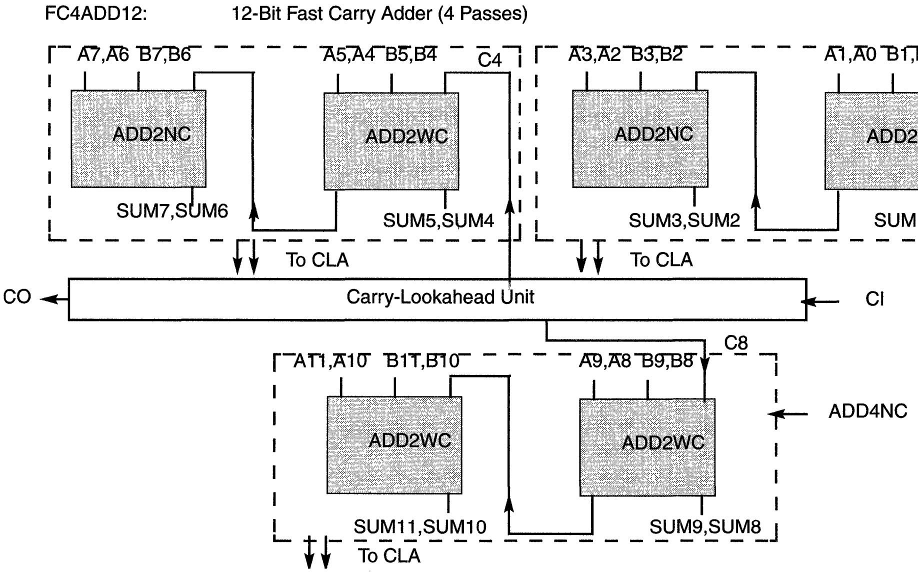


**Figure 9. 12-Bit Full Carry-Lookahead Adder using ADD4NC**

```
--A 12-bit Full carry-lookahead adder built using the ADD2WC and ADD2NC
--elements. The ADD2WC and ADD2NC elements are part of the ADD4NC in the
--same order

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE WORK.ADD2WC.ALL;
USE WORK.ADD2NC.ALL;

ENTITY fc4add12 IS
  PORT (
    A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0 : IN STD_LOGIC;
    B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0 : IN STD_LOGIC;
    CI : IN STD_LOGIC;
    CO : OUT STD_LOGIC;
    SUM11,SUM10,SUM9,SUM8,SUM7,SUM6,SUM5,SUM4,SUM3,
    SUM2,SUM1,SUM0 : OUT STD_LOGIC);
END fc4add12;
ARCHITECTURE fc4add12arch OF fc4add12 IS
```

```
 SIGNAL E1,E2 : STD_LOGIC;
 SIGNAL R1,R2  : STD_LOGIC;
 SIGNAL C2,C4,C6,C8,C10 : STD_LOGIC;
attribute synthesis_off of C2,C4,C6,C8,C10 : signal is true;
attribute synthesis_off of E1,E2 : signal is true;
attribute synthesis_off of R1,R2 : signal is true;
BEGIN
i1: add2wc PORT MAP(CI,A1,A0,B1,B0,SUM1,SUM0,C2);
i2: add2nc PORT MAP(C2,A3,A2,B3,B2,SUM3,SUM2);
i3: add2wc PORT MAP(C4,A5,A4,B5,B4,SUM5,SUM4,C6);
i4: add2nc PORT MAP(C6,A7,A6,B7,B6,SUM7,SUM6);
i5: add2wc PORT MAP(C8,A9,A8,B9,B8,SUM9,SUM8,C10);
i6: add2wc PORT MAP(C10,A11,A10,B11,B10,SUM11,SUM10,CO);
E1 <=  (A3 AND B3)
  OR ((A2 AND B2) AND (A3 OR B3))
  OR ((A1 AND B1) AND (A2 OR B2) AND (A3 OR B3))
  OR ((A0 AND B0) AND (A1 OR B1) AND (A2 OR B2) AND (A3 OR B3));
R1 <= (A3 OR B3) AND (A2 OR B2) AND (A1 OR B1) AND (A0 AND B0);
C4 <= E1 OR (R1 AND CI);
E2 <=  (A7 AND B7)
  OR ((A6 AND B6) AND (A7 OR B7))
  OR ((A5 AND B5) AND (A6 OR B6) AND (A7 OR B7))
  OR ((A4 AND B4) AND (A5 OR B5) AND (A6 OR B6) AND (A7 OR B7));

R2 <= (A7 OR B7) AND (A6 OR B6) AND (A5 OR B5) AND (A4 AND B4);

C8 <= E2 OR (E1 AND R2) OR (R2 AND R1 AND CI);
END fc4add12arch;
```

## Subtracters

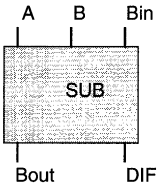
Subtracters are just a modified form of adders. The discussion presented for the adders can be easily extended to the subtracters. For any given sized adder or subtracter, the resource utilization is exactly the same for both in all respects.

### SUB: 1-Bit Full Subtracter

The basic component used in subtracting two operands is called a *Full subtracter.* The full subtracter element will be referred to as the 'SUB' component. The block diagram and functionality of SUB is shown in *Figure 10.* A (minuend) and B (subtrahend) are the two operands to be subtracted and Bin is the Borrow-in to the component. DIF and Bout are the Difference and Borrow-out from the component.

The VHDL code describing the functionality of the SUB component is shown here. This design takes one pass through the Product Term Matrix logic array to fit into a FLASH370i CPLD. The SUB component instantiated in the VHDL code has the exact same functionality shown in *Figure 10.*

SUB: 1-Bit Full Subtracter (1 Pass)



(Basic building block)

Functionality: DIF = *NOT* (*NOT* (A *XOR* B) *XOR* Bin)

Bout = (*NOT* A *AND* B) OR (*NOT* A *AND* CI) or (B *AND* CI)

**Figure 10. Block Diagram and Functionality of a Full Subtracter**

```
-- This VHDL code implements the element SUB within the package MATHPKG
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
```

```
PACKAGE mathpkg IS
  COMPONENT sub
    PORT (BIN: IN STD_LOGIC;
      A, B: IN STD_LOGIC;
      DIF: OUT STD_LOGIC;
      BOUT: OUT STD_LOGIC);
  END COMPONENT;
END mathpkg;

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY sub IS
  PORT (Bin: IN STD_LOGIC;
    A, B: IN STD_LOGIC;
    DIF: OUT STD_LOGIC;
     Bout: OUT STD_LOGIC);
END sub;

ARCHITECTURE archsub OF sub IS
BEGIN
  DIF <= NOT (NOT (A0 XOR B0) XOR Bin);
  Bout <= (A and (not B)) or (A and Bin) or ((not B) and Bin);
END archsub;
```

### SUB2WB: A 2-Bit Subtracter with a Borrow-Out

The structure of a 2-bit group subtracter (SUB2WB) is very similar to that of the ADD2WC and is shown here. This component can be used as a building block to build larger sized subtracters, exactly like ADD2WC was used to build larger sized adders. The block diagram of the SUB2WB is shown in *Figure 11.* The corresponding VHDL code used to describe the functionality of the SUB2WB is also attached. As in the case of ADD2WC, the functionality for SUB2WB is realized in one pass through the logic array.

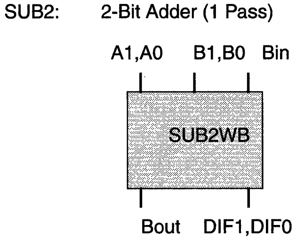


**Figure 11. Block Diagram of a 2-Bit Subtracter with a Borrow-Out**

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

PACKAGE sub2wb_pkg IS
  COMPONENT sub2wb PORT(
    Bin : IN STD_LOGIC;
        A1,A0: IN STD_LOGIC;
        B1,B0: IN STD_LOGIC;
        DIF1,DIF0  : OUT STD_LOGIC;
        Bout: OUT STD_LOGIC);
  END COMPONENT;
END sub2wb_pkg;
ENTITY sub2wb IS
  PORT (Bin : IN STD_LOGIC;
    A1,A0: IN STD_LOGIC;
    B1,B0: IN STD_LOGIC;
    DIF1,DIF0  : OUT STD_LOGIC;
    Bout: OUT STD_LOGIC);
END sub2wb;

ARCHITECTURE archsub2wb OF sub2wb IS

BEGIN
```

```
DIF0 <= NOT (NOT (A0 XOR B0) XOR Bin);
DIF1 <= NOT (NOT (A1 XOR B1) XOR ((NOT A0 AND B0) OR (NOT A0 AND Bin) OR
        (B0 AND Bin)));

Bout    <= (NOT A0 AND B0 AND B1)
  OR (NOT A0 AND B0 AND NOT A1)
  OR (BI AND B0 AND B1)
  OR (BI AND B0 AND NOT A1)
  OR (BI AND NOT A0 AND B1)
  OR (BI AND NOT A0 AND NOT A1)
  OR (NOT A1 AND B1);

END archsub2wb;
```

### FB2SUB12: 12-Bit Full Borrow-Lookahead Subtracter using 2-Bit Subtracters

It was mentioned before that we can build equivalent subtracter models for all the adder models discussed earlier. The functionality and the implementation of an FB2SUB12 (subtracter equivalent of an FC2ADD12) is shown here as an example. The implementation of all the possible subtracter elements is not discussed in this application note, since the concept involved in building them is identical to that of the adders.

The block diagram of the FB2SUB12 is very similar to that of the adder element FC2ADD12 and is shown in *Figure 12.* The FB2SUB12 is built using the basic elements SUB2WB and SUB2NC (2-bit subtracter with no borrow-out). This takes three passes through the logic array. The values of the various Es and Rs are generated in the first pass, the intermediate carries (borrows) in the second pass, and the various DIFs in the third pass. Note that the value of BO is generated in the second pass. The VHDL code for FB2SUB12 is also attached.

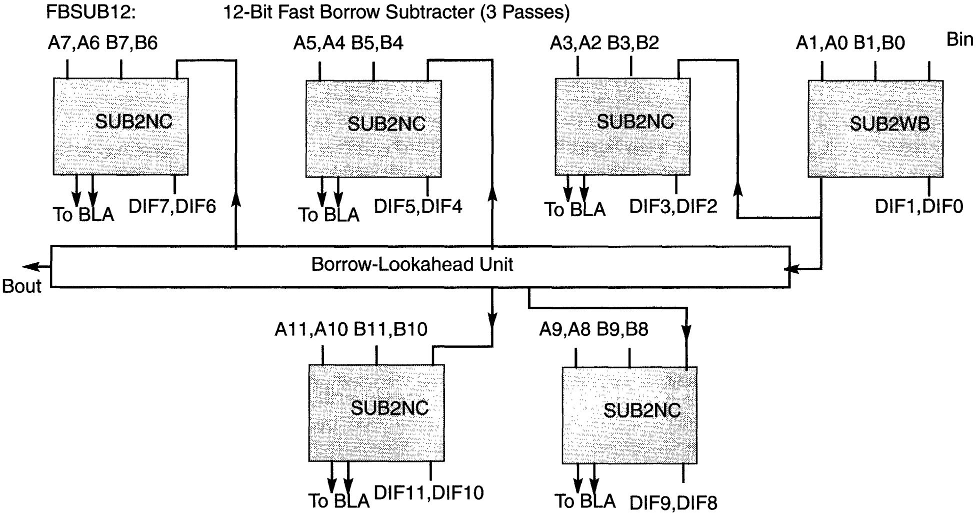


**Figure 12. 12-Bit Fast Borrow Subtracter Built using SUB2WB and SUB2NC**

```
--A 12-bit Full borrow-lookahead subtracter built using the SUB2WC and
--SUB2NC elements

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE WORK.SUB2WB.ALL;
USE WORK.SUB2NC.ALL;

ENTITY fb2sub12 IS
  PORT (Bin : IN STD_LOGIC;
       A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0: IN STD_LOGIC;
       B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0: IN STD_LOGIC;
```

```
    DIF11,DIF10,DIF9,DIF8,DIF7,DIF6,DIF5,DIF4,
    DIF3,DIF2,DIF1,DIF0 : OUT STD_LOGIC;
    Bout: OUT STD_LOGIC);
END fb2sub12;

ARCHITECTURE archfb2sb12 OF fb2sub12 IS

 SIGNAL C2, C4, C6, C8, C10 : STD_LOGIC;
 SIGNAL E1,E2,E3,E4,E5   : STD_LOGIC;
 SIGNAL R1,R2,R3,R4,R5   : STD_LOGIC;

--The internal carries are referred to as C's to distinguish between
--borrow-out's and the operands
 attribute synthesis_off of E1,E2,E3,E4,E5 : signal is true;
 attribute synthesis_off of R1,R2,R3,R4,R5 : signal is true;
 attribute synthesis_off of C2, C4, C6, C8, C10 : signal is true;

BEGIN

 i1:  sub2wb PORT MAP(Bin,A1,A0,B1,B0,DIF1,DIF0,C2);
 i2:  sub2nc PORT MAP(C2,A3,A2,B3,B2,DIF3,DIF2);
 i3:  sub2nc PORT MAP(C4,A5,A4,B5,B4,DIF5,DIF4);
 i4:  sub2nc PORT MAP(C6,A7,A6,B7,B6,DIF7,DIF6);
 i5:  sub2nc PORT MAP(C8,A9,A8,B9,B8,DIF9,DIF8);
 i6:  sub2nc PORT MAP(C10,A11,A10,B11,B10,DIF11,DIF10);

  E1 <= (NOT A3 AND B3) OR ((NOT A3 OR B3) AND (NOT A2 AND B2));
  R1 <= (NOT A3 OR B3) AND (NOT A2 OR B2);

  C4 <= E1 OR (C2 AND R1);

  E2 <= (NOT A5 AND B5) OR ((NOT A5 OR B5) AND (NOT A4 AND B4));
  R2 <= (NOT A5 OR B5) AND (NOT A4 OR B4);

  C6 <= E2 OR ((E1 OR (C2 AND R1)) AND R2);

  E3 <= (NOT A7 AND B7) OR ((NOT A7 OR B7) AND (NOT A6 AND B6));
  R3 <= (NOT A7 OR B7) AND (NOT A6 OR B6);

  C8 <= E3 OR ((E2 OR ((E1 OR (C2 AND R1)) AND R2)) AND R3);

  E4 <= (NOT A9 AND B9) OR ((NOT A9 OR B9) AND (NOT A8 AND B8));
  R4 <= (NOT A9 OR B9) AND (NOT A8 OR B8);

  C10 <= E4 OR ((E3 OR ((E2 OR ((E1 OR (C2 AND R1)) AND R2)) AND R3))
         AND R4);

  E5 <= (NOT A11 AND B11) OR ((NOT A11 OR B11) AND (NOT A10 AND B10));
  R5 <= (NOT A11 OR B11) AND (NOT A10 OR B10);

  Bouy <= E5 OR ((E4 OR ((E3 OR ((E2 OR ((E1 OR (C2 AND R1)) AND R2))
          AND R3)) AND R4)) AND R5);

END archfb2sub12;
```

**Table 1. Comparison of Different 12-Bit Adder Schemes.**

| Resource | R1ADD12 | R2ADD12 | R3ADD12 | FC2ADD12 | FC3ADD12 | FC4ADD12 |
|---|---|---|---|---|---|---|
| PTs used | 84 | 138 | 165 | 148 | 153 | 169 |
| MCs used | 24 | 18 | 16 | 28 | 26 | 22 |
| # of passes | 12 | 6 | 5 | 3 | 4 | 4 |

## Comparison of Resource Utilization for Different Schemes in Building a 12-Bit Adder

A comparison chart showing the resource utilization for the different models that can be used in building a 12-bit adder is shown in *Table 1*. This table summarizes some of the key issues that have been presented in the discussion so far. Some comparisons and comments from the charts and are listed here:

### Ripple Carry Adders

1. For a given group-size, the number of passes taken to yield results is *dependent* on the size of the adder being built.
2. As the group-size increases, the number of passes taken through the logic array is *(n/k) -1 + # of passes for final stage*, where *n* is the size of the adder and *k* is the group size. For example, a R2ADD12 takes *(12/2) -1 + 1* = 6 passes to yield the desired result.
3. In the R3ADD12 (ripple carry adder built using 3-bit groups) scheme, the value of the MSB sum bit within a 3-bit group is produced only in the second pass through the array. This, however, does not affect the 12-bit adder yielding results in 5 passes *(12/3) -1 + 2 = 5)* as expected. This is possible because the carry-out from the 3-bit group is produced in the first pass. The implementation of the ADD3WC was discussed in detail earlier. This solution is a very desirable solution for most applications that use small sized adders.
4. The R1ADD12 uses fewer PTs and more MCs among the different versions of ripple-carry adders. The opposite is the case for the R3ADD12. The R2ADD12 provides an intermediate solution between the two extremes.
5. The macrocell count in R1ADD12 can be brought down from 24 to 18, if the attribute 'synthesis_off' is used on the even-numbered carries only. The number of passes is also brought down from 12 to 6. This, however, pushes the product term count from 84 to 138. In either case, none of the equations sum-split. This is, in fact, R2ADD12. The designer can choose the implementation that best chooses the application.
6. The R4ADD12 (ripple carry adder built using 4-bit groups) is not a viable solution, since the carry-out from one of the 4-bit groups would take two passes to be generated. This results in a implementation that takes six passes to yield results as opposed to the expected three passes. This solution is inefficient and is not considered.

### Carry-Lookahead Adders

1. For a given group-size, the number of passes taken to yield results is *largely independent* of the size of the adder being built. This is the biggest advantage with carry-lookahead adders.
2. All the *group generates* (*Es*) and *group propagates* (*Rs*) are generated in the first pass and the carry-ins to all groups in the second pass through the logic array. The Sum outputs are generated in the third or the fourth pass, depending on the group-size being used.
3. The FC2ADD12 takes three passes to complete, and four passes for the FC3ADD12 and FC4ADD12. The number of passes *remains the same* up to 32-bit versions of the adder.
4. Similar to the ripple carry adders, the FC2ADD12 uses fewer PTs and more MCs among the different versions of carry-lookahead adders. The opposite is the case for the FC4ADD12. The FC3ADD12 provides an intermediate solution between the two extremes.
5. The FC5ADD12 (carry-lookahead adder built using 5-bit groups) is not a viable solution, since the extra number of PTs and number of passes (5) taken through the logic array do not justify its usage. The design is also not modular and difficult to deal with. A designer can, however, extend the discussion presented to build his own FC5ADD12 model if the application demands it. This, however, would be an extreme case and is not presented.

### Summary

Comparing ripple carry and carry-lookahead adders, it is evident that ripple carry adders are area efficient but have poor speed performance. The carry-lookahead adders on the other hand are faster but utilize more resources. Given the different choices, the user needs to make a careful selection of the scheme best suited for his application.

## Large-Sized Adders/Subtracters

As PLDs have grown in size and speed over the past few years, a lot of designers have been pushing towards larger sized adders and subtracters. A lot of focus and time has been dedicated towards building efficient smaller sized elements and it was left to the designer's discretion to build and implement the algorithm. Cypress believes in providing designers with the best possible implementation for their adders/subtracters and relieve them of the problems they would normally face. This provides the customer with the opportunity to get the best implementation for their application without spending a lot of time.

*Table 2* talks about the resource utilization for 24-bit and 32-bit adders using 2-bit, 3-bit, and 4-bit group-sizes with carry/borrow-lookahead principle. In the previous sections, different implementation strategies and the VHDL codes for a 12-bit full-carry-lookahead adder were shown as an example. The VHDL codes for most variations of the 24- and 32-bit implementations are not presented here due to space constraints. The codes are provided, however, as a part of the tutorial section in the *Warp* VHDL compiler. *Figure 12* illustrates three schemes used in implementing a 24-bit adder. The VHDL code for a 24-bit carry-lookahead adder with a 4-bit group size is shown here as an example. The code for other models is very similar and can be easily extrapolated.

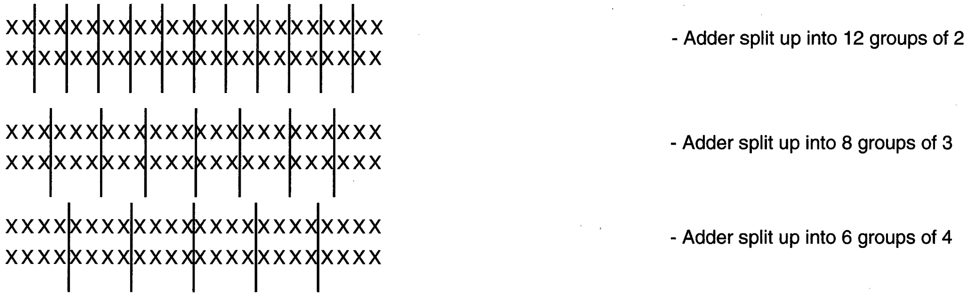

- Adder split up into 12 groups of 2
- Adder split up into 8 groups of 3
- Adder split up into 6 groups of 4

**Figure 13. Three Different Carry-Lookahead Schemes to Implement a 24-Bit Adder**

**Table 2. Comparison of Different 24-Bit and 32-Bit Adder Schemes.**

| Resource | FC2ADD24 | FC3ADD24 | FC4ADD24 | FC2ADD32 | FC3ADD32 | FC4ADD32 |
|---|---|---|---|---|---|---|
| PTs used | 272 | 314 | 359 | 393 | 427 | 488 |
| MCs used | 58 | 54 | 46 | 78 | 73 | 62 |
| # of passes | 3 | 4 | 4 | 3 | 4 | 4 |

```
--24-bit Fast Carry lookahead adder with 4-bit groups
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE work.add2wc_pkg.all;
USE work.add2nc_pkg.all;
ENTITY fc4add24 IS
PORT (
A23,A22,A21,A20,A19,A18,A17,A16,A15,A14,A13,A12,
A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0 : IN STD_LOGIC;
B23,B22,B21,B20,B19,B18,B17,B16,B15,B14,B13,B12,
B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0 : IN STD_LOGIC;
CI : IN STD_LOGIC;
CO : OUT STD_LOGIC;
SUM23,SUM22,SUM21,SUM20,SUM19,SUM18,SUM17,SUM16,SUM15,SUM14,SUM13,SUM12,SUM11,SUM10,SUM9,SUM8
,SUM7,SUM6,SUM5,SUM4,SUM3,SUM2,SUM1,SUM0: OUT STD_LOGIC);
END fc4add24;
ARCHITECTURE fc4add24arch OF fc4add24 IS
SIGNAL  E1,E2,E3,E4,E5 : STD_LOGIC;
SIGNAL  R1,R2,R3,R4,R5  : STD_LOGIC;
SIGNAL  C2,C4,C6,C8,C10,C12,C14,C16,C18,C20,C22 : STD_LOGIC;
attribute synthesis_off of C2,C4,C6,C8,C10,C12,C14,C16,C18,C20,C22 : signal is true;
attribute synthesis_off of E1,E2,E3,E4,E5 : signal is true;
attribute synthesis_off of R1,R2,R3,R4,R5 : signal is true;
BEGIN
i1: add2wc PORT MAP (CI,A1,A0,B1,B0,SUM1,SUM0,C2);
i2: add2nc PORT MAP (C2,A3,A2,B3,B2,SUM3,SUM2);
i3: add2wc PORT MAP (C4,A5,A4,B5,B4,SUM5,SUM4,C6);
i4: add2nc PORT MAP (C6,A7,A6,B7,B6,SUM7,SUM6);
i5: add2wc PORT MAP (C8,A9,A8,B9,B8,SUM9,SUM8,C10);
i6: add2nc PORT MAP (C10,A11,A10,B11,B10,SUM11,SUM10);
i7: add2wc PORT MAP (C12,A13,A12,B13,B12,SUM13,SUM12,C14);
i8: add2nc PORT MAP (C14,A15,A14,B15,B14,SUM15,SUM14);
```

```
i9: add2wc PORT MAP (C16,A17,A16,B17,B16,SUM17,SUM16,C18);
i10: add2nc PORT MAP (C18,A19,A18,B19,B18,SUM19,SUM18);
i11: add2wc PORT MAP (C20,A21,A20,B21,B20,SUM21,SUM20,C22);
i12: add2wc PORT MAP (C22,A23,A22,B23,B22,SUM23,SUM22,Co);
E1 <=  (A3 AND B3)
  OR ((A2 AND B2) AND (A3 OR B3))
  OR ((A1 AND B1) AND (A2 OR B2) AND (A3 OR B3))
  OR ((A0 AND B0) AND (A1 OR B1) AND (A2 OR B2) AND (A3 OR B3));
R1 <= (A3 OR B3) AND (A2 OR B2) AND (A1 OR B1) AND (A0 AND B0);
C4 <= E1 OR (R1 AND CI);
E2 <=  (A7 AND B7)
  OR ((A6 AND B6) AND (A7 OR B7))
  OR ((A5 AND B5) AND (A6 OR B6) AND (A7 OR B7))
  OR ((A4 AND B4) AND (A5 OR B5) AND (A6 OR B6) AND (A7 OR B7));

R2 <= (A7 OR B7) AND (A6 OR B6) AND (A5 OR B5) AND (A4 AND B4);

C8 <= E2 OR (E1 AND R2) OR (R2 AND R1 AND CI);
E3 <=  (A11 AND B11)
  OR ((A10 AND B10) AND (A11 OR B11))
  OR ((A9 AND B9) AND (A10 OR B10) AND (A11 OR B11))
  OR ((A8 AND B8) AND (A9 OR B9) AND (A10 OR B10) AND (A11 OR B11));

R3 <= (A11 OR B11) AND (A10 OR B10) AND (A9 OR B9) AND (A8 AND B8);

C12 <= E3 OR (E2 AND R3) OR (E1 AND R3 AND R2)  OR (R3 AND R2 AND R1 AND
CI);
E4 <=  (A15 AND B15)
  OR ((A14 AND B14) AND (A15 OR B15))
  OR ((A13 AND B13) AND (A14 OR B14) AND (A15 OR B15))
  OR ((A12 AND B12) AND (A13 OR B13) AND (A14 OR B14) AND (A15 OR B15));

R4 <= (A15 OR B15) AND (A14 OR B14) AND (A13 OR B13) AND (A12 AND B12);

C16 <= E4 OR (E3 AND R4) OR (E2 AND R4 AND R3) OR (E1 AND R4 AND R3 AND R2)
OR (R3 AND R2 AND R1 AND CI);
E5 <=  (A19 AND B19)
  OR ((A18 AND B18) AND (A19 OR B19))
  OR ((A17 AND B17) AND (A18 OR B18) AND (A19 OR B19))
  OR ((A16 AND B16) AND (A17 OR B17) AND (A18 OR B18) AND (A19 OR B19));
R5 <= (A19 OR B19) AND (A18 OR B18) AND (A17 OR B17) AND (A16 AND B16);

C20 <= E5 OR (E4 AND R5) OR (E3 AND R5 AND R4) OR (E2  AND R5  AND R4 AND
  R3) OR (E1 AND R5 AND R4 AND R3 AND R2) OR (R5 AND R4 AND R3 AND R2 AND
  R1 AND CI);
END fc4add24arch;
```

## Equality Comparators

Equality comparators are used often to compare the value of two operands. Equality comparators are built using the Exclusive-OR gate as the building block. A bit-wise comparison of the two data streams is done using XOR gates and each of the individual results are OR-ed together to obtain the final result.

### EQCOMP4: 4-Bit Equality Comparator

The EQCOMP4 is a 4-bit equality compare element. The model can be described as:

```
EQ = NOT ((A3 XOR B3)
    OR (A2 XOR B2)
    OR (A1 XOR B1)
    OR (A0 XOR B0))
```

This implementation takes 8 PTs. *Figure 14* shows the block diagram for EQCOMP4. NEQCOMP4 is the 4-bit non-equality comparator. The EQCOMP4 is implemented as an inverted version of the NEQCOMP4. The NEQCOMP4 element takes 8 PTs and the EQCOMP4 takes 16 PTs. The FLASH370i CPLD has a polarity control in the macrocell and can create the EQCOMP4 element using the NEQCOMP4 element, resulting in a implementation with a reduced product term count.

The equality comparator for all bit sizes greater than 8 takes more than 16 PTs to produce the result and takes two passes,

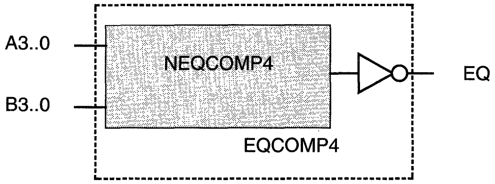


**Figure 14. Block Diagram of a 4-Bit Equality Compare**

since the FLASH370i CPLD architecture takes in a maximum of 16 PTs into one macrocell.

### EQCOMP24: 24-Bit Equality Comparator

The EQCOMP24 uses three EQCOMP8s in parallel and combines the results of the three components to produce the result. This takes two passes through the logic array, 4 MCs, and 49 PTs. The block diagram of this model is shown in *Figure 15*.

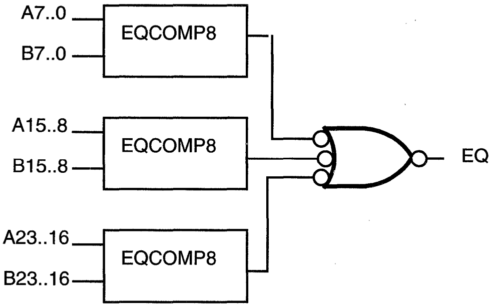


**Figure 15. Block Diagram of a 24-Bit Equality Compare**

```
-- Flattened version of the Magnitude comparator

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE work.std_arith.all;
ENTITY magcomp IS
PORT (
A,B : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
MAG : OUT STD_LOGIC);
END magcomp;
ARCHITECTURE magarch OF magcomp IS
BEGIN
MAG <= '1' WHEN (A < B) ELSE '0';
END magarch;
```

## Magnitude Comparators

Magnitude comparators are also widely used in the industry in comparing values of two operands. The magnitude comparators provide information if a signal is greater than (>), or less than (<) another signal of the same length.

### MAGCOMP8: 8-Bit Magnitude Comparator

This is the generic implementation of a magnitude comparator and does a bit-wise comparison, similar to that of the equality comparison. However, in the case of a magnitude comparator the results of a bit-wise comparison are to be retained and passed onto the succeeding set of bits. This passage of information continues and tends to increase the resource utilization of the design exponentially.

The VHDL implementation of an 8-bit magnitude comparator is shown here. The design takes 255 PTs and fits in two passes through the logic array. The block diagram of MAGCOMP8 is shown in *Figure 16*.

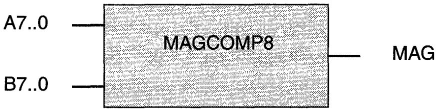


**Figure 16. Block Diagram of an 8-Bit Magnitude Compare**

A fully flattened implementation of a magnitude comparator would take ($2^n$ - 1) PTs to implement. It is, however, not recommended to use the fully-flattened version of the magnitude comparator for any bit-size greater than 4 bits. This is to ensure that there is no sum-splitting involved in the equations. There are other means to achieve better results and the best scheme is presented next.

### FB2MGCMP8: 8-Bit Borrow-Lookahead Magnitude Comparator

The block diagram of a 8-bit magnitude compare is shown in *Figure 17.*

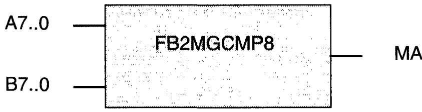


**Figure 17. Block Diagram of an 8-Bit Magnitude Compare**

This scheme uses a different approach to compare the magnitudes of two binary bit vectors. As an example, the scheme is illustrated for a 8-bit magnitude comparator. The 4 MSB bits of the bit vectors A[7:0] and B[7:0] are called $A_M$ and $B_M$, respectively. Similarly, the 4 LSB bits are referred to as $A_L$ and $B_L$ respectively. The bit vector A is greater than B if ($A_M > B_M$) or if ($A_M = B_M$) and ($A_l > B_L$).

It is evident from the set of equations in *Figure 18* that the magnitude comparison of two binary bit vectors can be done by evaluating the values of $G_M$, $G_L$ and $P_M$. $G_M$ and $G_L$ are the *generate* functions for the MSHalf (most significant half) and the LSHalf (least significant half) for the two bit vectors and $P_M$ is the *propagate* function for the MSHalf. This scheme is a stripped down version of the borrow-lookahead scheme used to build fast subtracters. In this implementation we need to determine the values of the *generate* and *propagate* functions for the bit vectors and need not produce any of the difference results. The borrow-out signal determines the output of the magnitude comparison. If the borrow-out is a '1' then (A < B), else (A q B).

This scheme allows for a fast and efficient means to do magnitude comparisons. Magnitude Comparators up to 32 bits can be built to produce the result in just 2 passes. The number of PTs used is also substantially less than the 'flattened' implementation of the magnitude comparators.

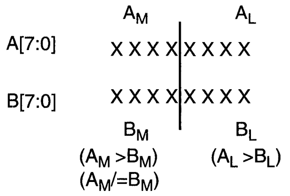
The discussion presented earlier on group-sizes can also be extended here. The group-size over which the *propagate* and *generate* functions are generated can be varied to be 2, 3 or 4. In *all* cases the design takes 2 passes to produce the desired result. The various values of Es and Rs are generated in the first pass and the value of the borrow-out in the second pass. However, there is a trade-off between the number of PTs and MCs used among the different group-sizes chosen. A comparison between these different implementations is discussed later.

The number of PTs used to implement the $P_M$ (propagate) function can be halved if 'OR' gates are used instead of 'XOR' gates. This was mentioned earlier in the discussion on carry-lookahead. This extension makes the implementation of the borrow-lookahead magnitude comparator fast and efficient.

### Comparison of Two Implementations of a 12-Bit Magnitude Compare

Two different implementations of a 12-bit magnitude comparator are shown here. The first implementation is an extension of MAGCOMP4. The second implementation uses the borrow-lookahead scheme and is built using borrow-lookahead over a group-size of 2 bits. This comparison illustrates the advantage of using FB2MGCMP12 over the simple MAGCOMP12.

The block diagram of MAGCOMP12 is shown in *Figure 19.* The flattened version of MAGCOMP12 takes ($2^{12}$ - 1) PTs. This is a large amount of logic and will not fit into any of the FLASH370i CPLDs. The MAGCOMP12 with the synthesis_off attribute on the intermediate signals uses 44 unique PTs, but is very slow and takes 11 passes through the array.



$$(A > B) = \underbrace{(A_M > B_M)}_{G_M} + \underbrace{\overline{((A_M /= B_M)}}_{P_M} * \underbrace{(A_L > B_L))}_{G_L}$$

$$(A > B) = G_M + P_M * G_L$$

**Figure 18. Bit Vector Magnitude Comparison Equations**

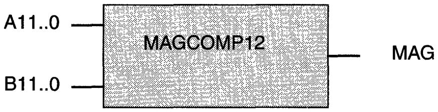


**Figure 19. Block Diagram of a 12-Bit Magnitude Compare**

The block diagram of FB2MGCMP12 is shown in *Figure 20.* The VHDL code for this design is also shown here. This design takes just two passes through the array and uses 36 unique PTs. The various values of Es and Rs are generated in the first pass and the value of the borrow-out in the second pass. Each of the Es uses 3 PTs and Rs 2 PTs and the output MAG takes 6 PTs. This is clearly a much better implementation than the MAGCOMP12.

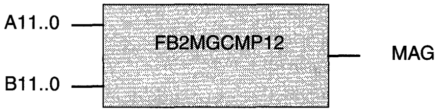


**Figure 20. Block Diagram of a 12-Bit Magnitude Compare with Borrow-Lookahead**

```
--The borrow-lookahead principle using 2-bit groups was used to build this
--element

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY fb2mgcmp12 IS
  PORT (
    A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0: IN STD_LOGIC;
    B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0: IN STD_LOGIC;
        MAG: OUT STD_LOGIC);
END fb2mgcmp12;

ARCHITECTURE archfb2mgcmp12 OF fb2mgcmp12 IS

  SIGNAL E0,E1,E2,E3,E4,E5  : STD_LOGIC;
  SIGNAL R0,R1,R2,R3,R4,R5  : STD_LOGIC;
  SIGNAL BO : STD_LOGIC;

  attribute synthesis_off of E0,E1,E2,E3,E4,E5 : signal is true;
  attribute synthesis_off of R0,R1,R2,R3,R4,R5 : signal is true;

BEGIN

    E0 <= (NOT A1 AND B1) OR ((NOT A1 OR B1) AND (NOT A0 AND B0));
    R0 <= (NOT A1 OR B1) AND (NOT A0 OR B0);

    E1 <= (NOT A3 AND B3) OR ((NOT A3 OR B3) AND (NOT A2 AND B2));
    R1 <= (NOT A3 OR B3) AND (NOT A2 OR B2);

    E2 <= (NOT A5 AND B5) OR ((NOT A5 OR B5) and (NOT A4 AND B4));
    R2 <= (NOT A5 OR B5) AND (NOT A4 OR B4);

    E3 <= (NOT A7 AND B7) OR ((NOT A7 OR B7) AND (NOT A6 AND B6));
    R3 <= (NOT A7 OR B7) AND (NOT A6 OR B6);

    E4 <= (NOT A9 AND B9) OR ((NOT A9 OR B9) AND (NOT A8 AND B8));
    R4 <= (NOT A9 OR B9) AND (NOT A8 OR B8);

    E5 <= (NOT A11 AND B11) OR ((NOT A11 OR B11) AND (NOT A10 AND B10));
     R5 <= (NOT A11 OR B11) AND (NOT A10 OR B10);

    BO <= E5 OR
        (R5 AND E4) OR
        (R5 AND R4 AND E3) OR
        (R5 AND R4 AND R3 AND E2) OR
        (R5 AND R4 AND R3 AND R2 AND E1) OR
        (R5 AND R4 AND R3 AND R2 AND R1 AND E0);
```

```
MAG <= '1' WHEN (BO = '1') ELSE '0';

--MAG is a '1' if B > A

END archfb2mgcmp12;
```

A comparison between 2-, 3-, and 4-bit group sized implementation of a 12-bit magnitude comparator based on the borrow-lookahead scheme is shown in *Table 3.* As mentioned before, the number of passes through the logic array is the same for all group-bit-sizes. The number of PTs and MCs used vary as shown in the table. The user has a wide choice and needs to choose the right group-size depending on the application.

**Table 3. Comparison of a 12-Bit Magnitude Compare between Different Group-Sizes.**

| Group-Bit-Size | 2 | 3 | 4 |
|---|---|---|---|
| # of PTs | 34 | 44 | 60 |
| # of MCs | 13 | 9 | 7 |
| # of passes | 2 | 2 | 2 |

## Three-Output Comparators

The discussion on magnitude comparators has so far been restricted to the values of less than (<) and greater than or equal to (q) only. The discussion in this section talks about producing all three outputs, namely '<', '>' and '='.

### FB2EQMCMP12: 12-Bit Borrow-Lookahead Three-Output Magnitude Comparator Using 2-Bit Groups

This model combines all the concepts discussed in the magnitude comparator section into one design. This uses borrow-lookahead, 2-bit groups, and also produces three outputs. The block diagram of this model is shown in *Figure 21.*

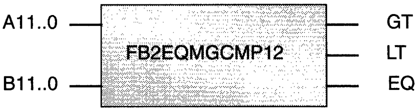


**Figure 21. Block Diagram of a 12-Bit Borrow-Lookahead Three-Output Magnitude Compare**

There are two ways in which the Borrow-lookahead principle can be used to achieve the functionality of a three-output comparator.

1. Use two passes for 'A < B' and 'A = B' each, then use a third pass for A > B using the results from A < B and A = B. This uses 62 PTs. The EQCOMP12 required for this model is built using three EQCOMP4s similar to the block diagram shown in *Figure 15.* The EQCOMP12 can also be built using four EQCOMPs, or two EQCOMP6s, or an EQCOMP8 and an EQCOMP4 or any other combination. As long as the EQCOMP model chosen does not sum-split, the value of EQCOMP12 can be realized in two passes using 25 PTs.
2. Use two passes to generate all three outputs. In this implementation a set of Es and Rs is required to create a value of LT (A - B). A second set of Es and Rs is required to obtain the value of GT (B - A). The value of EQ is also produced in 2 passes along with GT and LT. This scheme uses 97 PTs.

The first scheme is area efficient, but takes three passes though the logic array to generate the final results. The VHDL implementation for the first scheme is presented here. It is very easy to extrapolate the code for the second scheme.

```
--This VHDL code describes the implementation of a 3-output magnitude
--comparator. The borrow-lookahead principle using 2-bit groups was used
--to build this element

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY fb2eqmgcmp12 IS
  PORT (
    A11,A10,A9,A8,A7,A6,A5,A4,A3,A2,A1,A0: IN STD_LOGIC;
    B11,B10,B9,B8,B7,B6,B5,B4,B3,B2,B1,B0: IN STD_LOGIC;
    EQ,LT,GT: OUT STD_LOGIC);
END fb2eqmgcmp12;

ARCHITECTURE archfb2eqmgcmp12 OF fb2mgeqcmp12 IS

  SIGNAL E0,E1,E2,E3,E4,E5   : STD_LOGIC;
  SIGNAL R0,R1,R2,R3,R4,R5   : STD_LOGIC;
  SIGNAL X11,X10,X9,X8,X7,X6,X5,X4,X3,X2,X1,X0 : STD_LOGIC;
  SIGNAL INT1, INT2, INT3: STD_LOGIC;
  SIGNAL BO : STD_LOGIC;
  attribute synthesis_off of E0,E1,E2,E3,E4,E5 : signal is true;
  attribute synthesis_off of R0,R1,R2,R3,R4,R5 : signal is true;
  attribute synthesis_off of INT1, INT2, INT3 : signal is true;

BEGIN

   E0 <= (NOT A1 AND B1) OR ((NOT A1 OR B1) AND (NOT A0 AND B0));
      R0 <= (NOT A1 OR B1) AND (NOT A0 OR B0);

   E1 <= (NOT A3 AND B3) OR ((NOT A3 OR B3) AND (NOT A2 AND B2));
   R1 <= (NOT A3 OR B3) AND (NOT A2 OR B2);
   E2 <= (NOT A5 AND B5) OR ((NOT A5 OR B5) and (NOT A4 AND B4));
   R2 <= (NOT A5 OR B5) AND (NOT A4 OR B4);

   E3 <= (NOT A7 AND B7) OR ((NOT A7 OR B7) AND (NOT A6 AND B6));
   R3 <= (NOT A7 OR B7) AND (NOT A6 OR B6);

   E4 <= (NOT A9 AND B9) OR ((NOT A9 OR B9) AND (NOT A8 AND B8));
   R4 <= (NOT A9 OR B9) AND (NOT A8 OR B8);

   E5 <= (NOT A11 AND B11) OR ((NOT A11 OR B11) AND (NOT A10 AND B10));
   R5 <= (NOT A11 OR B11) AND (NOT A10 OR B10);

   BO <= E5 OR
         (E4 AND R5) OR
         (E3 AND R5 AND R4) OR
         (E2 AND R5 AND R4 AND R3) OR
         (E1 AND R5 AND R4 AND R3 AND R2) OR
         (E0 AND R5 AND R4 AND R3 AND R2 AND R1);

LT <= ’1’ WHEN (BO = ’1’) ELSE ’0’;

-- LT is a ’1’ if A < B

GT <= ’1’ WHEN (LT = ’0’ AND EQ = ’0’ ) ELSE ’0’;

-- GT is a ’1’ if A > B

   X11 <= A11 XOR B11;
   X10 <= A10 XOR B10;
   X9 <= A9 XOR B9;
   X8 <= A8 XOR B8;
   X7 <= A7 XOR B7;
   X6 <= A6 XOR B6;
   X5 <= A5 XOR B5;
   X4 <= A4 XOR B4;
   X3 <= A3 XOR B3;
   X2 <= A2 XOR B2;
   X1 <= A1 XOR B1;
   X0 <= A0 XOR B0;
   INT1 <= (X11 OR X10 OR X9 OR X8);
   INT2 <= (X7 OR X6 OR X5 OR X4);
   INT3 <= (X3 OR X2 OR X1 OR X0);

   EQ <= NOT (INT1 OR INT2 OR INT3);

END archfb2eqmgcmp12;
```

## Summary

A number of arithmetic elements frequently used in various applications were presented in this application note. The underlying concepts and the final implementations for all these models were also presented. Designs created with an understanding of the target architecture always perform better than generic designs. The LPM elements available in *Warp* are all geared towards obtaining the best performance, both in speed and area, for CPLDs. The concepts and implementations presented in this application note are used to build the various LPM elements. Understanding this application note will enable the user to understand the LPM elements better and exploit their availability in the best possible manner.

CPLDs are getting to be very popular with the programmable logic industry, and are widely used in DSP applications, PCs, Motherboards, Data Communication equipment, Multimedia, Instrumentation, etc. They have many advantages over other programmable logic devices. A few key advantages are listed here:

- Ease of use—Simple extension of AND-OR structure of small PLDs like 22V10
- Predictable timing model
- No fanout penalty
- Provide high speed of operation
- Off the shelf availability
- Cost effective solution

These advantages make CPLDs an ideal platform to implement high-performance arithmetic circuits in a cost-effective manner.

FPGAs inherently have more usable gates than CPLDs and also provide a very fine grain architecture. The major constraints to deal with FPGAs are I/O utilization, logic utilization, and timing. A particular design can be literally placed in many different places in an FPGA because of its fine grain architecture. In CPLDs the structure is very coarse grained and this pushes the number of constraints higher. The typical constraints to deal with arithmetic designs in CPLDs are product term count, macrocell count, number of inputs into a logic block, product term and macrocell placement, number of passes through logic array, and sum-splits. All of these facts make designing arithmetic operations with CPLDs a tougher task. Understanding the structure and capabilities of CPLDs is absolutely essential in creating efficient designs.

With the background provided in this application note, a designer should be able to create any algorithm or implementation for an arithmetic application. The user is strongly encouraged to read the VHDL textbook written by the PLD applications group to get a good grasp of VHDL and using it to implement efficient designs in CPLDs and FPGAs.

FLASH370i and *Warp* are trademarks of Cypress Semiconductor Corporation.

CYPRESS

# The FLASH370i™ Family Of CPLDs and Designing with *Warp2*®

This application note covers the following topics: (1) a general discussion of complex programmable logic devices (CPLDs), (2) an overview of the FLASH370i™ family of CPLDs, and (3) using the *Warp2*® VHDL Compiler for the FLASH370i family.

## Overview of CPLDs

CPLDs extend the concept of the PLD to a higher level of integration to improve system performance, use less board space, improve reliability, and reduce cost. Instead of making the PLD bigger with more input terms and product terms, a CPLD architecture is composed of multiple PLDs or logic blocks (LABs) connected together with a programmable interconnect matrix (PIM). Multiple Logic Array Blocks (LABs) provide comparable speed to a PLD because the basic propagation path is through one LAB and each LABs product term array is comparable to a PLD array. Multiple LABs provide the higher integration. The number of LABs in a CPLD is typically between 2 for the smaller CPLDs and 16 for the larger ones. In addition to LABs interconnected by the PIM, are the input/output macrocells and the dedicated input macrocells. *Figures 1* and *2* show the CPLD generic block diagram and the logic block diagram respectively.

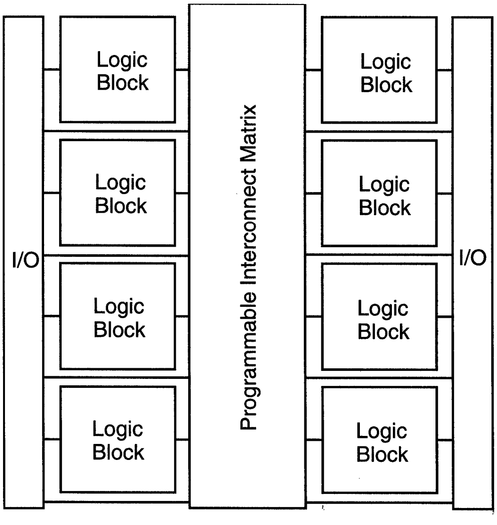


**Figure 1. Generic Block Diagram**

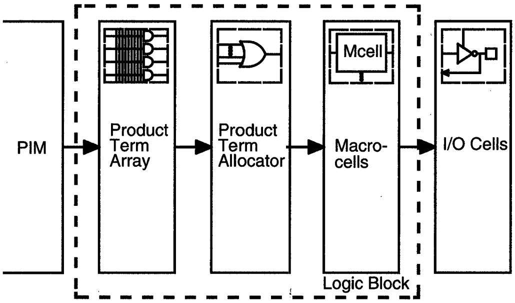


**Figure 2. Logic Block Diagram**

The architectural components of the LAB are: (1) the product term array, (2) the product term allocator, and (3) the macrocell. The product term array is the same in the CPLD as in the PLD except that the inputs into the array can now also come from the PIM. The product term allocator is a new concept in the CPLD where product terms are not fixed to a macrocell with its associated input/output pin but can be routed to different macrocells depending on where they are needed. The result is a more efficient allocation of product terms and higher integration. Implementation of the product term allocator varies across CPLD vendors which is more fully discussed in the section describing the features of the FLASH370i family.

The macrocell accepts the single output of the product term allocator which is the ORing of a variable number of product terms. In some macrocells this input feeds into a two input XOR gate with the other input potentially carrying the Q feedback. This configures the D flip flop to a T flip flop which can provide an improvement in capacity for certain designs such as counters. After the XOR gate, the macrocell is configurable as registered, combinatorial, and in some cases latched. There are two kinds of macrocells which are input/output dedicated and buried. Dedicated macrocells output to the input/output macrocell and also provide feedback into the product term array. Buried macrocells only provide feedback into the product term array.

The function of the PIM is to distribute the needed fraction of the total available resources, all outputs from the LAB and possibly also dedicated inputs and inputs/outputs, to the appropriate LAB. There are two common methods of PIM implementation: array based interconnect and mux based interconnect.

*Figure 3* shows the data path of communication between two LABs using the array based interconnect. In the array based interconnect, each output of the LAB can potentially connect to any number of PIM input terms through a memory element.

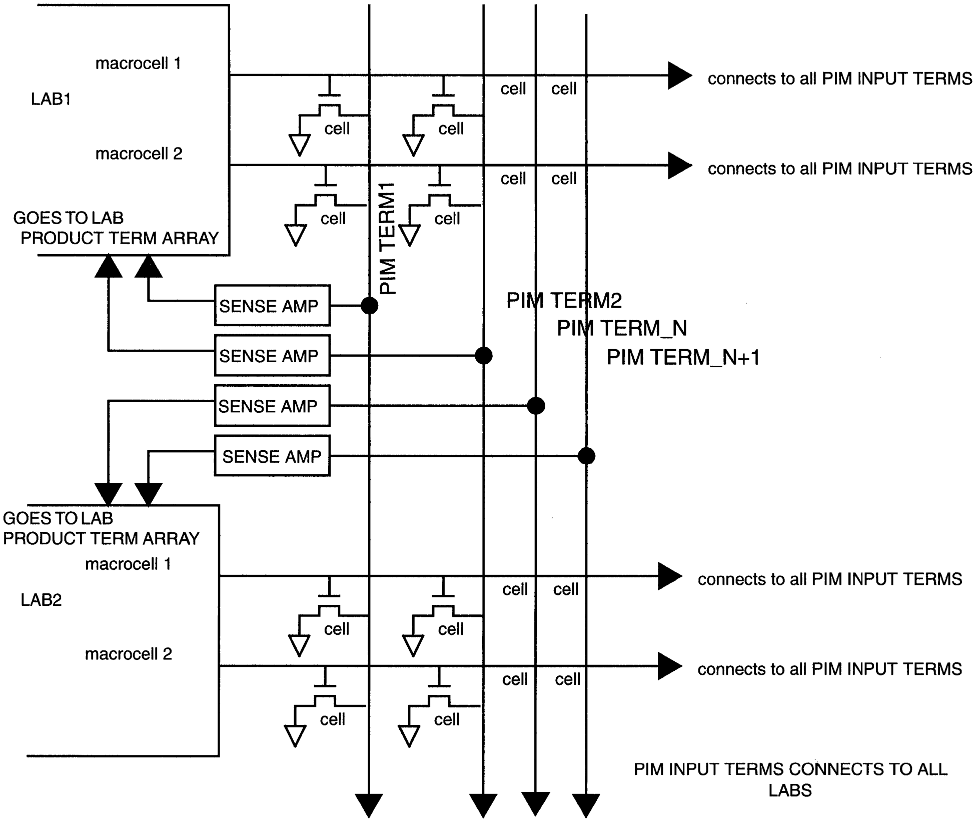


**Figure 3. Array-Based Interconnect**

Each PIM input term is assigned to a specific LAB and functions as an input term into the LABs product term array. In this example only four PIM input terms are shown two going to LAB1 and two going to LAB2. There is a sense amp per input term to detect the logic level, buffer the signal, and drive it into the LAB. The true and complement of the PIM signal feed into the product term array (not shown in the figure). Since every LAB output can connect to any PIM input, the interconnect is considered 100 percent routable. It never limits the ability of the device to fit logic. A macrocell output can connect to one or multiple PIM input terms. The major drawback from using a memory element as an interconnect is the slower propagation delay than the muxed based interconnect.

*Figure 4* shows the data path of communication between two LABs using the muxed based interconnect. In the muxed based interconnect a mux chooses one of a number of potential PIM input terms into the LAB. The PIM input terms differ from the array based interconnect in that they are output from a 1 of n (where "n" is the number of inputs of the mux) mux instead of the output of a wired nor memory array. The inputs into the muxes are all the outputs of the LABs as well as dedicated inputs and input/output pins. *Figure 3* shows two PIM input terms output from two 4-to-1 muxes. In this example, macrocell 2 from LAB1 and macrocell 2 from LAB2 both show 2 chances to route into the muxes with other inputs having only 1 chance. The wider the mux (the number of inputs into the mux) the more likely all desired inputs into each LAB will be successfully routed and the more chances each signal gets to route into a LAB. The disadvantage of larger muxes is a larger slower propagation delay through the PIM and increased die size. Implementations of mux-based interconnect vary in the size of the mux.

## Features of the FLASH370i CPLDs

The FLASH370™ family of CPLDs offers densities from 2 to 8 LABs. *Figure 5* shows the block diagram of the CY7C374i/5i with 8 LABs. The even numbers of the family (372i,374i) bury half of the macrocells for maximum integration with the same pinout as the (371i,373i,375i) respectively. *Table 1* shows the family members offered.

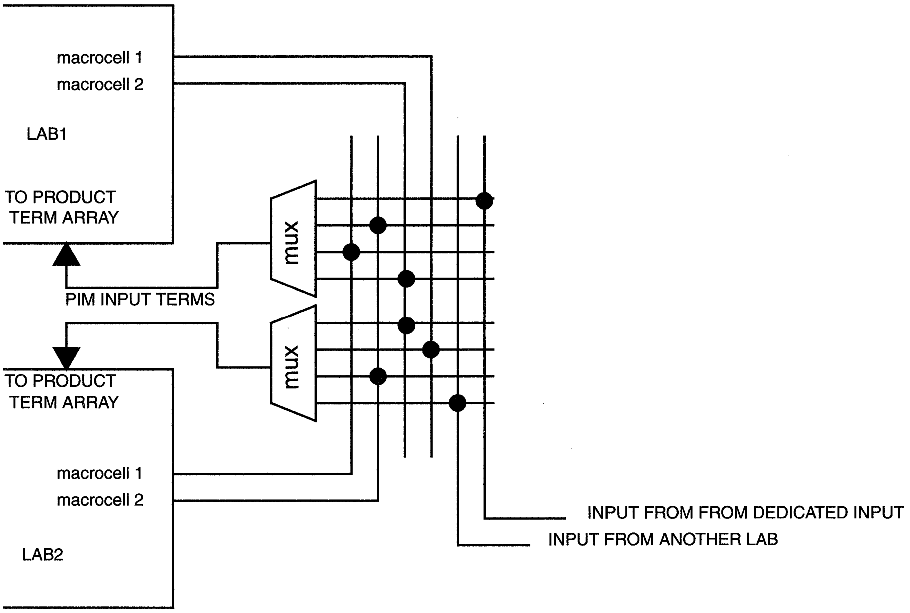


**Figure 4. Mux-Based Interconnect**

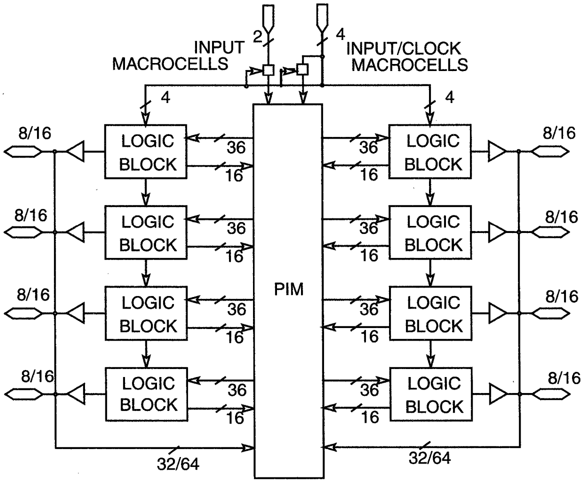


**Figure 5. CY7C374i/5i Block Diagram**

**Table 1. FLASH370i Family Members.**

| Feature | CY7C371i | CY7C372i/3i | CY7C374i/5i |
|---|---|---|---|
| Macrocells | 32 | 64 | 128 |
| Dedicated Inputs | 6 | 6 | 6 |
| I/O pins | 32 | 32/64 | 64/128 |
| Dedicated Inputs Usable as Clocks | 2 | 2/4 | 4/4 |
| Speed ($t_{PD}$) | 8.5 ns | 10 ns | 12 ns |
| Primary Packages | 44–PLCC | 44/84–PLCC<br>100–TQFP | 84–PLCC<br>100/160–TQFP |

*Figures 6* and 7 show the product term array, product term allocator, macrocells, and input/output macrocells for the FLASH370i family. Each LAB features 36 inputs, which can adequately handle 32-bit operations plus control signals with one pass through the LAB. The product term array features the true and complement polarities of each PIM output signal for a total of 72 inputs. 80 standard product terms are provided to the product term allocator which allocates from 0 to 16 product terms to each of the 16 macrocells. Additionally, 6 special product terms are also generated in the product term array. They are an asynchronous preset, asynchronous reset, and two groups of 2 bank output enable product terms.

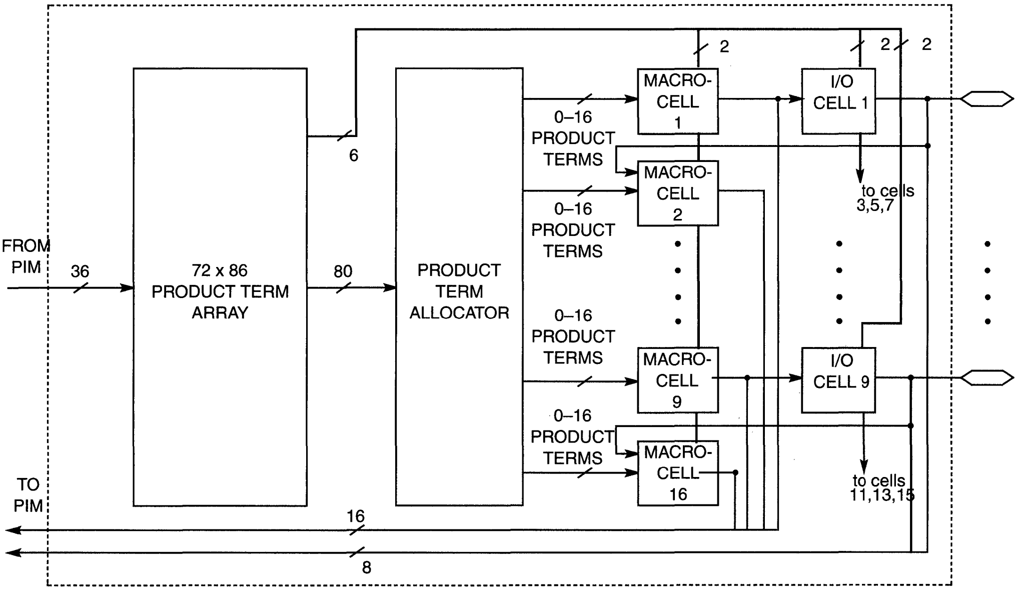


**Figure 6. Logic Block for CY7C372i and CY7C374i (Register Intensive)**

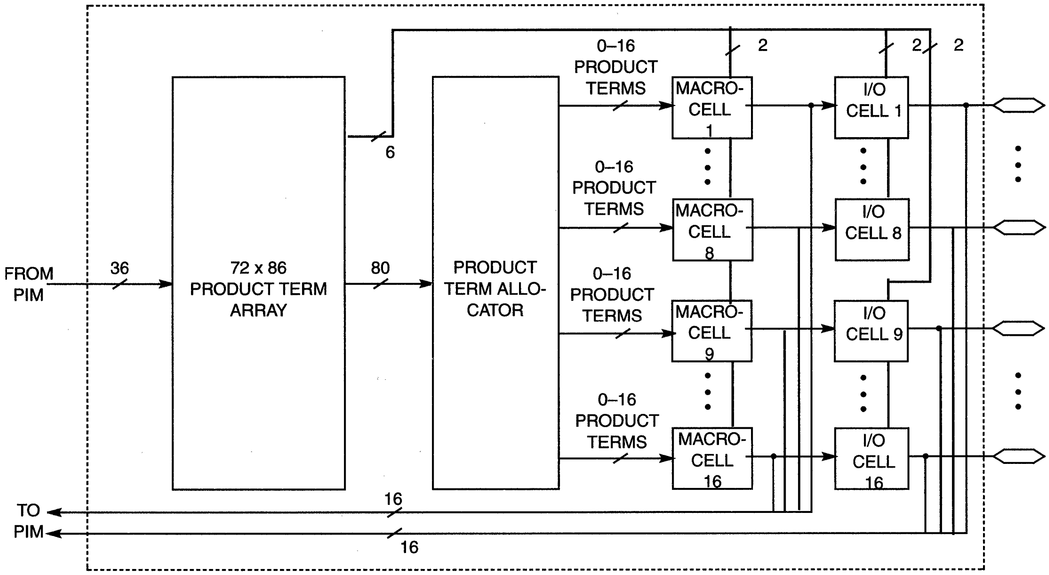


**Figure 7. Logic Block for CY7C371, CY7C373, CY7C375, and CY7C377 (I/O Intensive)**

The output macrocell (*Figure 8*) provides a selection of four output controlling options: (1) control from one output enable, (2) control from a second output enable, (3) permanently enabled, or (4) permanently disabled. Each LAB contains 4 output enable product terms, 2 for the upper 8 macrocells and 2 for the lower 8 macrocells.

The state macrocell (*Figure 8*) contains options to register, latch, or send data through combinatorially. For the input/output macrocell there is an additional output polarity mux to improve capacity before the signal goes to the input/output macrocell. For buried macrocells there is an additional mux which can configure the state register as an input register. If the buried macrocell is configured as an input, zero product terms will be allocated from the array. In *Figure 8* architecture bit C7 can choose the feedback from the input/output pin as the input into the register instead of from the product term array.

There is one asynchronous preset and reset product term for each LAB. There are polarity muxes for the clocks, preset and reset. Each macrocell can choose among two clocking options for the CY7C371i/372i and four clocking options for the CY7C373i/374i/375i. All macrocells in a LAB receive the same polarity of the clock, set and reset. Polarities are configurable per LAB. *Figure 8* shows the input/output macrocell and input/output plus buried macrocell.

*Figures 9* and *10* show the input/clock and input macrocells. The input macrocell provides the flexibility to let the input enter combinatorially, latched, single registered, or double registered (for maximum metastability performance). For the CY7C371i/372i there are two input/clocks pins and four input pins. For the CY7C373i/374i/375i there are four input/clock pins and two input pins. For added flexibility, each clock can be configurable for either positive or negative polarity.

In order to fully understand the operation of the FLASH370i product term allocator, two important aspects of product term allocator design need to be introduced: product term steering and product term sharing. Steering refers to the assignment of a product term resource to a macrocell. In the traditional PLD there is no steering flexibility. Each macrocell has assigned product terms that can only be used by that macrocell. In many designs each macrocell requires a different number of product terms putting an emphasis on the ability to allocate product terms individually on an as needed basis. Product term sharing refers to a product term being used by multiple macrocells. The logic equations for different macrocells sometimes contain the same minterm. Instead of generating this same minterm multiple times, it is generated on only one product term and shared across macrocells, thereby improving capacity.

*Figure 11* is a conceptual representation of the FLASH370i product term allocator. The product term allocator functions like a segmented OR array by ORing from 0 to 16 product terms for each macrocell. Product terms can be steered and shared on an individual basis. This architecture has several advantages over other implementations that steer product terms away from one macrocell to serve another.

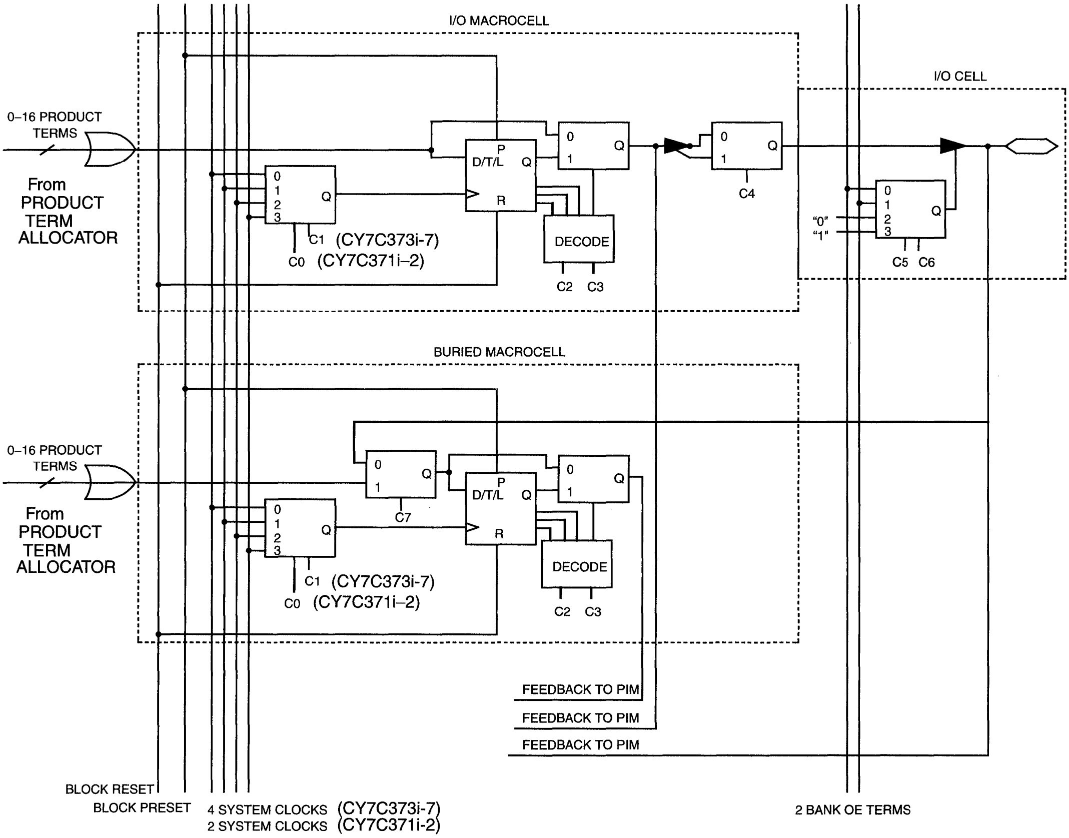


**Figure 8. I/O and Buried Macrocells**

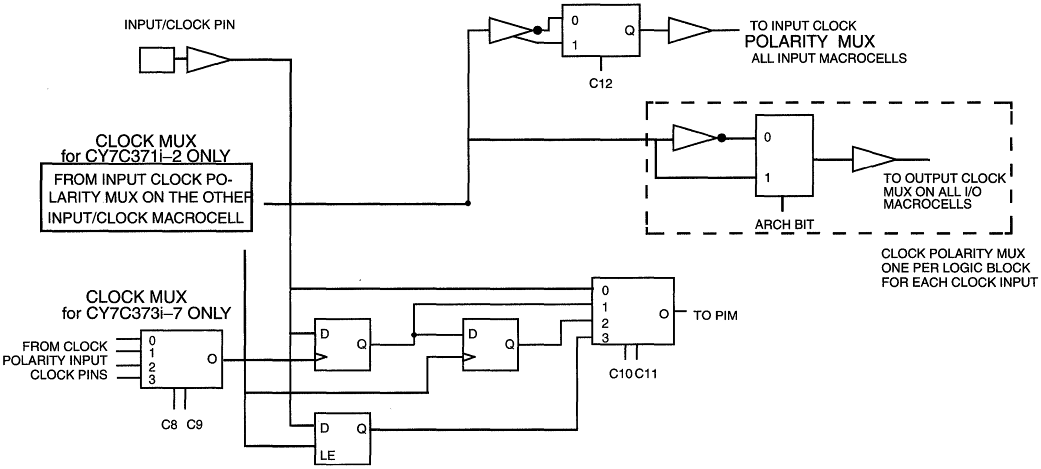

INPUT/CLOCK PIN
TO INPUT CLOCK
POLARITY MUX
ALL INPUT MACROCELLS
C12
CLOCK MUX
for CY7C371i–2 ONLY
FROM INPUT CLOCK POLARITY MUX ON THE OTHER INPUT/CLOCK MACROCELL
TO OUTPUT CLOCK MUX ON ALL I/O MACROCELLS
ARCH BIT
CLOCK POLARITY MUX ONE PER LOGIC BLOCK FOR EACH CLOCK INPUT
CLOCK MUX
for CY7C373i–7 ONLY
FROM CLOCK POLARITY INPUT CLOCK PINS
C8 C9
TO PIM
C10 C11
D
Q
LE

**Figure 9. Input/Clock Pins**

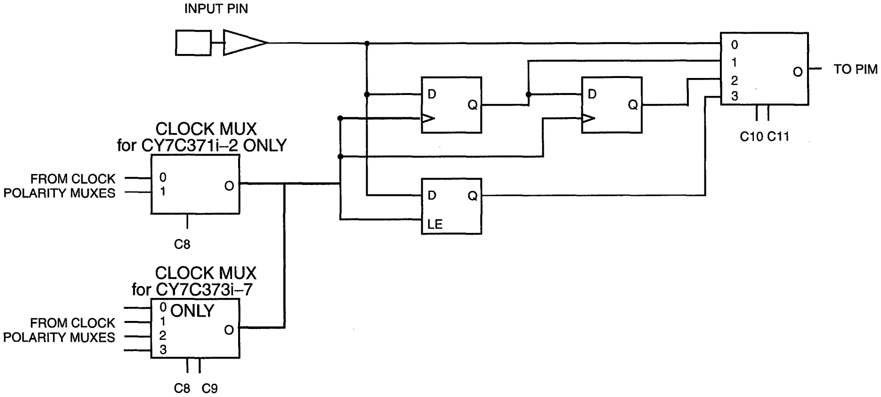


**Figure 10. Input Pins**

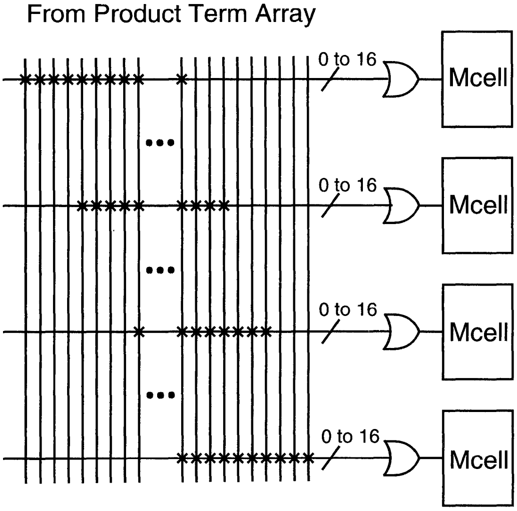


**Figure 11. FLASH370i Product Term Allocator Representation**

*Figure 12* is a conceptual representation of the MACH™ product term allocator. It shows no ability to share product terms across macrocells. Each cluster of four product terms can route to only one macrocell. The product terms are routed in groups of four which is a much higher granularity of product term allocation and not as efficient.

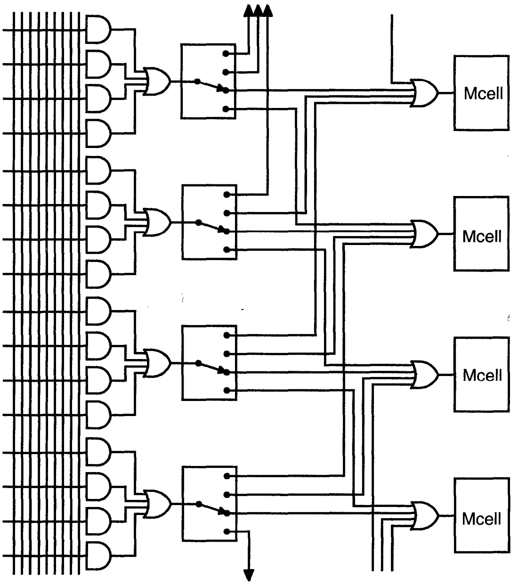


**Figure 12. MACH Product Term Allocator Representation**

To demonstrate this inefficiency, consider a macrocell that needs five product terms to implement its logic. Two product term clusters with a total of eight available product terms are needed. This wastes the resources of three product terms from the borrowed cluster since these product terms can not be rerouted to another macrocell.

The MAX7000™ product term allocator representation (*Figure 13*) shows the use of expander terms. Expander terms allow two passes through the array which can produce very high capacity. These expanders are also shared among all product terms in the LAB. The problem with using the expanders is in the additional propagation delay of two passes through the array. This complicates the timing model and links the performance of the device to the use of the expander product terms. As with the MACH product term allocator, the MAX7000 allocator also has five product term clusters. It therefore suffers from the same problem of product term wasting when more than one cluster is routed to a macrocell.

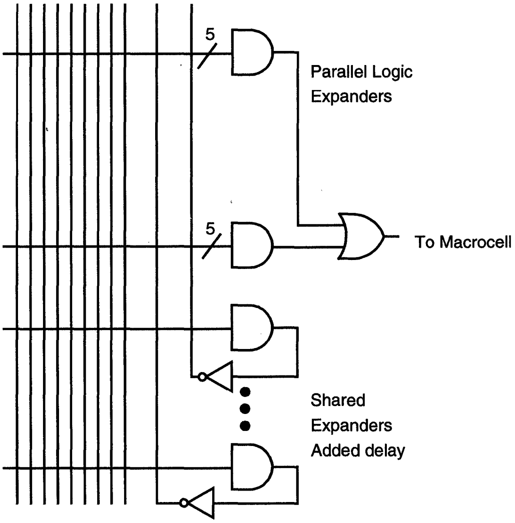


**Figure 13. MAX Product Term Allocator Representation**

The FLASH370i product term allocator provides the most effective method of steering and sharing product terms. The propagation of signals through the product term allocator is independent of the number of product terms allocated to each macrocell. Additionally the flexibility of this product term allocator, with the PIM, enables a change in the design without a modification to the external pinout of the device. There is no need for input and output switch matrices, which add extra delay and degrade performance.

The timing model of the FLASH370i family is far simpler than for other CPLD solutions for two reasons. First, all input signals into the LAB pass through the PIM. This includes all input/outputs, feedbacks from macrocell outputs, and dedicated inputs. Secondly, the propagation time through the product term allocator is independent of the number of product terms allocated to a macrocell. As a result, there are no expander delays, no dedicated versus input/output pin delays, no penalties for using up to 16 product terms, or no delay penalties for steering and or sharing product terms. The FLASH370i family of products provides timing as predictable as PLDs like the 22V10.

The PIM in the FLASH370i was designed to approach the 100 percent routability of the array based interconnect but not made so wide that performance and die size suffered.

## Using *Warp*™ to Design with the FLASH370i

Development software is extremely important for ease of use and efficiency of resource allocation when designing with CPLDs. Cypress offers two software packages that will fully support the FLASH370i family of products as well as all other PLDs and state machine PROMs. *Warp2* provides full VHDL language support which is becoming the industry standard for describing hardware design. A functional simulator is also provided. *Warp3*® additionally includes schematic capture and exact timing simulation capability.

The simplified timing model of the FLASH370i often makes exact timing simulation unnecessary because performance can be predicted directly from the datasheet. Therefore the functional simulator of *Warp2* may be a cost effective design solution. With *Warp* no manual intervention for fitting the designs into the devices are necessary. In addition to *Warp*, customers also have third party support from a variety of vendors.

*Warp* products take in VHDL designs and automatically fit them into the chosen device. The following section explains how to exploit the special features of the FLASH370i with VHDL. A thorough treatment of VHDL constructs is found in the *Warp2* Reference Manual. Topics covered here are: (1) using the single/double registered options for the dedicated inputs, and registering signals from the IO pins, (2) using the clock polarity mux feature, (3) describing registered versus latched versus combinatorial outputs, (4) using the output enable feature, (5) using the asynchronous preset/reset feature, and (6) Using the buried registers as for the (372/4/6).

To register the dedicated inputs one or two signals must be defined to represent the additional nodes for one and two registers respectively. Appendix A demonstrates how to use single and double registered inputs for a 4 bit loadable counter. In *proc2*, RESET1 and RESET2 are the outputs of the first and second registers. It requires 2 passes through *proc2* to activate RESET2. Signal RESET2 is then used in *proc1* to perform the reset. *Proc2* additionally registers the data to be loaded with the statement `regin <=temp.dat.` The signal REGIN is then used in process *Proc1* to load the counter with the statement `temp.cnt <= regin.` If the same clock is used for the inputs as for the state registers, then the statements in process *proc2* could be incorporated into *proc1* and only one process is needed. The assignment of the entity output pins is handled by the instantiation of the *bufoe* component (called in the statement `use work.rtlpkg.all`), which takes the signal TEMP.CNT as input and transfers it to the output (in this case called COUNT) when the output enable control (called OUTEN) is HIGH. Registering the inputs from the input/output pins is better suited for the 372/374/376 members of the family since the signal does not need to go through the PIM and logic block.

Clocking on the falling instead of the rising edge of the clock is simply done by changing the statement `wait until (clk = '1')` to `wait until (clk = '0')`. Events occurring on the rising and falling edge of a clock can be incorporated into the same design by defining a separate process for the event, provided that sufficient logic blocks are available.

VHDL describing combinatorial and registered outputs is identical to other part implementations as with the FLASH370i. The registered equations must be inserted inside a process and after a `wait until clock=` statement.

Appendix B shows an example of how to implement the combinatorial macrocell option with maximum usage of output enable flexibility for the CY7C371. A total of eight different input signals control the output enable functionality. The entire function is handled by the *bufoe* component where the input into the buffer is the external input pin. No signals are necessary.

The latch option is unique to the FLASH370i family. Appendix C shows an example of how to latch a signal using the IF-THEN-ELSE construct. In this example the signal is latched when the clock is HIGH by setting the signal value to itself with the statements `signala <= signala` and `signalb <= signalb`. When the clock is LOW the path is combinatorial and the signal value gets the input. This is handled in the code `if clk='0' then signala <= inputa; signalb <= inputb`. Two signals are defined, SIGNALA and SIGNALB, to latch the data when the clock is in the right polarity (in this case HIGH).

Appendix D shows the full registered configuration. As in Appendix C, the signals SIGNALA and SIGNALB are defined and the function of the register is defined within a process. On the rising edge of the clock, SIGNALA gets INPUTA and SIGNALB gets INPUTB.

Appendix E uses latches for the output enable control. Signals need to be generated from the array and are passed as the output enable parameter into the *triout* component. This function behaves similarly to the *bufoe* but does not include the feedback parameter.

Appendix F shows how to use the buried registers to implement the least significant bits in a counter. A bit vector signal is defined to represent all the register states. Those states that are needed as outputs are assigned to the entity output pins outside of the process with the statement `count (0 to 11) <= fullcnt (4 to 15)`. If output enable control is desired then this last statement is omitted and the signal to output assignment is handled with the *bufoe* component.

Appendix G is the same as Appendix F except that the registers are reset asynchronously. The format of the process is much different from Appendix F but functions exactly the same except for the asynchronous instead of synchronous reset. The process uses a "sensitivity list" that includes all the parameters that will activate the process. The synchronous part of the process is initiated by the statement `clk'event and clk='1'` instead of `wait until clk='1'`. The asynchronous preset/reset is similar to other Cypress PLDs except for the additional polarity mux feature that enables active HIGH or LOW. To specify clock polarity, the VHDL construct for active HIGH is `if reset = '1' then` and for active LOW is `if reset = '0' then`.

## Appendix A. inregcnt

```
-- The bufoe port map parameters are:
-- bufoe port map(signal going to the input of the tristateable buffer,
--     tristate control signal,
--     the output signal that is the entity output pin,
--     the feedback signal from the entity input/output pin)
-- In this example the last entry is "open" meaning no feedback.

USE work.bv_math.all;          -- necessary for inc_bv();
USE work.rtlpkg.all;         -- necessary for bufoe

ENTITY inregcnt IS
       PORT   (clk, clkin, reset, load, outen: IN bit;
         count: INOUT x01z_VECTOR(0 TO 3));
END inregcnt;

ARCHITECTURE behavior OF inregcnt IS
TYPE bufRec IS                 -- record for bufoe
       RECORD                  -- inputs and feedback
            cnt: bit_vector(0 TO 3);
            dat: bit_vector(0 TO 3);
END RECORD;
SIGNAL temp: bufRec;
SIGNAL regin: bit_vector(0 to 3);-- for registering input loaded data
SIGNAL reset1, reset2:bit;       -- for registering the reset input
CONSTANT counterSize: integer := 3;
BEGIN
g1:    FOR i IN 0 TO counterSize GENERATE
            bx: bufoe PORT MAP(temp.cnt(i), outen, count(i), temp.dat(i));
       END GENERATE;
proc1: PROCESS
       BEGIN
            WAIT UNTIL (clk = '1');
            IF reset2 = '1' THEN -- uses the double registered signal
                temp.cnt <= "0000";
            ELSIF load = '1' THEN
                temp.cnt <= regin; -- uses the single registered signal
            ELSE
                temp.cnt <= inc_bv(temp.cnt);  -- increment bit vector
            END IF;
       END PROCESS;
-- Proc2 single registers the load operation and double registers the reset -- oper-
ation.  Note the two clkin's are needed for the double register.
proc2: PROCESS
       BEGIN
         WAIT UNTIL (clkin = '1');
          regin <= temp.dat; --single register for data load
          reset1 <= reset; --single register the reset signal
          reset2 <= reset1;--double register the reset signal
       END PROCESS;
END behavior;
```

## Appendix B. usecomb

```
--uses the full functionality of the oe features of the 371.
--macrocell is in combinatorial mode

USE work.rtlpkg.all;

ENTITY usecomb IS
       PORT (outen1, outen2, outen3, outen4, outen5, outen6, outen7,
            outen8; IN bit; inputa, inputb: IN bit_vector(0 to 1);
            outa,outb: INOUT x01z_vector(0 to 7));
END usecomb;

ARCHITECTURE behavior OF usecomb IS
BEGIN
g1:    FOR i IN 0 TO 1 GENERATE
            bx1: bufoe PORT MAP(inputa(i), outen1, outa(i), open);
            bx2: bufoe PORT MAP(inputa(i), outen2, outa(i+2), open);
            bx3: bufoe PORT MAP(inputa(i), outen3, outa(i+4), open);
            bx4: bufoe PORT MAP(inputa(i), outen4, outa(i+6), open);
            bx5: bufoe PORT MAP(inputb(i), outen5, outb(i), open);
            bx6: bufoe PORT MAP(inputb(i), outen6, outb(i+2), open);
            bx7: bufoe PORT MAP(inputb(i), outen7, outb(i+4), open);
            bx8: bufoe PORT MAP(inputb(i), outen8, outb(i+6), open);
       END GENERATE;
END behavior;
```

## Appendix C. uselatch

```
--uses the full functionality of the oe features of the 371.
--macrocell in latched mode

USE work.rtlpkg.all;

ENTITY uselatch IS
       PORT   (clk, outen1, outen2, outen3, outen4, outen5, outen6, outen7,
       outen8: IN bit;
            inputa, inputb: IN bit_vector(0 to 1);
            outa,outb: INOUT x01z_vector(0 to 7));
END uselatch;

ARCHITECTURE behavior OF uselatch IS
SIGNAL signala, signalb: bit_vector(0 to 1);
BEGIN
g1:    FOR i IN 0 TO 1 GENERATE
            bx1: bufoe PORT MAP(signala(i), outen1, outa(i), open);
            bx2: bufoe PORT MAP(signala(i), outen2, outa(i+2), open);
            bx3: bufoe PORT MAP(signala(i), outen3, outa(i+4), open);
            bx4: bufoe PORT MAP(signala(i), outen4, outa(i+6), open);
            bx5: bufoe PORT MAP(signalb(i), outen5, outb(i), open);
            bx6: bufoe PORT MAP(signalb(i), outen6, outb(i+2), open);
            bx7: bufoe PORT MAP(signalb(i), outen7, outb(i+4), open);
            bx8: bufoe PORT MAP(signalb(i), outen8, outb(i+6), open);
       END GENERATE;--the clk input is an active low latch enable
--the if then construct must be within a process.
PROCESS
  BEGIN
     IF clk='0' then
       signala <= inputa;
       signalb <= inputb;
     ELSE
       signala <= signala;
       signalb <= signalb;
     END IF;
  END PROCESS;
END behavior;
```

## Appendix D. usereg

```
--macrocell in registered mode

ENTITY usereg IS
       PORT   (clk, outen1, outen2, outen3, outen4, outen5, outen6, outen7,
       outen8: IN bit; inputa, inputb: IN bit_vector(0 to 1);
            outa,outb: INOUT x01z_vector(0 to 7));
END usereg;

ARCHITECTURE behavior OF usereg IS
SIGNAL signala, signalb: bit_vector(0 to 1);
BEGIN
g1:    FOR i IN 0 TO 1 GENERATE
            bx1: bufoe PORT MAP(signala(i), outen1, outa(i), open);
            bx2: bufoe PORT MAP(signala(i), outen2, outa(i+2), open);
            bx3: bufoe PORT MAP(signala(i), outen3, outa(i+4), open);
            bx4: bufoe PORT MAP(signala(i), outen4, outa(i+6), open);
            bx5: bufoe PORT MAP(signalb(i), outen5, outb(i), open);
            bx6: bufoe PORT MAP(signalb(i), outen6, outb(i+2), open);
            bx7: bufoe PORT MAP(signalb(i), outen7, outb(i+4), open);
            bx8: bufoe PORT MAP(signalb(i), outen8, outb(i+6), open);
       END GENERATE;    --the clk input is a rising edge triggered clock for
                        --the register
--the wait until construct must be within a process.
PROCESS
  BEGIN
    WAIT UNTIL clk='1';
    signala <= inputa;
    signalb <= inputb;
  END PROCESS;
END behavior;
```

## Appendix E. uselatch2

```
--This file shows the use of the triout component to perform the
--output enable function.

--COMPONENT triout
-- port (
--   x: IN bit; -- input to buffer
--   oe: IN bit; -- output enable
--   y: OUT bit); -- output
--END component

--The oe control is a function of the dedicated inputs and is latch
--controlled.

USE work.rtlpkg.all;        --to instantiate triout component

ENTITY uselatch2 IS
  PORT (clk1, clk2, in_oe1, in_oe2: IN bit;
        inputa, inputb: IN bit_vector(0 to 1);
         outa,outb: INOUT x01z_vector(0 to 7));
END uselatch2;

ARCHITECTURE behavior OF uselatch2 IS
SIGNAL signala, signalb: bit_vector(0 to 1);
SIGNAL sig_en1, sig_en2, sig_en3, sig_en4: bit;
BEGIN
g1:  FOR i IN 0 TO 1 GENERATE
       bx1: triout PORT MAP(signala(i), sig_en1, outa(i));
       bx2: triout PORT MAP(signala(i), sig_en2, outa(i+2));
       bx3: triout PORT MAP(signala(i), sig_en3, outa(i+4));
       bx4: triout PORT MAP(signala(i), sig_en4, outa(i+6));
       bx5: triout PORT MAP(signalb(i), sig_en1, outa(i));
       bx6: triout PORT MAP(signalb(i), sig_en2, outa(i+2));
       bx7: triout PORT MAP(signalb(i), sig_en3, outa(i+4));
       bx8: triout PORT MAP(signalb(i), sig_en4, outa(i+6));
     END GENERATE;

--The clock latches the data when high and is combinatorial when low
oecontrol: PROCESS
            BEGIN
              IF clk1= '0' then
                sig_en1 <= not(in_oe2) and not(in_oe1);
                sig_en2 <= not(in_oe2) and in_oe1;
                sig_en3 <= in_oe2 and not(in_oe1);
                sig_en4 <= in_oe2 and in_oe1;
              ELSE
                sig_en1 <= sig_en1;
                sig_en2 <= sig_en2;
                sig_en3 <= sig_en3;
                sig_en4 <= sig_en4;
              END IF;
            END PROCESS;

latch: PROCESS
```

## Appendix E. uselatch2 (continued)

```
            BEGIN
              IF clk2= '0' then
                signala <= inputa;
                signalb <= inputb;
              ELSE
                signala <= signala;
                signalb <= signalb;
              END IF;
            END PROCESS;
END behavior;
```

## Appendix F. buriedreg

```
-- The purpose of this example is to show how to use the
-- buried registers to create a 16 bit counter.  The 12
-- most significant bits are assigned to i/o registers
-- and the 4 least significant bits go to the buried registers.

USE work.bv_math.all;               -- necessary for inc_bv();

ENTITY buriedreg IS
       PORT   (clk, reset: IN BIT;
            count: INOUT bit_vector(0 TO 11));
END buriedreg;

ARCHITECTURE behavior OF buriedreg IS
SIGNAL fullcnt : bit_vector(0 to 15);
BEGIN
PROCESS
     BEGIN
       WAIT UNTIL (clk = '1');
            IF reset = '1' THEN       -- synchronous reset
                 FOR i IN 0 TO 15 LOOP
                      fullcnt(i) <= '0';
                 END LOOP;
            ELSE
                 fullcnt <= inc_bv(fullcnt);
            END IF;
     END PROCESS;
count(0 to 11) <= fullcnt(4 to 15);
END behavior;
```

## Appendix G. buriedreg2

```
-- The purpose of this example is to show how to use the
-- buried registers to create a 16 bit counter.  The 12
-- most significant bits are assigned to i/o registers
-- and the 4 least significant bits go to the buried registers.
-- This example also demonstrates how to do an asynchronous reset.

USE work.bv_math.all;               -- necessary for inc_bv();

ENTITY buriedreg2 IS
  PORT   (clk, reset: IN BIT;
     count: inout bit_vector(0 TO 11));
END buriedreg2;

ARCHITECTURE behavior OF buriedreg2 IS
SIGNAL fullcnt : bit_vector(0 to 15);
BEGIN
   PROCESS(clk,reset)--sensitivity list
          BEGIN
            IF reset = '1' THEN
              fullcnt <= x"0000";-- asychronous reset, the x stands for hex
            ELSIF (clk'event and clk = '1') then
              fullcnt <= inc_bv(fullcnt);-- synchronous count
            END IF;
          END process;
   count(0 to 11) <= fullcnt(4 to 15);      -- assignssignals to entityoutputs
  -- and defines buried registers
END behavior;
```

MAX7000 is a trademark of Altera Corporation.
MACH is a trademark of Advanced Micro Devices, Inc.
*Warp2* and *Warp3* are registered trademarks of Cypress Semiconductor Corporation.
*Warp* and FLASH370i are a trademarks of Cypress Semiconductor Corporation.

# Implementing a Reframe Controller for the CY7B933 HOTLink™ Receiver in a CY7C371 CPLD

## Introduction

This application note describes a reframe controller for the Cypress CY7B933 HOTLink™ Receiver. The primary function of the controller is to monitor the Receive Violation Symbol output, RVS, from the CY7B933 in order to detect framing errors and, under the correct conditions, assert the Reframe signal, RF, to the CY7B933. The controller function is designed with a state machine, a few counters, and some decode logic. All are implemented in VHDL and fit into a Cypress CY7C371 32-macrocell Flash CPLD. The exact implementation in this application note makes several assumptions about the next-higher-level controller that may not be universally applicable. However, the source code for the design is provided in Appendix A at the end of this application note so that modification and customization for other interfaces is easily possible.

## Why Reframing is Necessary

The CY7B923 and CY7B933 HOTLink Transmitter and Receiver are a pair of chips for high-speed point-to-point serial data communication. The CY7B923 is the transmitter, and the CY7B933 is the receiver. The CY7B923 takes in an 8-bit byte at a frequency between 16 and 33 MHz, encodes it into 10 bits, does a parallel-to-serial conversion, and then transmits the serial data at ten times the byte-rate clock (about 160 to 330 Megabits per second (Mbps)). At the other end of the link, the CY7B933 receives the serial data, does a serial-to-parallel conversion, unencodes the data back into its original form, and shifts the 8-bit parallel data out at the same byte-rate clock frequency used by the transmitter. (Note: the chips can also transmit and receive 10 bits of unencoded data. For a full description of the encoding and decoding functions, see the CY7B923/933 datasheet.)

The key element in the data-and-clock-recovery circuit on the receiver is the PLL, i.e., phase-locked loop, on the chip. It is triggered by the transitions in the incoming data stream, and it is used to both separate the data stream into individual bits and to generate the byte-rate clock going out of the chip. Once the PLL achieves synchronization with the incoming serial data stream and is receiving bits properly, the receiver must be given a reference point that will set the byte boundaries in the bit stream. This is done by the framing circuitry. Whenever the receiver's RF (reframe) input is asserted, the receiver's framing logic will check the incoming bit stream for the special pattern that defines a byte boundary. When this is found, the receiver logic sets a reference point and simply counts bits from that point on so it can properly execute the serial-to-parallel conversion on subsequent byte boundaries, and properly align the byte-rate clock rising edge.

Thus, framing is always required when the receiver begins receiving data for the first time, either at power-up or after switching from one transmitter source to another. Periodic reframing may also be necessary due to other conditions. If the PLL goes out of lock-that is, if it loses its synchronization with the incoming serial bit stream for any reason—the recovered data will be erroneous and the framing boundary information will be lost. Once the PLL gets back into synchronization with the incoming bit stream, it will be necessary to force the receiver to reframe in order to re-establish the proper byte boundary point.

## Using RVS to Know When to Reframe

The PLL out-of-lock condition can be detected by the behavior of the RVS output of the CY7B933 receiver. The CY7B933 asserts RVS when it detects an error in the bit stream. Infrequent errors, due to random noise in the environment or attenuation by the transmission medium, for example, are expected and do not necessarily mean that the PLL is out of lock or that the data needs to be reframed. Too many errors in too short a time indicates that the PLL has lost lock and reframing is necessary. The benchmark chosen in this controller is 16 errors occurring in a period of 64 bytes. If the controller counts RVS asserted 16 times during a 64-byte period, it will assume the PLL has lost lock and will assert RF to the receiver to force it to reframe.

The 16-out-of-64 benchmark is somewhat arbitrarily chosen, but it is justified by the fact that when the PLL is in lock, you would normally expect to see significantly fewer errors. The fact that 16 out of 64 is the criteria used does not mean that 15 out of 64, or 14 out of 64, etc., are acceptable error rates and that the PLL is not out of lock in these cases as well. But, it is fairly certain that if the PLL does go out of lock, you will get at least 16 errors in 64 byte-times, very quickly. Furthermore, there are counters inside the HOTLink Receiver that detect this same condition (16 errors in a 64-byte period) and when this detection occurs inside the CY7B933, it forces the PLL to re-lock onto the serial input data stream. Even if the PLL *is* out of lock, if fewer than 16 errors are detected in a 64-byte period, the PLL will not be forced to re-synchronize with the data stream and will stay out-of-lock until that condition is detected. Therefore, for consistency, the same criteria was selected for the reframe controller.

## Additional Functionality of the Reframe Controller

The reframe controller itself interfaces to a higher-level controller that controls the entire receiver system. That higher-level controller can force the reframe controller to initiate framing in the CY7B933, regardless of any errors. There are two ways to do this. The first is with the DO_REFRAME signal, which the higher-level controller asserts when it wants the reframe controller to go through the same procedure it goes through to initiate framing when an out-of-lock condition occurs. If the reframe controller sees this signal asserted, it acts just like it had detected an out-of-lock condition. The other way the higher-level controller can force a reframe is by asserting its FORCE_RF output. This simply forces the reframe controller's RF output HIGH and does not cause the internal logic or state machine to change. The reframe controller's RF output will stay asserted as long as its FORCE_RF input remains asserted.

The higher-level controller will normally assert DO_REFRAME on power-up or when the transmitter source is switched on in order to find the initial byte-boundary, as described above. The FORCE_RF signal could be used for any reason depending on specific system requirements. The most likely reason to use it is to force multibyte framing. When the receiver does multibyte framing, instead of looking for a single byte-boundary-indicating character, the receiver looks to detect two of these special characters within any four-byte sequence. This is a more reliable way of finding the byte boundary, simply because it causes the framing circuitry to verify its first find with another one. This may be useful in particularly noisy environments. To cause the receiver to do multibyte framing, you must assert its RF input for 2048 consecutive cycles; this is something the reframe controller would not ordinarily do. The higher-level controller can cause this to happen by asserting FORCE_RF to the reframe controller for 2048 cycles, thus causing its RF output to be asserted for the same length of time.

The reframe controller also implements a basic handshake with the higher-level controller to make sure the two controllers' operations stay consistent after forced reframes. Whenever the higher-level controller uses the DO_REFRAME signal to force the reframe controller to initiate framing, it will keep that signal asserted until the reframe controller asserts RFDONE_HS. This signal from the reframe controller indicates that the receiver has finished its reframing. The higher-level controller will then assert RFDONE_ACK, which acknowledges receipt of RFDONE_HS, and both the reframe controller and the higher-level controller will return to the state it normally returns to following a reframe.

In addition to the operations described above, the reframe controller also provides a decoding function. When the HOTLink Receiver detects a data error and asserts RVS, it also puts the code for the type of error on its eight data outputs, D7 - D0. The reframe controller decodes these signals and asserts one of two outputs, UNDEF_CHAR or RDISP_ERR, depending on the exact type of error decoded. The two types of errors are an undefined-character error and a running-disparity error. A running-disparity error means that the character received had too many consecutive 1s or 0s to be a valid byte of data (the purpose of the eight-bit-to-ten-bit encoding mentioned earlier is to encode the data in such a way as to minimize the imbalance of 1s and 0s in the bit stream). If the reframe controller detects the code for a running-disparity error, it will assert the RDISP_ERR output. If the received character has the correct running disparity but is not a valid code for any character, then it is an undefined-character error, and the reframe controller will assert the UNDEF_CHAR output instead.

## Design and Implementation

The out-of-lock detection, RF control, higher-level controller interface, and error-type decoding are implemented with a simple state machine, a few internal counters, and some decoding logic, and it is all fit into a 32-macrocell CY7C371 Flash CPLD (for more information on this CPLD, please refer to other application notes in the PLD section of this handbook and to the CY7C371 datasheet). The design was done in VHDL and compiled with Cypress's *Warp*™ PLD design tool. The receiver system, the reframe controller's interface, and the details of the design of the internal state machine, counters, and logic are described in detail in the rest of this section.

### Receiver System

*Figure 1* shows where this reframe controller fits into the overall system. The CY7B933 receiver connects (through a physical connector) to the actual transmission medium, which can be either twisted pair, coaxial cable, or fiberoptic cable. The reframe controller interfaces to the receiver, and it also interfaces to the higher-level system controller.

### Controller Interface

The complete set of reframe controller inputs and outputs is shown in *Figure 2*, and their source or destination, polarity, and functionality are described below.

*Inputs*

**RF_ENABLE.** Overall enable. It comes from a higher-level controller. When asserted (HIGH), reframe controller is enabled. When deasserted, reframe controller is disabled and does not operate.

**CLK.** Clock signal to the reframe controller that comes from the recovered byte-rate-clock output, RCLK, of the CY7B933, and is also used in the rest of the system as the system clock.

**RESET.** Resets the state machine and the internal counters and status registers (HIGH = asserted).

**RVS.** Received Violation Symbol. It comes from RVS output of the CY7B933 (HIGH = asserted).

**FORCE_RF.** When asserted, this forces the RF output to also be asserted regardless of other conditions. It comes from a higher-level controller (HIGH = asserted).

**DO_REFRAME.** When asserted, it causes internal state machine to initiate framing in the receiver just as if it had detected an out-of-lock condition. It comes from higher-level controller (HIGH = asserted).

**RFDONE_ACK.** Handshake signal from the higher-level controller acknowledging that it received confirmation that the reframe controller completed the framing procedure. The handshake is only done when the framing was triggered by the DO_REFRAME signal, not by an out-of-lock condition (HIGH = asserted).

**$\overline{\text{RDY}}$.** Ready signal. It comes from the CY7B933 $\overline{\text{RDY}}$ output and indicates to the reframe controller that the receiver has completed the reframe operation (LOW = asserted).

**D[7:0], SC/$\overline{\text{D}}$.** Eight-bit data byte and control/data indicator bit from the CY7B933 receiver. The information on these lines can be decoded during a receive violation to determine the error type.

*Outputs*

**RF.** Reframe output. It goes to the RF input of the CY7B933 receiver and causes the HOTLink Receiver to begin a framing operation on the incoming data stream (HIGH = asserted).

**RFDONE_HS.** This is the handshake signal to the higher-level controller telling it that the reframe it requested with the

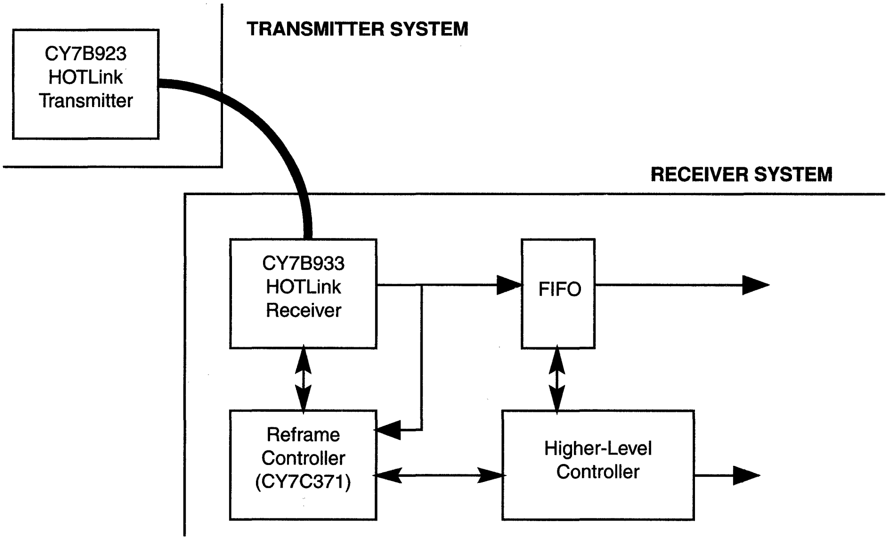


**Figure 1. Block Diagram**

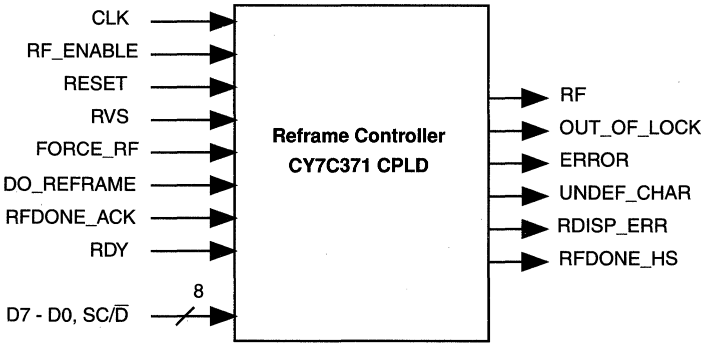


**Figure 2. Controller Inputs and Outputs**

DO_REFRAME signal has been completed (HIGH = asserted).

**OUT_OF_LOCK.** This signal indicates that the HOTLink Receiver's PLL has gone out of lock with the incoming serial bit stream. This is inferred by counting sixteen or more RVS assertions in a single 64-byte period. Once asserted, it remains asserted until the PLL regains lock and reframing has been accomplished (HIGH = asserted).

**ERROR.** When asserted (HIGH) it indicates to the higher-level controller that an error of some type (as indicated by the RVS signal from the receiver) has occurred.

**UNDEF_CHAR.** This is an undefined-character-error signal, one of two types of errors that can be decoded from the D7 - D0, SC/D̅ inputs during receive violations. This signal is only valid when the ERROR output is also asserted, and it can only be asserted when RDISP_ERR is deasserted (HIGH = asserted).

**RDISP_ERR.** Running-disparity-error signal. This is the other of the two types of errors that can be decoded from the D7–D0, SC/D̅ inputs during data-receive violations. This signal is only valid when the ERROR output is also asserted, and it can only be asserted when UNDEF_CHAR is deasserted (HIGH = asserted.)

### Counters

The primary function of the controller, which is to detect the out-of-lock condition by monitoring RVS and initiate a reframe when necessary, is implemented through the use of two counters. The VHDL for this function is shown in *Figure 3.* The

```
-- relevant VHDL code for counter functions

signal count2: bit_vector(0 to 1);          -- 2-bit counter
signal error_count: bit_vector(0 to 4);     -- 5-bit counter
signal rcvdbyts_count: bit_vector(0 to 6);   -- 7-bit counter

counters: process (CLK) begin

  if (clk'event and clk = '1') then

    if (reset = '1') then
       fb_out_of_lock    <= '0';
       rcvdbyts_count <= "0000000";
       error_count    <= "00000";
    elsif (error_count = "10000") then
       fb_out_of_lock    <= '1';
       rcvdbyts_count <= "0000000";
       error_count    <= "00000";
    elsif (rcvdbyts_count = "1000000") then
       rcvdbyts_count <= "0000000";
       error_count    <= "00000";
    else
       rcvdbyts_count  <= rcvdbyts_count + 1;
       if (RVS = '1') then
         error_count <= error_count + 1;
       end if;
    end if;

    if (current_state = LOOK_FOR_xRDY) and (xRDY = '0') then
      fb_out_of_lock <= '0';
    end if;

    if (current_state = COUNT_2_CLOCKS) then
      count2 <= count2 + 1;
    else
      count2 <= "00";
    end if;

    end if;

end process; --counters
```

**Figure 3. VHDL for Counter Functions**

first counter, rcvdbyts_count, is a seven-bit counter that counts the number of bytes received (0 to 64) and the second counter, error_count, is a five-bit counter that counts the number of times that RVS is asserted. If error_count reaches 16 before rcvdbyts_count reaches 64, then the out-of-lock condition will be declared. If rcvdbyts_count reaches 64 before error_count reaches 16, then fewer than 16 errors occurred in the given 64-byte window and out-of-lock is not declared. If rcvdbyts_count reaches 64 before error_count reaches 16, both rcvdbyts_count and error_count are set back to zero and a new 64-byte window begins. If the out-of-lock condition is declared (error_count = 16 and rcvdbyts_count ≤ 64), then the out-of-lock flip-flop is set to HIGH and a reframe operation is initiated. The out-of-lock flip-flop stays HIGH until the receiver successfully reframes. At that point, the out-of-lock flip-flop is set back to LOW and the search for the out-of-lock condition is started again.

### State Machine

The state machine is described by the diagram in *Figure 4*, and the VHDL code that implements it is shown in *Figure 5.*

#### *IDLE state*

The normal, quiescent state of the state machine, and the state it enters upon reset, is IDLE. In this state, the RF output is deasserted and the state machine waits for either a DO_REFRAME input from the outside or for the counters to set the out-of-lock flip-flop. If neither of these conditions occur, the state machine simply stays in the IDLE state. Once either one of these conditions occurs, the state machine must initiate a reframe, so it will go to the START_REFRAME state.

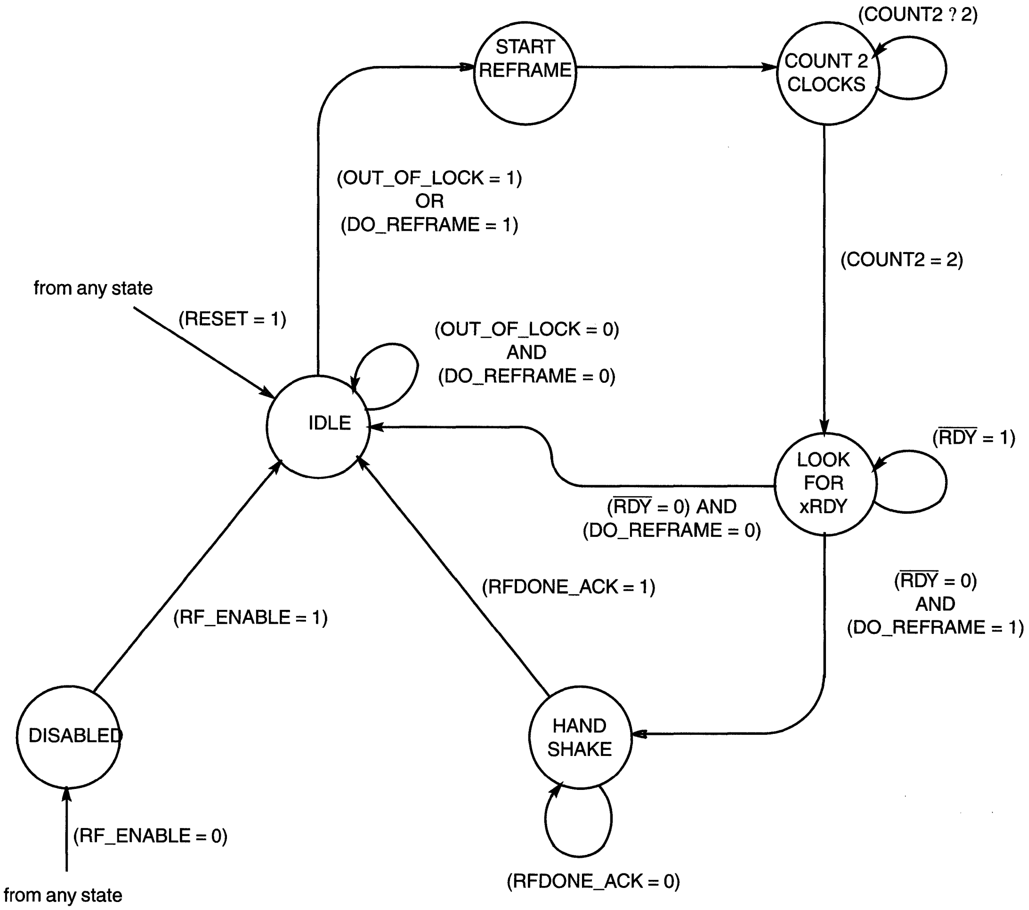


**Figure 4. State Diagram**

*START_REFRAME state*

In the START_REFRAME state, RF is asserted, and the state machine unconditionally transitions to the COUNT_2_CLOCKS state.

*COUNT_2_CLOCKS state*

The COUNT_2_CLOCKS state enables a two-bit counter to start counting incoming clock cycles. After two clock cycles have been counted, the state machine transitions to the LOOK_FOR_xRDY state. Two clock cycles must be counted before looking for the $\overline{\text{RDY}}$ signal from the outside because a total of three clocks must pass after RF is asserted until the value of $\overline{\text{RDY}}$ can be guaranteed valid (see the “HOTLink CY7B933 $\overline{\text{RDY}}$ Pin Description” application note for more details on this). One clock cycle passed during the START_REFRAME state, so the COUNT_2_CLOCKS state is used to count two more clock cycles to get to the requirement of three. RF is asserted throughout this state.

*LOOK_FOR_xRDY state*

On the fourth clock cycle from the start of RF, the value of $\overline{\text{RDY}}$ is guaranteed to be valid and the state machine, in the LOOK_FOR_xRDY state, continues to assert RF and waits until the HOTLink Receiver asserts $\overline{\text{RDY}}$. Once the receiver asserts RDY, it has successfully reframed and is ready to resume normal receiver operation. Thus, once an asserted $\overline{\text{RDY}}$ is detected in the LOOK_FOR_xRDY state, the state machine exits that state and goes back to the IDLE state. If the reframe was started by an out-of-lock detection, the transition back to the IDLE state is immediate; if the reframe was started by the DO_REFRAME input, then the state machine goes to the HANDSHAKE state first.

```
-- Relevant VHDL code for state machine

subtype StateType is bit_vector(0 to 2);      -- State Type
constant       DISABLED: StateType := b"111"; -- State Defns.
constant           IDLE: StateType := b"000";
constant  START_REFRAME: StateType := b"001";
constant COUNT_2_CLOCKS: StateType := b"010";
constant  LOOK_FOR_xRDY: StateType := b"011";
constant      HANDSHAKE: StateType := b"100";
signal current_state, next_state : StateType; --State declaration

-- State Machine Description

if (RESET = '1') then
  next_state <= IDLE;
elsif (RF_ENABLE = '0') then
  next_state <= DISABLED;
else
  case current_state is

  when IDLE =>
    if (fb_OUT_OF_LOCK = '1') or (DO_REFRAME = '1') then
      next_state <= START_REFRAME;
    else
      next_state <= current_state;
    end if;

  when START_REFRAME =>
    next_state <= Count_2_Clocks;

  when COUNT_2_CLOCKS =>
    if (count2 = "10") then
      next_state <= LOOK_FOR_xRDY;
    else
      next_state <= current_state;
    end if;

  when LOOK_FOR_xRDY =>
    if (xRDY = '0') and (DO_REFRAME = '1') then
      next_state <= HANDSHAKE;
    elsif (xRDY = '0') and (DO_REFRAME = '0') then
      next_state <= IDLE;
    else
      next_state <= current_state;
    end if;

  when HANDSHAKE =>
    if (RFDONE_ACK = '1') then
      next_state <= IDLE;
    else
      next_state <= current_state;
    end if;
```

**Figure 5. VDHL Code for State Machine**

```
  when DISABLED =>
    if (RF_ENABLE = '0') then
      next_state <= current_state;
    else
      next_state <= IDLE;
    end if;

  end case;
end if;

if (clk'event and clk = '1') then
    current_state <= next_state;
end if;
```

**Figure 5. VDHL Code for State Machine** (continued)

*HANDSHAKE state*

The HANDSHAKE state is used to make sure the reframe controller and the higher-level controller are consistent with each other. The only way this state will ever be entered is if the higher-level controller initiated a reframe by asserting DO_REFRAME to the reframe controller. Once that reframe has been completed by the receiver, the reframe controller communicates this to the higher-level controller by asserting RFDONE_HS. Once the higher-level controller acknowledges this assertion and is ready to proceed with normal receiving operation, it will assert RFDONE_ACK as confirmation to the reframe controller. It will simultaneously deassert DO_REFRAME so that once the state machine goes back to the IDLE state, that input is deasserted and does not erroneously cause another immediate pass into the reframe procedure. Once the state machine detects the RFDONE_ACK assertion, it exits the HANDSHAKE state and returns to the IDLE state. The RF operation is deasserted throughout the HANDSHAKE state.

*DISABLED state*

There is one more state, the DISABLED state, which is treated separately. As long as RF_ENABLE, the overall controller enable, is asserted, the state machine will never enter this state. If RF_ENABLE gets deasserted, the state machine will transition to the DISABLED state no matter what state it was in, and it will stay there until RF_ENABLE is once again asserted. Once RF_ENABLE is reasserted, the state machine goes to the IDLE state and resumes normal operation.

It was mentioned previously that the out-of-lock flip-flop is set when the out-of-lock condition is detected, and it stays set until the reframe has been completed. The exact time when the OUT_OF_LOCK flip-flop gets cleared is at the rising clock edge when the state machine exits the LOOK_FOR_xRDY state. This is because that is the exact point where the receiver has signalled to the controller, with $\overline{RDY}$, that it has successfully competed the reframe.

```
-- relevant VHDL code for Decode Logic

if (clk'event and clk = '1') then

  if (RVS = '1') then
    ERROR <= '1';
    if (D = x"E4" or D = x"E2" or D = x"E1") then
      UNDEF_CHAR <= '0';
      RDISP_ERR  <= '1';
    else
      UNDEF_CHAR <= '1';
      RDISP_ERR  <= '0';
    end if;
  else
    ERROR <= '0';
    UNDEF_CHAR <= '0';
    RDISP_ERR  <= '0';
  end if;

end if;
```

**Figure 6. VHDL Code for Decode Logic**

## Decode Logic

The error-decode logic is very straightforward, and the VHDL code for it is shown in *Figure 6.* The ERROR output is a registered version of the RVS input. The RDISP_ERR and UNDEF_CHAR outputs are decoded from the D7–D0, SC/$\overline{D}$ inputs. These outputs are also registered.

When the receiver asserts RVS, it will also put a code for the error type on its eight data outputs. If this code is E4, E2, or E1 (hex), it indicates the error is a running-disparity error, (explained earlier), and the RDISP_ERR output is asserted. If it is any other hex code, the receiver has detected some kind of illegal or undefined character, and the UNDEF_CHAR output will be asserted instead. These outputs are mutually exclusive: if one is asserted, the other must be deasserted. However, it is only meaningful to decode the data outputs when an error condition is detected, so the ERROR signal must be examined by the higher-level controller as well. If ERROR is not asserted, the output from RDISP_ERR and UNDEF_CHAR is no longer valid.

## VHDL, CY7C371 Utilization, and CY7C371 Speed Considerations

The complete VHDL description for this design is given in Appendix A. The full source code consists of the fragments shown throughout this application note along with the other code necessary to mesh it together, (process declarations, signal declarations, and package-entity declarations). As the fragments and complete source file show, VHDL is a very simple, efficient way for describing PLD designs. For example, the counter functions are simply bit vectors that are used in the manner: COUNT <= COUNT + 1. Upper limits for the counters, clearing functions, resets, and presets are all implemented with a few simple IF-THEN-ELSE statements. The entire state machine is implemented with a CASE statement and IF-THEN-ELSE statements that have a straightforward, natural, one-to-one correspondence with the bubble diagram shown in *Figure 4*. The entire set of decode logic is implemented in a single IF-THEN-ELSE clause. Furthermore, the VHDL code provided is easy to understand and can be very easily modified. For example, it can be modified to interface to different higher-level-controller interfaces than the one assumed in this application note, or it could be incorporated into the higher-level controller design, with that design consisting of other VHDL code and implemented in a larger FLASH370 CPLD or even a gate array.

This design used all 32 of the CY7C371's macrocells and 37 of its 38 I/O and input pins. It could have used fewer pins if necessary, by making the various counters be internal counters only. The outputs of the counters were brought out to output pins in this example, however, for easier simulation and debugging. The speed of the CY7C371 ranges from 66 MHz (with a 15-ns combinatorial propagation delay and a 12-ns clock-to-output time) to 143 MHz (with a 8.5-ns combinatorial propagation delay and a blazing 6-ns clock-to-output time). For this application, the maximum byte-rate clock of the CY7B933 is 33-MHz, and this and the corresponding set-up and hold times on the CY7B933 make the CY7C371-66 quite sufficient. The higher-level controller may have tighter timing requirements, but there is plenty of speed to be gained by going to the faster speed bins of the CY7C371. The design can, thus, easily meet much faster system timing requirements.

## Conclusion

The serial data received by the CY7B933 needs to be framed, i.e., aligned to the proper byte boundaries. This must always be done when the serial communication first begins, and it must always be redone if the PLL loses lock on the incoming serial bit stream. This application note described a controller that will manage this operation and provided some guidelines for determining when the periodic reframing is necessary. It assumed a particular interface to a higher-level controller, but the design was done in VHDL, which is provided in the appendix, to make it very easily modifiable and adaptable to any other specific interface. The controller itself is implemented in a CY7C371 32-macrocell CPLD, which had sufficient resources and routability to implement this fairly substantial function. It was able to do this exceeding system speed requirements even in its slowest speed bin.

## Appendix A. VHDL Description

```
-- Application Note
-- Using a CY7C371 as a HOTLink Reframe Controller
-- Cypress Semiconductor

entity CONTROLLER is port (
    CLK, RVS, RESET, xRDY, DO_REFRAME, FORCE_RFOUT, RFDONE_ACK,
      RF_ENABLE : in bit;
    D           : in bit_vector(0 to 7);
    curr_st     : out bit_vector (0 to 2);
    rb_cntr     : out bit_vector (0 to 6);
    err_cntr    : out bit_vector (0 to 4);
    RF, RFDONE_HS, OUT_OF_LOCK, UNDEF_CHAR, RDISP_ERR, ERROR : out bit
);
end CONTROLLER;

architecture CNTRL933 of CONTROLLER is

    subtype StateType is bit_vector(0 to 2);       -- State Type
    constant       DISABLED: StateType := b”111";   -- State Definitions
    constant           IDLE: StateType := b”000";
    constant  START_REFRAME: StateType := b”001";
    constant COUNT_2_CLOCKS: StateType := b”010";
    constant  LOOK_FOR_xRDY: StateType := b”011";
    constant      HANDSHAKE: StateType := b”100";

    signal current_state, next_state : StateType;
    signal fb_OUT_OF_LOCK : bit;

    signal count2: bit_vector(0 to 1);         -- 2-bit counter
    signal error_count: bit_vector(0 to 4);    -- 5-bit counter
    signal rcvdbyts_count: bit_vector(0 to 6); -- 7-bit counter

begin

counters: process (CLK) begin

  if (clk’event and clk = ’1’) then

    if (reset = ’1’) then
      fb_out_of_lock    <= ’0’;
      rcvdbyts_count <= ”0000000";
      error_count    <= ”00000";
    elsif (error_count = ”10000") then
      fb_out_of_lock     <= ’1’;
      rcvdbyts_count <= ”0000000";
      error_count    <= ”00000";
    elsif (rcvdbyts_count = ”1000000") then
      rcvdbyts_count <= ”0000000";
      error_count    <= ”00000";
    else
      rcvdbyts_count  <= rcvdbyts_count + 1;
      if (RVS = ’1’) then
        error_count <= error_count + 1;
      end if;
    end if;
```

## Appendix A. VHDL Description (continued)

```
    if (current_state = LOOK_FOR_xRDY) and (xRDY = '0') then
      fb_out_of_lock <= '0';
    end if;

    if (current_state = COUNT_2_CLOCKS) then
      count2 <= count2 + 1;
    else
      count2 <= “00";
    end if;

  end if;

end process; --counters

next_st_comb:  process (fb_OUT_OF_LOCK, DO_REFRAME, COUNT2, xRDY,
                        RFDONE_ACK, RESET, RF_ENABLE, current_state) begin

  if (RESET = '1') then
    next_state <= IDLE;
  elsif (RF_ENABLE = '0') then
    next_state <= DISABLED;
  else
    case current_state is

    when IDLE =>

      if (fb_OUT_OF_LOCK = '1') or (DO_REFRAME = '1') then
        next_state <= START_REFRAME;
      else
        next_state <= current_state;
      end if;

    when START_REFRAME =>

      next_state <= Count_2_Clocks;

    when COUNT_2_CLOCKS =>

      if (count2 = “11") then
        next_state <= LOOK_FOR_xRDY;
      else
        next_state <= current_state;
      end if;

    when LOOK_FOR_xRDY =>

      if (xRDY = '0') and (DO_REFRAME = '1') then
        next_state <= HANDSHAKE;
      elsif (xRDY = '0') and (DO_REFRAME = '0') then
        next_state <= IDLE;
      else
        next_state <= current_state;
      end if;
```

## Appendix A. VHDL Description (continued)

```
    when HANDSHAKE =>

      if (RFDONE_ACK = '1') then
        next_state <= IDLE;
      else
        next_state <= current_state;
      end if;

    when DISABLED =>

      if (RF_ENABLE = '0') then
        next_state <= current_state;
      else
        next_state <= IDLE;
      end if;

    end case;
  end if;
end process; --next_st_comb

outp_comb:  process (current_state, FORCE_RFOUT) begin

  if (FORCE_RFOUT = '1') then
    RF <= '1';
  else
    case current_state is

    when IDLE =>
      RF         <=  '0';
      RFDONE_HS  <=  '0';

    when START_REFRAME =>
      RF         <=  '1';
      RFDONE_HS  <=  '0';

    when COUNT_2_CLOCKS =>
      RF         <=  '1';
      RFDONE_HS  <=  '0';

    when LOOK_FOR_xRDY =>
      RF         <=  '1';
      RFDONE_HS  <=  '0';

    when HANDSHAKE =>
      RF         <=  '0';
      RFDONE_HS  <=  '1';

    end case;
  end if;
end process; --outp_comb
```

## Appendix A. VHDL Description (continued)

```
seq_assgnmnt:  process (clk) begin

  if (clk'event and clk = '1') then

    current_state <= next_state;

    if (RVS = '1') then
      ERROR <= '1';
      if (D = x"E4" or D = x"E2" or D = x"E1") then
        UNDEF_CHAR <= '0';
        RDISP_ERR  <= '1';
      else
        UNDEF_CHAR <= '1';
        RDISP_ERR  <= '0';
      end if;
    else
      ERROR <= '0';
      UNDEF_CHAR <= '0';
      RDISP_ERR  <= '0';
    end if;

  end if;

end process; --seq_assgnmnt

-- concurrent assignment statements
-- outputs and local feedback signals made the same

curr_st      <=  current_state;
rb_cntr      <=  rcvdbyts_count;
err_cntr     <=  error_count;
OUT_OF_LOCK <=  fb_out_of_lock;

end CNTRL933;  -- end architecture
```

HOTLink, *Warp*, and FLASH370 are trademarks of Cypress Semiconductor Corporation.

# Implementing a 128Kx32 Dual-Port RAM Using the FLASH370™

## Introduction

More and more communication systems require the use of very deep, high-speed dual-port memories to provide a common storage area for use between processors. System designers are looking for dual-port memories of 128 KByte and larger in size. These same systems are using 32-bit buses. These larger dual-port memories are not readily available as monolithic devices. As a result, the designer is left with the task of implementing these devices using discrete components. A full-featured implementation would include some static RAM combined with external support logic, arbitration, and control functions. This application note describes how to implement a 128K x 32-bit-wide dual-port memory or larger, using high-speed 1M SRAMs and a Cypress CPLD, the CY7C371. The CPLD, or Complex Programmable Logic Device, will be used to implement the memory control functions of the dual-port system and will be coded using VHDL.

## Dual-Port Block Diagram

A good reference for the function and operation of a dual-port memory can be found in the application note in the Cypress *Applications Handbook* titled “Understanding Dual-Port RAMs.” To reiterate, the block diagram of a standard dual-port memory is shown in *Figure 1*. This block diagram indicates the various blocks associated with a dual-port. There are four major blocks: the memory array, the arbitration/control function, the right port or interface, and the left port or interface.

As can be seen from the block diagram in *Figure 1*, there are a series of signals that are required both internal and external to this system. The external signals are the normal signals that a monolithic dual-port chip would have. These are the signals that are labeled in the block diagram. The other signals are the internal signals that are used to allow the pieces of this dual-port system to communicate with one another. These are the address output enables for the address interface logic, the data output enable and the latch enable for the data interface logic, and the RAM output enable and write enable. These will be discussed in detail later.

The memory array consists of a single, standard SRAM or group of SRAMs to make up the overall array size. This array can be expanded in depth and width as needed. The arbitration/control logic accepts asynchronous read or write requests from each port or interface and sequences through a series of internal states that perform the read or write operation on the memory array. A CPLD is used in this example to implement this logic. The control logic must arbitrate between requests as well as synchronize the inputs to the internal clock frequency of the control function. The address buffers are used to isolate the address bus of the memory array from the left and right address ports. This allows the control-logic CPLD to select the correct address at the proper time. The bidirectional, latched data path allows data to be written to or read from the memory array. The data is also held in the latch during the remainder of the access.

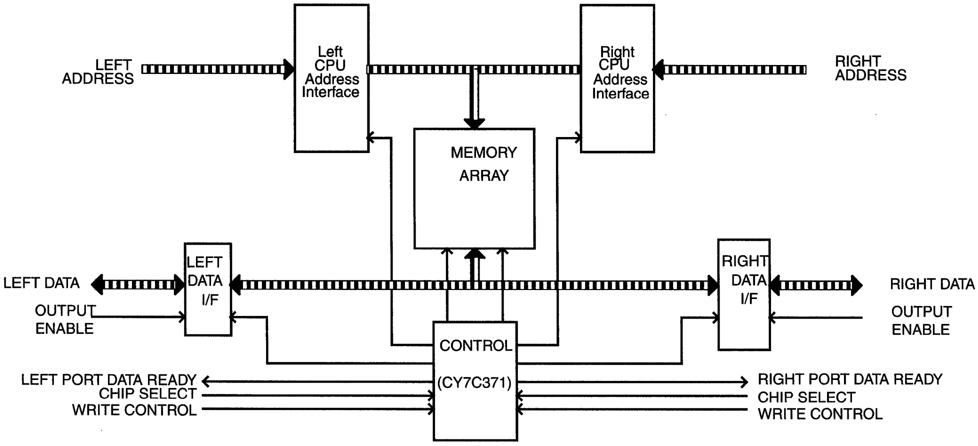


**Figure 1. Dual-Port Memory Array Block Diagram**

## Use of SRAM for Dual-Port

A 128Kx8 SRAM (like the Cypress CY7C109, 25-ns SRAM, as used in this note) was chosen here to implement a 128K x 32 sized array. Appendix A shows the schematic representation of the design. The array can be any size; this note shows this configuration because it depicts how to expand in the width direction. Cascading devices to expand the depth of the array is just as easily implemented. In either case, the contents of the control logic CPLD remain the same. The array could also be implemented with a single SRAM device if the array size warrants it.

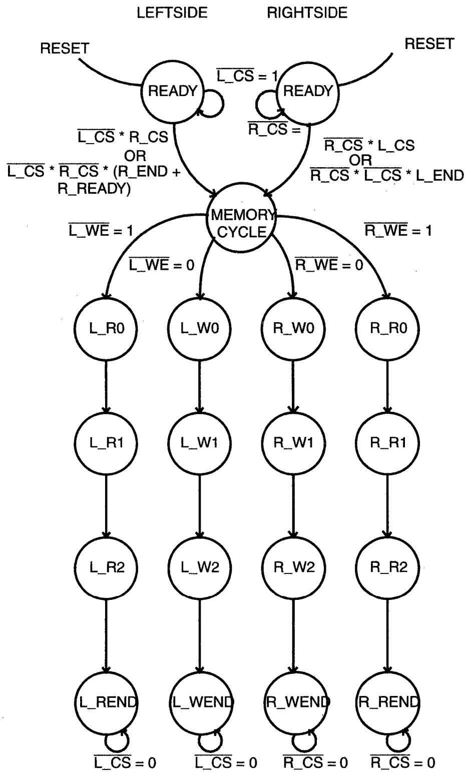
## A Brief Description of the CY7C371

The CY7C371 is a complex PLD with 32 macrocells, 32 I/O pins and 6 dedicated input pins (including 2 clock pins). The macrocells are grouped into two Logic Blocks of 16 macrocells each. There is a programmable interconnect matrix or PIM that connects the two logic blocks to the inputs and to each other. The macrocells themselves contain a register that can be configured as a T flip-flop, a D flip-flop, a level-triggered latch, or can be bypassed for combinatorial product terms. Each macrocell can support up to 16 product terms. For more detailed information on the CY7C371 and the whole FLASH370™ family of CPLDs, please consult the application note "The FLASH370 Family Of CPLDs and Designing with *Warp2®*" in the *Cypress Applications Handbook.*

The CY7C371 is well suited to this application. The dedicated inputs can be configured with a double registering mechanism to synchronize asynchronous signals so that they can be used synchronously inside the CPLD. The double registering will also dramatically reduce the chance of a metastable condition. The CPLD architecture is optimal for state machine designs and this arbiter requires three state machines to define it. The double-registered input configuration will be used in this example to resync the asynchronous chip select and write control inputs from both ports.

## State Machine Design

The finite state machine that controls the dual-port memory array is really comprised of three "dependent" state machines operating concurrently as shown in *Figure 2.* Dependent state machines monitor or depend on the state of another state machine in order to change state. The first two machines, called "leftside" and "rightside," are identical. Their primary task is to monitor the interface of both ports. When the chip select input ($\overline{\text{R\_CS}}$ or $\overline{\text{L\_CS}}$) goes active (logic LOW), the appropriate machine advances from the Ready state to the Memory Cycle state. The Memory Cycle state will start one of the memory access sequences. The length of each memory sequence (i.e., the number of state machine cycles) can be "tuned" to the access time of the SRAMs in the memory array. The memory cycle state machine will cycle back to the Ready state at the same time the memory access sequence ends and the select input goes inactive. It will either wait for a new request or start another memory access depending on the state of the other state machine ("leftside" or "rightside"). In the case where two requests are pending or appear at the same time, the left port gets priority. This means that the memory access for the left port is performed first. A READY signal (L_READY and R_READY) indicates when data is available on either port, it can only be active when either select input is active.



ALL RETURN TO 'READY' WHEN CS GOES INACTIVE

**Figure 2. Memory Control Function State Machine**

## State Machine Implementation

The actual implementation of the state machines in the CY7C371 is done using VHDL. The structure of VHDL allows for simplification in coding these dependent state machines; the use of multiple processes and the CASE statement prove to be very powerful and efficient ways to perform this task.

Upon reset, both rightside and leftside state machines enter the Ready state and wait for a memory access. The leftside state machine will be used as an example. Both sides are identical at this point. Once a request is detected (for example $\overline{\text{L\_CS}}$ goes active (=0)), the leftside state machine transitions into a memory cycle. A priority scheme favoring the left port is encoded into the process for both state machines. If two accesses occur simultaneously, the left one is performed first. If one port request is detected before the other, it is completed while the other is held off. This extends the overall access time of the memory, but allows for "fair" operation. Each memory access sequence, Left Read, Left Write, Right Read, and Right Write, is comprised of four states. The four states (R0, R1, R2, REND or W0, W1, W2, WEND) run sequentially, one per clock cycle. They are there to allow the proper timing for the generation of control signals to the various components in the dual-port system. The REND or WEND state indicates the end of a memory cycle and is also a hold state if the CS is still active for that particular port. Once the REND or WEND state is reached and the CS is inactive, the state machine returns to the READY state and another access can be initiated.

## CY7C371 Signals

A total of ten outputs are required to control the memory array and both the left and right ports. Refer to Appendix A for the 128K x 8 dual-port memory array schematic. The SRAM in the array is controlled by RAM_OE and RAM_WE. The RAM_OE signal is created when either port executes a read successfully. Therefore, the RAM_OE signal is enabled during either read sequence only during the R0 through R2 cycles. Writes to the SRAM are controlled by the write state machine for either port. The RAM_WE is generated for either port during the W1 and W2 cycles of a write access only. The port address inputs are isolated from the memory array by a set of 74FCT244Ts. The left port is controlled by L_ADD_OE and is generated during the left memory access sequence states 0 through 2 for either a read or a write to the left port. The right port address is controlled in the same manner, by using the right memory access sequence states 0 through 2. The data buffer functions are implemented using 74FCT543Ts with the "B" (HIGH current) side interfaced to the outside and the "A" side interfaced to the memory array. During reads, the latch enables (L_LAT_EN, R_LAT_EN) are used to hold the data read from the array in the latches. The output enables (L_OE, R_OE) are then driven directly to access the read data. During writes, the output enables (L_DAT_OE, R_DAT_OE) are used to allow the data to pass from the outside into the memory array. These output enables and latch enables are controlled by the OR of the appropriate memory access sequence states. Mealy outputs are used for the L_READY and R_READY signals. These outputs are active whenever the respective state machine is in state 2 and the CS is active. Using Mealy outputs here allows the ready signal to go inactive as soon as the CS input (L_CS or R_CS) goes inactive instead of waiting for the state machine to transition back to the READY state.

## VHDL Code for Controller in 371

Appendix B contains the VHDL code used for the CY7C371 in this design. This code was compiled with the Cypress *Warp2* tool and targeted for the CY7C371 to generate the programming (JEDEC) and simulation file(s). The Nova simulator in the *Warp2* tool was used to verify the design. For details on these tools please refer to the *Warp2 User's Guide*. Furthermore, a thorough explanation of VHDL constructs can be found in the *Warp2 Reference Manual*.

The code in Appendix B starts out by defining the inputs and outputs and the internal signals required. The first process is for the Chip Select and Write Enable resync. This is where the double registering occurs, as mentioned in the description of the CY7C371 earlier in this application note. The next process is where the state machine definitions start. It begins by defining the rightside state machine and uses a separate process to define the leftside state machine. Buried within each of these processes is the Memory Cycle state machine for the READ and WRITE cycles of each port. The next process is used to define the RAM_OE and RAM_WE for the memory array control. This is a simple IF-THEN-ELSE clause. The last process is used to generate the signal which gets used in the Mealy equations for the leading edge of the L_READY and R_READY signals. Lastly, the L_READY and R_READY signals are defined outside of a process by gating state2 with the CS input.

## Performance Evaluation

To evaluate the performance of this dual-port system, three different timing scenarios were looked at. The first scenario is for an unarbitrated access from either port. This assumes that both port state machines are in the Ready state and only one access occurs. The second scenario involves the right port being granted access shortly before the left port, forcing the left port to wait. The third involves simultaneous accesses from each port. In this case the left side has priority (by design) and the right side is held off. These cases are shown in the following three timing diagrams (*Figures 3*, *4*, and *5*). From these it is possible to determine the timing of each access by counting the number of clock cycles for each scenario.

*Table 1* lists the number of clock cycles for each of the three cases of *Figures 3*, *4*, and *5*. These numbers reflect the worst case situations for Case #2 and #3 where the maximum possible delay is assumed.

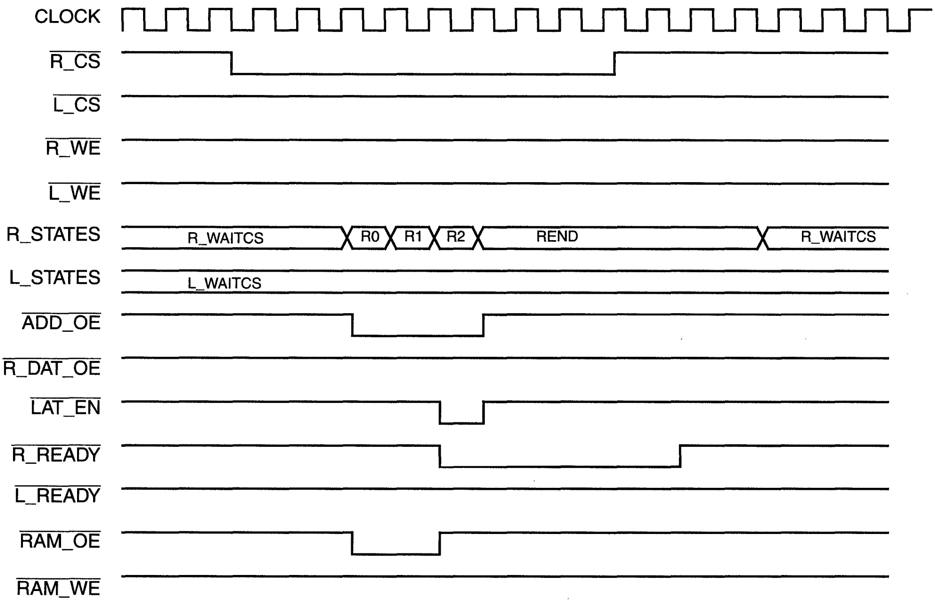


**Figure 3. Timing Diagram—Unarbitrated Access From Right Port**

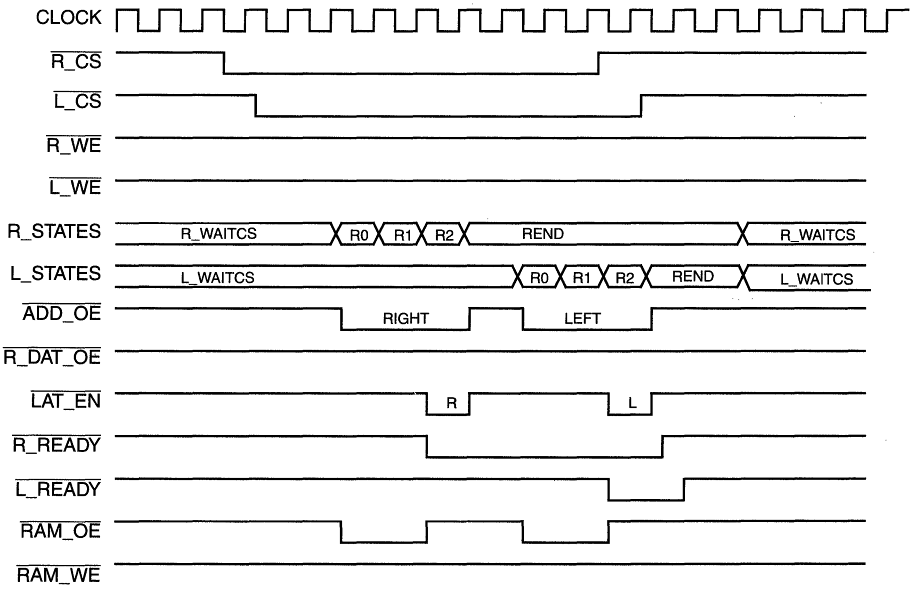


**Figure 4. Timing Diagram—Right Port Access Before Left Port**

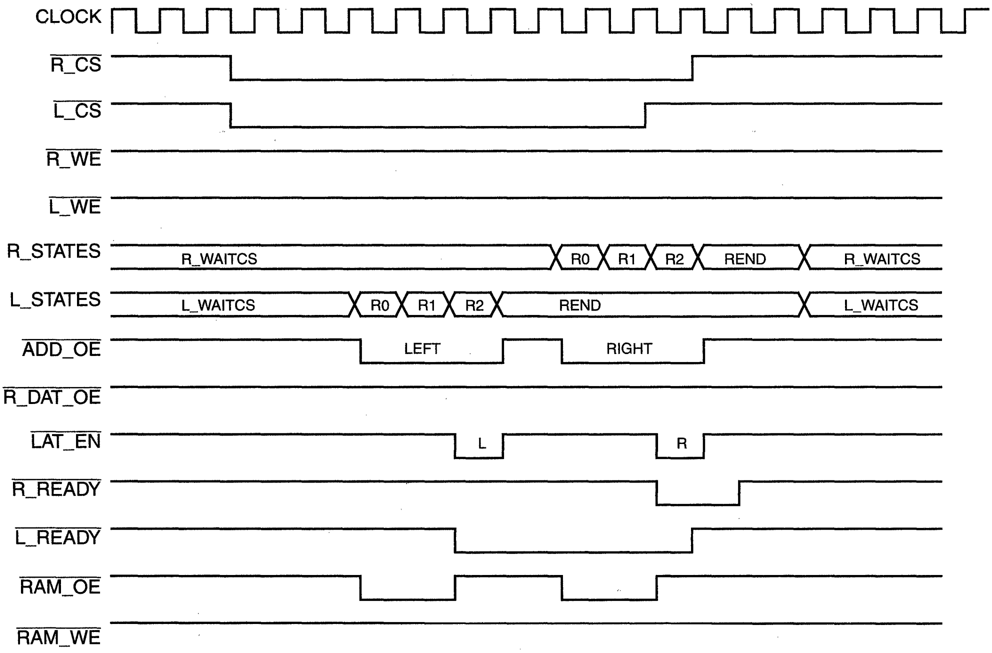


**Figure 5. Timing Diagram—Simultaneous Access**

**Table 1. Access Time in Clock Cycles**

| **Timing Parameter** | **Case #1** | **Case #2** | **Case #3** |
|---|---|---|---|
| LEFT | | | |
| Input Set-Up Timing | 2 clocks | Note 1 | 2 clocks |
| Arbitration Cycle | 1 clock | Note 1 | 1 clock |
| Memory Access | 3 clocks | 3 clocks | 3 clocks |
| Latch Hold Cycle | 1 clock | 1 clock | 1 clock |
| Total Number of Clock Cycles | 7 clocks | 11 clocks | 7 clocks |
| RIGHT | | | |
| Input Set-Up Timing | N/A[2] | 2 clocks | Note 1 |
| Arbitration Cycle | N/A | 1 clock | Note 1 |
| Memory Access | N/A | 3 clocks | 3 clocks |
| Latch Hold Cycle | N/A | 1 clock | 1 clock |
| Total Number of Clock Cycles | N/A | 7 clocks | 11 clocks |

**Notes:**

1. Worst case input set-up timing and arbitration cycle assumes 7 clock access delay on opposite port.
2. N/A means No Activity on this port.

To calculate the access time in nanoseconds, the following formula is applied:

$t_{ACC} = t_{IS371} + [t_{CYC371} \text{ x \#clocks}] + t_{PD543}$

Where:

$t_{ACC}$ = total access time

$t_{IS371}$ = CY7C371 input register set-up time = 2 ns

$t_{CYC371}$ = clock cycle of CY7C371 = 7 ns

#clocks = number of clocks from *Table 1*

$t_{PD543}$ = 74FCT543CT transparent to latched propagation delay = 7 ns

Since the CY7C371 inputs are double registered, two clock cycles are required to resync the Chip Select and Write Enable inputs. If the input set-up timing can be guaranteed, this internal delay of two cycles can be eliminated by using single- or non-registered inputs.

## Memory Expansion

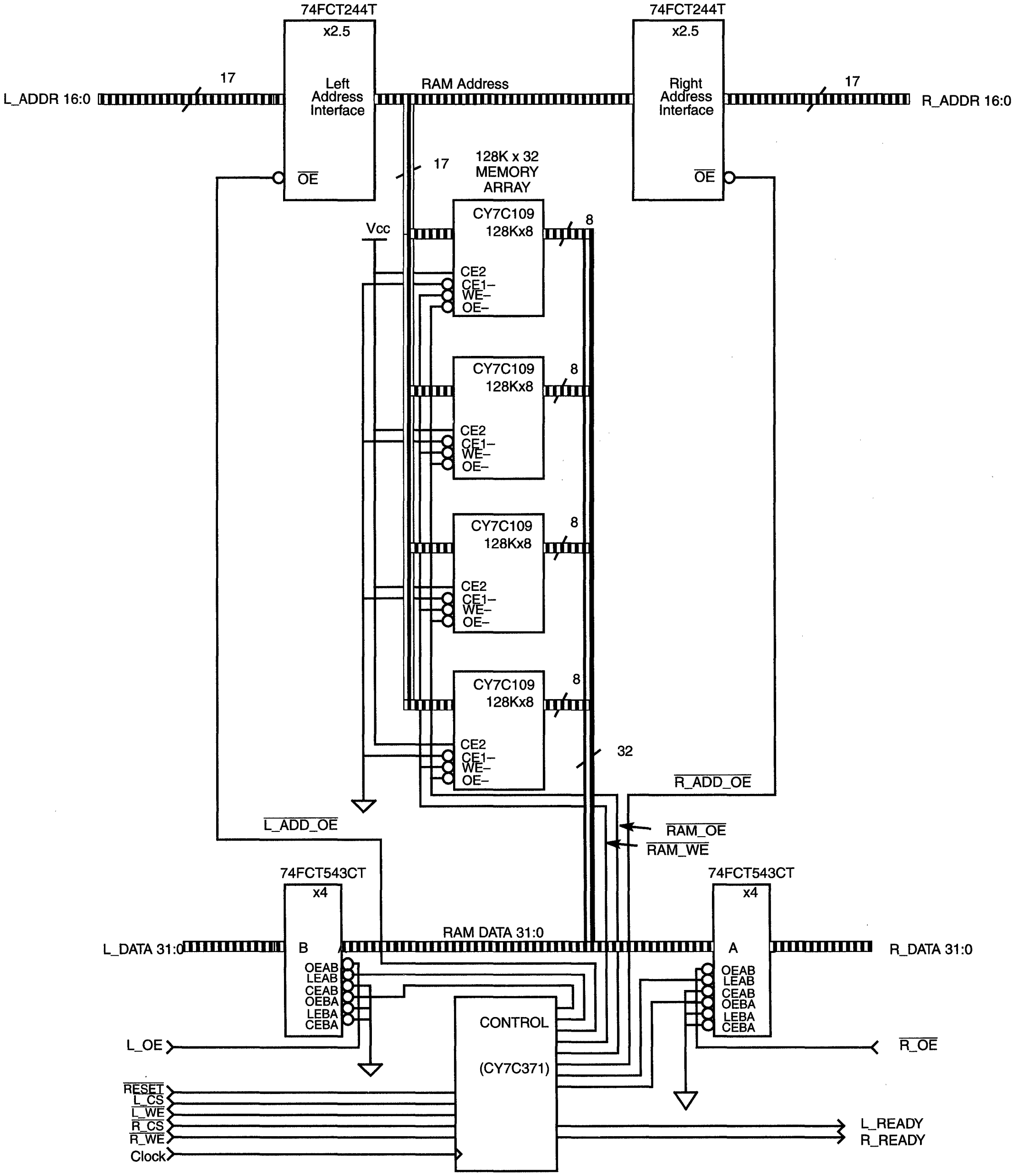
The example used here shows that an array of any size can be easily implemented. The addition of memories and associated address buffers makes depth expansion easy. The width may also be increased by cascading memories and adding additional buffers. Both techniques would be utilized to expand in depth and width. These enhancements are possible without making any changes to the CY7C371 Control Function PLD design. Likewise this design could implement a smaller array than shown here, again without revising the CY7C371.

## Summary

This application note has demonstrated the implementation of a large asynchronous dual-port memory array by utilizing standard memory and logic devices and the CY7C371. The performance of this design is limited by various factors. The access time of the SRAM and the clock speed of the CY7C371 used are two factors that could improve performance without changing the VHDL code for the CY7C371. Another option would require some design changes, though minor. Making one or both ports synchronous with respect to the CPU would eliminate the two-clock delay associated with the resync function of the CY7C371. The implementation of these improvements offers the designer a few options to tailor the design to fit specific system requirements and achieve the desired level of performance.

5

## Appendix A. Schematic



## Appendix B. VHDL Code for Controller

```
-- Dual-port memory controller

ENTITY dpram IS
        PORT (clock, r_we_n, r_cs_n, l_we_n, l_cs_n, reset_n: IN BIT;    --INPUTS
              ram_oe_n, ram_we_n : OUT BIT;                              --OUTPUTS
              r_ready, r_add_oe, r_dat_oe, r_lat_en : OUT BIT;
              l_ready, l_add_oe, l_dat_oe, l_lat_en : OUT BIT
              );
END dpram;

--------------------------------------------------------------------
USE work.rtlpkg.all;

ARCHITECTURE ARCHdpram OF dpram IS
  TYPE ctrl_states IS (waitcs, r0, r1, r2, rend, w0, w1, w2, wend);      --Internal signal declaration
  SIGNAL rightside, leftside : ctrl_states;
  SIGNAL r_we_ndd, r_we_nd, l_we_ndd, l_we_nd : BIT;
  SIGNAL r_cs_ndd, r_cs_nd, l_cs_ndd, l_cs_nd : BIT;
  SIGNAL r_ready_int, l_ready_int : BIT;
BEGIN

--Double register the input we and cs signals for sync & metastability hardening
  PROCESS BEGIN
    WAIT UNTIL clock = '1';
      r_we_ndd <= r_we_nd; r_we_nd <= r_we_n;
      l_we_ndd <= l_we_nd; l_we_nd <= l_we_n;
      r_cs_ndd <= r_cs_nd; r_cs_nd <= r_cs_n;
      l_cs_ndd <= l_cs_nd; l_cs_nd <= l_cs_n;
  END PROCESS;

--RIGHTSIDE STATE MACHINE
  PROCESS BEGIN
    WAIT UNTIL clock = '1';
      CASE rightside IS
        WHEN waitcs =>
          r_add_oe <= '1'; r_dat_oe <= '1'; r_lat_en <= '1';
--goto state 0 if : r_cs is active + L_cs  inactive or r_cs active + (l_cs active but at end)
          IF (((r_cs_ndd = '0') AND (l_cs_ndd = '1')) OR
              ((r_cs_ndd = '0') AND (l_cs_ndd = '0') AND
                ((leftside = wend) OR (leftside = rend)))) THEN
--start write state machine if WE active
            IF r_we_ndd = '0' THEN
              rightside <= w0;
              r_add_oe <= '0'; r_dat_oe <= '0'; r_lat_en <= '1';
            ELSE
--start read state machine if WE inactive
              rightside <= r0;
              r_add_oe <= '0'; r_dat_oe <= '1'; r_lat_en <= '1';
            END IF;
          ELSE
            rightside <= waitcs;
          END IF;

--RIGHTSIDE READ STATE MACHINE
        WHEN r0 =>
          rightside <= r1;
          r_add_oe <= '0'; r_dat_oe <= '1'; r_lat_en <= '1';
        WHEN r1 =>
          rightside <= r2;
          r_add_oe <= '0'; r_dat_oe <= '1'; r_lat_en <= '0';
        WHEN r2 =>
          rightside <= rend;
          r_add_oe <= '1'; r_dat_oe <= '1'; r_lat_en <= '1';
```

## Appendix B. VHDL Code for Controller (continued)

```
        WHEN rend =>
          r_add_oe <= '1'; r_dat_oe <= '1'; r_lat_en <= '1';
          IF r_cs_ndd = '1' THEN
            rightside <= waitcs;
          ELSE
            rightside <= rend;
          END IF;

--RIGHTSIDE WRITE STATE MACHINE
        WHEN w0 =>
          rightside <= w1;
          r_add_oe <= '0'; r_dat_oe <= '0'; r_lat_en <= '1';
        WHEN w1 =>
          rightside <= w2;
          r_add_oe <= '0'; r_dat_oe <= '0'; r_lat_en <= '1';
        WHEN w2 =>
          rightside <= wend;
          r_add_oe <= '1'; r_dat_oe <= '1'; r_lat_en <= '1';
        WHEN wend =>
          r_add_oe <= '1'; r_dat_oe <= '1'; r_lat_en <= '1';
          IF r_cs_ndd = '1' THEN
            rightside <= waitcs;
          ELSE
            rightside <= wend;
          END IF;
        WHEN others =>
          rightside <= waitcs;
          r_add_oe <= '1'; r_dat_oe <= '1'; r_lat_en <= '1';
      END CASE;
  END PROCESS;
--LEFTSIDE STATE MACHINE
  PROCESS BEGIN
    WAIT UNTIL clock = '1';
      CASE leftside IS
        WHEN waitcs =>
          l_add_oe <= '1'; l_dat_oe <= '1'; l_lat_en <= '1';
--goto state 0 if l_cs is active + r_cs is inactive or l_cs active + (r_cs active but at end
or in waitcs   state)
          IF (((l_cs_ndd = '0') AND (r_cs_ndd = '1')) OR
              ((l_cs_ndd = '0') AND (r_cs_ndd = '0') AND
               ((rightside = wend) OR (rightside = rend) OR (rightside = waitcs)))) THEN
--start write state machine if WE active
            IF l_we_ndd = '0' THEN
              leftside <= w0;
              l_add_oe <= '0'; l_dat_oe <= '0'; l_lat_en <= '1';
            ELSE
--start read state machine if WE inactive
              leftside <= r0;
              l_add_oe <= '0'; l_dat_oe <= '1'; l_lat_en <= '1';
            END IF;
          ELSE
            leftside <= waitcs;
          END IF;
--LEFTSIDE READ STATE MACHINE
        WHEN r0 =>
          leftside <= r1;
          l_add_oe <= '0'; l_dat_oe <= '1'; l_lat_en <= '1';
        WHEN r1 =>
          leftside <= r2;
          l_add_oe <= '0'; l_dat_oe <= '1'; l_lat_en <= '0';
        WHEN r2 =>
          leftside <= rend;
          l_add_oe <= '1'; l_dat_oe <= '1'; l_lat_en <= '1';
```

## Appendix B. VHDL Code for Controller (continued)

```
        WHEN rend =>
          l_add_oe <= '1'; l_dat_oe <= '1'; l_lat_en <= '1';
          IF l_cs_ndd = '1' THEN
            leftside <= waitcs;
          ELSE
            leftside <= rend;
          END IF;

--LEFTSIDE WRITE STATE MACHINE
        WHEN w0 =>
          leftside <= w1;
          l_add_oe <= '0'; l_dat_oe <= '0'; l_lat_en <= '1';
        WHEN w1 =>
          leftside <= w2;
          l_add_oe <= '0'; l_dat_oe <= '0'; l_lat_en <= '1';
        WHEN w2 =>
          leftside <= wend;
          l_add_oe <= '1'; l_dat_oe <= '1'; l_lat_en <= '1';
        WHEN wend =>
          l_add_oe <= '1'; l_dat_oe <= '1'; l_lat_en <= '1';
          IF l_cs_ndd = '1' THEN
            leftside <= waitcs;
          ELSE
            leftside <= wend;
          END IF;
        WHEN others =>
          leftside <= waitcs;
          l_add_oe <= '1'; l_dat_oe <= '1'; l_lat_en <= '1';
      END CASE;
  END PROCESS;

--RAM_OE and RAM_WE control signal logic
  PROCESS BEGIN
    WAIT UNTIL clock = '1';
          IF (((rightside = waitcs) AND ((((r_cs_ndd = '0') AND (l_cs_ndd = '1') AND (r_we_ndd = '1')) OR
              ((r_cs_ndd = '0') AND (l_cs_ndd = '0') AND (r_we_ndd = '1') AND
                ((leftside = wend) OR (leftside = rend))))))
             OR
             ((leftside = waitcs) AND (((l_cs_ndd = '0') AND (r_cs_ndd = '1') AND (l_we_ndd = '1')) OR
              ((l_cs_ndd = '0') AND (r_cs_ndd = '0') AND (l_we_ndd = '1') AND
                ((rightside = wend) OR (rightside = rend) OR (rightside = waitcs)))))
             OR
             (rightside = r0) OR (rightside = r1)
             OR
             (leftside = r0) OR (leftside = r1))
          THEN
            ram_oe_n <= '0';
          ELSE
            ram_oe_n <= '1';
          END IF;

          IF ((leftside = w0) OR (leftside = w1) OR
              (rightside = w0) OR (rightside = w1))
          THEN
            ram_we_n <= '0';
          ELSE
            ram_we_n <= '1';
          END IF;
  END PROCESS;
```

## Appendix B. VHDL Code for Controller (continued)

```
--READY signal logic for leading edge of signal
  PROCESS BEGIN
    WAIT UNTIL clock = '1';
      IF ((rightside = r1) OR (rightside = w1)) THEN
        r_ready_int <= '0';
      END IF;
      IF ((r_cs_nd = '1') OR (reset_n = '0')) THEN
        r_ready_int <= '1';
      END IF;

      IF ((leftside = r1) OR (leftside = w1)) THEN
        l_ready_int <= '0';
      END IF;
      IF ((l_cs_nd = '1') OR (reset_n = '0')) THEN
        l_ready_int <= '1';
      END IF;
  END PROCESS;

--MEALY outputs for READY signal to turn off as soon as CS goes inactive
  l_ready <= '0' WHEN ((l_ready_int = '0') AND (l_cs_nd = '0')) ELSE '1';
  r_ready <= '0' WHEN ((r_ready_int = '0') AND (r_cs_nd = '0')) ELSE '1';

END ARCHdpram;
```

FLASH370 is a trademark of Cypress Semiconductor Corporation.
*Warp2* is a registered trademark of Cypress Semiconductor Corporation.

CYPRESS

# An Introduction to In System Reprogramming with FLASH370i™

## Introduction

This application note provides an introduction to the FLASH370i™ family of In System Reprogrammable (ISR™) CPLDs. The FLASH370i ISR CPLD family is a superset replacement for the popular FLASH370™ CPLD family. All of the features and advantages of the FLASH370 CPLD architecture are present in the FLASH370i CPLD architecture, with the FLASH370i family also providing additional capabilities. The most celebrated new feature is the capability for the FLASH370i devices to be reprogrammed in the user's system.

In System Reprogrammability is the capability of a programmable device to be reprogrammed after being soldered onto a printed circuit board. This allows the configuration and functionality of a device, and therefore the electronic system in which it is contained, to be modified after the system has completed the manufacturing cycle. Reprogramming can happen at any time, such as prior to the shipment of the system or product, in the form of a field upgrade to provide additional capabilities and features or to correct previous problems, or during actual system operation so that a different function or algorithm is implemented.

In many cases, ISR capability is used only to simplify the manufacturing cycle, by eliminating the handling of a traditional programmable logic device to facilitate programming. With an ISR device, one handling step can be removed if desired. Additionally, ISR devices can be reprogrammed again and again as required.

Several topics and issues are introduced in this application note. These are: the compatibility of the FLASH370i CPLD family with the FLASH370 family, the features of ISR, the FLASH370i pin descriptions and pinouts, the ISR programming options, the ISR programming interface, the FLASH370i ISR kit, and the other application notes that address other ISR design issues more specifically.

## Compatibility of the FLASH370i and FLASH370 CPLD Families

Like the FLASH370 CPLD family, the FLASH370i family consists of five devices at three standard densities: 32, 64, and 128 macrocells. For each density, there is a device that has all macrocells associated with I/O pins, and for the 64-macrocell and 128-macrocell densities, there are devices that have half of the macrocells associated with I/O pins. Each of the five devices has additional dedicated input pins with up to four of these available as high-speed clocks.

The CY7C371i 32-macrocell device has the same number of flexible macrocells, the same number and superior allocation of product terms, the same robust interconnect that provides the same excellent routability, and the same output enable capability as the CY7C371 device that it replaces. This same statement applies equally to the other four devices of the FLASH370i CPLD family. The normal PAL mode functionality of a FLASH370i device is also exactly the same as the normal PAL mode functionality of its FLASH370 counterpart. The speed grades of the two CPLD families match, while maintaining the same parallel programming algorithms.

As previously mentioned, the major difference between the FLASH370i and the FLASH370 CPLD families is that the FLASH370i CPLD family provides the capability for in system reprogramming. Other new features of the FLASH370i devices are outputs that are fully PCI (peripheral component interconnect) compliant and bus-hold structures on all pins. While the FLASH370 CPLDs were suitable for use in PCI bus applications, the FLASH370i CPLDs are fully compliant with the electrical requirements of the PCI specification over all operating temperatures and voltages. The bus-hold structures eliminate the need for connecting unused inputs and I/O pins to a supply by providing an improved version of an internal pull-up resistor. The bus-hold structure provides a weak internal pull-up or pull-down, depending upon the last driven state on the pin, and it is initialized at power-up to provide a weak pull-up.

## Overview of the FLASH370i ISR Features

The FLASH370i ISR capability is provided through a four-pin JTAG interface comprised of a clock (SCLK), a mode select (SMODE), a serial data in (SDI), and a serial data out (SDO). Additionally, there is a fifth pin, ISRVPP, that provides the high voltage required for programming and erasing the flash programmable portion of the CPLD.

The ISR interface is provided in one of two modes. These two modes are termed "single function" and "dual function." The basic difference between these two modes is that the ISR interface pins have either one function or two functions. *Table 1* shows the FLASH370i devices and the function mode for each.

**Table 1. FLASH370i Device Function Modes**

| Device | # Macrocells | # Pins | Function Mode | |
|---|---|---|---|---|
| CY7C371i | 32 | 44 | – | dual |
| CY7C372i | 64 | 44 | – | dual |
| CY7C373i | 64 | 84 | – | dual |
| CY7C373i | 64 | 100 | single | – |
| CY7C374i | 128 | 84 | – | dual |
| CY7C374i | 128 | 100 | single | – |
| CY7C375i | 128 | 160 | – | dual |

For the CY7C373i and CY7C374i in 100-pin packages, the four basic ISR interface pins, SCLK, SMODE, SDI, and SDO, are assigned to pins that were previously no-connects on the CY7C374. The ISRVPP pin is also assigned to a CY7C374 no-connect pin. This is the simpler ISR interface, and since every pin only has a single function, these are referred to as the single-function FLASH370i devices.

All other FLASH370i CPLDs are dual-function mode devices. As shown in *Table 1*, this includes the CY7C371i, CY7C372i, and CY7C375i devices, and also the CY7C373i and CY7C374i devices in 84-pin packages.

The single-function solution also provides extra diagnostic capability as compared to the dual-function devices. For all FLASH370i ISR devices, when the ISRVPP pin is at high voltage, the full capability to program, erase, read, or verify the device is provided. For single-function FLASH370i devices, reading and verifying the device is provided even when the ISRVPP pin is not at a high voltage. This allows diagnostic software to have normal, operational voltage access to the silicon ID, user ID, bypass mode, and complete programmed pattern when using the single-function devices.

Cascading and program security are also features of the FLASH370i ISR family of CPLDs. The FLASH370i ISR interface provides unlimited cascading capability, as FLASH370i CPLDs can be cascaded homogeneously or with other devices that support the IEEE 1149.1 JTAG interface. FLASH370i devices are cascaded by connecting the SDO pin of one device to the SDI pin of the next device, while also connecting the SCLK, SMODE, and ISRVPP pins of all devices in parallel, just as provided in the IEEE 1149.1 specification. Also, a security bit is included and can be programmed if desired to disallow the reading or verifying of a FLASH370i device, thus protecting the programmable device's contents from being discovered.

## FLASH370i Pin Descriptions

*Tables 2, 3, 4*, and *5* describe the functionality of all of the pins for the FLASH370i devices. *Tables 2* and *3* are for the dual-function FLASH370i devices, and *Tables 4* and *5* are for the single-function devices. *Tables 2* and *3* describe the pin functionality when the ISRVPP pin is at high voltage, i.e., 12V. *Tables 4* and *5* give the pin functionality for the ISRVPP pin being in the normal operating range of 0V to 5V.

**Table 2. Dual-Function Pin Descriptions (ISRVPP = 12V; ISR Interface Enabled)**

| Pin | Mode | Description |
|---|---|---|
| ISRVPP | Input | Provides the high voltage to the programming circuitry and is used to differentiate the function of the other pins; the bus-hold structure is connected |
| SCLK (I/O) | Input | I/O three-stated with the bus-hold structure connected; functions as SCLK input for the ISR interface |
| SMODE (I/O) | Input | I/O three-stated with the bus-hold structure connected; functions as SMODE input for the ISR interface |
| SDI (I/O) | Input | I/O three-stated with the bus-hold structure connected; functions as SDI input for the ISR interface |
| SDO (I/O) | Output | Functions as SDO output for the ISR interface; the output is enabled when the JTAG state machine is in the Shift-IR or Shift-DR states and is three-stated in the other states as defined in the 1149.1 spec; the bus-hold structure is connected |
| Other I/Os | High-Z | Output three-stated with the bus-hold structure connected |
| Other Inputs | High-Z | Unused with the bus-hold structure connected |
| Other I/CLKs | High-Z | Unused with the bus-hold structure connected |

**Table 3. Dual-Function Pin Descriptions (ISRVPP not 12V; ISR Interface Disabled)**

| Pin | Mode | Description |
|---|---|---|
| I/O (SCLK)<br>I/O (SMODE)<br>I/O (SDI)<br>I/O (SDO) | I/O<br>I/O<br>I/O<br>I/O | Normal I/O function; each of these pins functions according to how the device is programmed; the bus-hold structure is connected; the JTAG state machine returns to the "Test-Logic-Reset" state when ISRVPP is in the normal operating range and does not interfere with the normal operation of the device |
| Other I/Os | I/O | Normal I/O function; each of these pins functions according to how the device is programmed; the bus-hold structure is connected |
| Other Inputs | Input | Normal input function; the bus-hold structure is connected |
| Other I/CLKs | I/CLK | Normal input/clock function; the bus-hold structure is connected |

**Table 4. Single-Function Pin Descriptions (ISRVPP = 12V; ISR Enabled)**

| Pin | Mode | Description |
|---|---|---|
| ISRVPP | Input | Provides the high voltage to the programming circuitry to enable all ISR operations; the JTAG state machine is completely operational independent of the state of the ISRVPP pin |
| SCLK | Input | Functions as SCLK input for the ISR interface; the bus-hold structure is connected |
| SMODE | Input | Functions as SMODE input for the ISR interface; the bus-hold structure is connected |
| SDI | Input | Functions as SDI input for the ISR interface; the bus-hold structure is connected |
| SDO | Output | Functions as SDO output for the ISR interface; the output is enabled when the JTAG state machine is in the Shift-IR or Shift-DR states as defined in the 1149.1 spec; the bus-hold structure is not connected |
| All I/Os | High-Z | Output three-stated with the bus-hold structure connected |
| All Inputs | High-Z | Unused with the bus-hold structure connected |
| All I/CLKs | High-Z | Unused with the bus-hold structure connected |

**Table 5. Single-Function Pin Descriptions (ISRVPP not 12V; ISR Programming Disabled)**

| Pin | Mode | Description |
|---|---|---|
| ISRVPP | Input | All ISR operations except for the programming and erasing functions are enabled |
| SCLK | Input | Functions as SCLK input for the ISR interface; the bus-hold structure is connected |
| SMODE | Input | Functions as SMODE input for the ISR interface; the bus-hold structure is connected |
| SDI | Input | Functions as SDI input for the ISR interface; the bus-hold structure is connected |
| SDO | Output | Functions as SDO output for the ISR interface; the output is enabled when the JTAG state machine is in the Shift-IR or Shift-DR states as defined in the 1149.1 spec; the bus-hold structure is not connected |
| All I/Os | I/O | Normal I/O function; the bus-hold structure is connected |
| All Inputs | Input | Normal input function; the bus-hold structure is connected |
| All I/CLKs | I/CLK | Normal input/clock function; the bus-hold structure is connected |

## FLASH370i Pinouts

As previously described, there are single-function and dual-function FLASH370i ISR devices. The CY7C373i and CY7C374i in 100-pin packages are the single-function FLASH370i ISR solutions, and the ISR pin assignments for these are shown in *Table 6*.

**Table 6. ISR Pins for the 100-Pin CY7C373i and CY7C374i**

| ISR Pin | Normal Mode | TQFP Pin # |
|---|---|---|
| ISRVPP | N/A | 88 |
| SCLK | SCLK | 1 |
| SMODE | SMODE | 26 |
| SDI | SDI | 75 |
| SDO | SDO | 50 |

All other FLASH370i ISR devices besides the 100-pin CY7C373i and CY7C374i are dual-function ISR solutions. The pin assignments for all of these devices are shown in *Tables 7, 8,* and *9*. *Table 7* shows the ISR pin assignments for the CY7C373i and CY7C374i in 84-pin packages. *Table 8* shows the ISR pin assignments for the CY7C375i, which has 160 total pins. *Table 9* shows the ISR pin assignments for the CY7C371i and CY7C372i in 44-pin packages

**Table 7. ISR Pins for the 84-Pin CY7C373i and CY7C374i**

| ISR Pin | Normal Mode | PLCC Pin # | CLCC Pin # | PGA Pin # |
|---|---|---|---|---|
| ISRVPP | N/A | 83 | 83 | C7 |
| SCLK | I/O 10 | 14 | 14 | C1 |
| SMODE | I/O 26 | 35 | 35 | L3 |
| SDI | I/O 54 | 72 | 72 | C10 |
| SDO | I/O 38 | 51 | 51 | K9 |

Note: The CY7C373i is not available in the CLCC or PGA packages.

**Table 8. ISR Pins for the 160-Pin CY7C375i**

| ISR Pin | Normal Mode | TQFP Pin # | CQFP Pin # | PGA Pin # |
|---|---|---|---|---|
| ISRVPP | N/A | 139 | 139 | N8 |
| SCLK | I/O 20 | 6 | 6 | M14 |
| SMODE | I/O 52 | 46 | 46 | B12 |
| SDI | I/O 108 | 116 | 116 | N2 |
| SDO | I/O 76 | 76 | 76 | C4 |

**Table 9. ISR Pins for the 44-Pin CY7C371i and CY7C372i**

| ISR Pin | Normal Mode | PLCC Pin # | TQFP Pin # |
|---|---|---|---|
| ISRVPP | N/A | 11 | 5 |
| SCLK | I/O 5 | 7 | 1 |
| SMODE | I/O 13 | 19 | 13 |
| SDI | I/O 27 | 39 | 22 |
| SDO | I/O 19 | 27 | 21 |

Note: The CY7C372i is not available in the TQFP package.

## Overview of ISR Programming Options

There are four programming options available for FLASH370i devices. The first is to use a PC with the FLASH370i ISR programming cable and software. With this method, the ISR pins of the FLASH370i devices are routed to a connector at the edge of the printed circuit board. The ISR programming cable is then connected between the parallel port of the PC and this connector. A simple configuration file instructs the ISR software of the programming operations to be performed on each of the FLASH370i devices in the system. The ISR software then automatically completes all of the necessary data manipulations required to accomplish the programming, reading, verifying, and other ISR functions that are supported.

The second method for programming FLASH370i devices is on an ATE (automatic test equipment). If this method is selected, the FLASH370i ISR software generates programming test vectors for the target tester. This functionality will be available in the second half of 1997.

The third programming option for FLASH370i devices is to utilize the embedded controller or processor that already exists in the electronic system. The FLASH370i ISR software assists in this method by converting the device JEDEC maps into the ISR serial stream that contains the ISR instruction information and the addresses and data of locations to be programmed. The embedded controller then simply directs this ISR stream to the chain of FLASH370i devices to complete the desired reconfiguring or diagnostic operations. An application note describing this will be available mid 1997.

The fourth method for programming FLASH370i devices is to use the same parallel programmer that is currently being used to program FLASH370 devices. Since the programming algorithms are the same, normal third party programming support for the FLASH370i family is readily available.

## FLASH370i ISR Programming Interface

Although detailed knowledge of the ISR internal architecture is not required for users to design with or program FLASH370i devices, the basics are provided here for those interested. The IEEE 1149.1 specification, or JTAG (Joint Test Action Group), requires that the finite state machine shown in *Figure 1* be implemented for controlling the sequence of operations. Transitions are made on the rising edge of SCLK, and the SMODE input determines the next state.

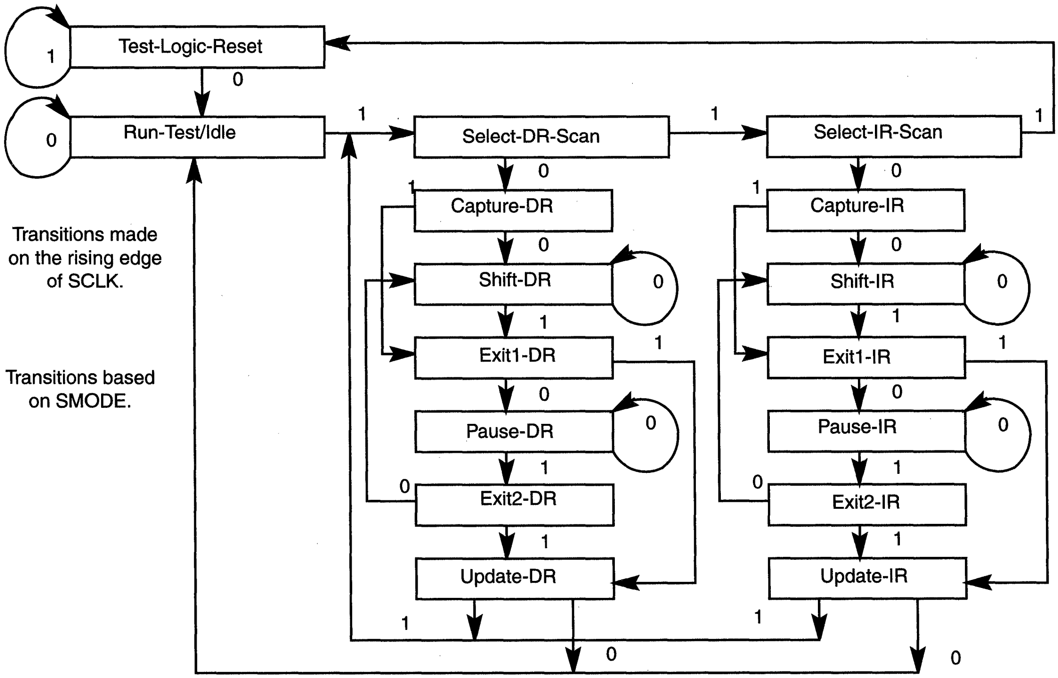


**Figure 1. JTAG State Diagram**

*Figure 2* shows the ISR internal registers that are linked between the serial data input (SDI) and the serial data output (SDO). There are five registers: the Byte Register (BR), the Address Register (AR), the Fast Verify Register (FVR), the Bypass Register (BPR), and the Instruction Register (IR).

The 4-bit instruction register holds the instruction to be executed. The supported instructions and their codes are shown in *Table 10*. The execution of each instruction is beyond the scope of this application note, but the instructions are presented to indicate that the codes are consistent with the IEEE 1149.1 specification, which requires that the EXTEST and BYPASS be assigned the codes with all 0s and 1s, respectively.

The address register holds the address of the memory location to be programmed, read, or verified. As indicated in *Figure 2*, there is an additional routing block that allows the address register and the byte register to be concatenated during shifting. This reduces the time required to complete some of the programming operations, but is transparent to the user.

The byte register is used for multiple purposes. The first is to shift in and hold the data that is used for programming a memory location. The second use is for holding and shifting the memory location data during a read operation. The third use of the byte register is to hold and shift the data when executing the read silicon ID instruction. The fourth use is to hold and shift the data when executing the read user ID instruction.

**Table 10. FLASH370i ISR Instruction Set**

| Code | Instruction |
|---|---|
| 0000 | (Reserved for EXTEST) |
| 0100 | Read Location (Shift AR&BR) |
| 0101 | Program / Shift AR&BR |
| 0111 | Program / Shift AR |
| 1000 | Initiate Power-On-Reset |
| 1001 | Read Silicon ID (Shift BR) |
| 1010 | Program / Fast Verify (Shift FVR) |
| 1100 | Erase Device |
| 1101 | Program / Shift BR |
| 1111 | BYPASS |

The fast verify register function enhances the programming and verification process. During portions of the programming flow, a fast verify operation compares the addressed memory byte with the contents of the byte register, and a single bit is loaded into FVR to indicate the outcome of the comparison. This single bit is then shifted out of the FLASH370i device.

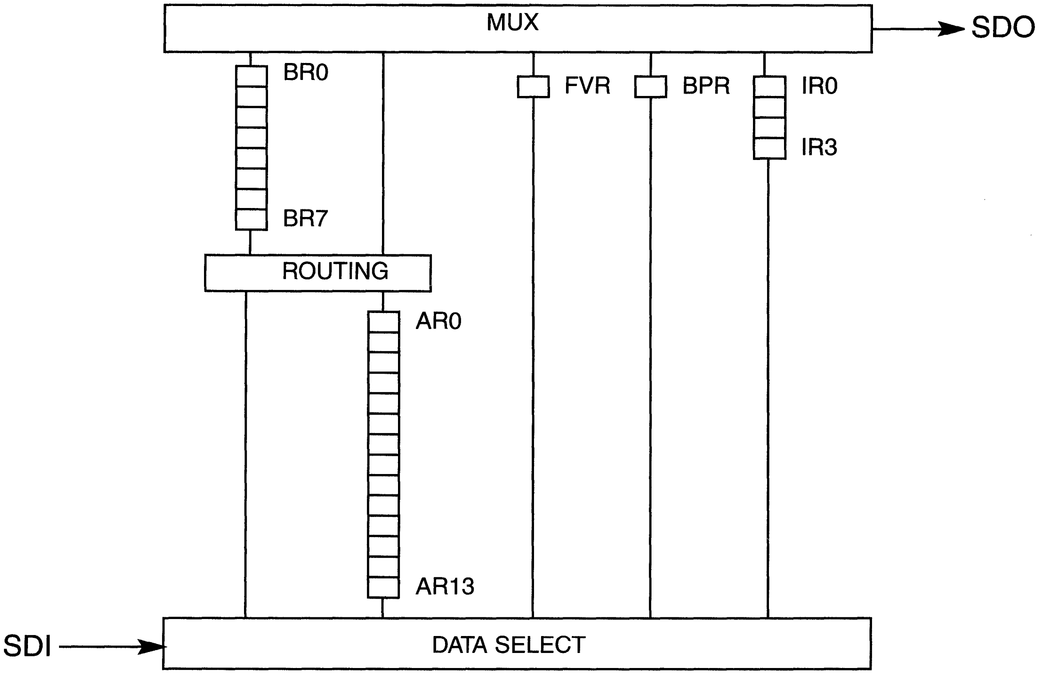


**Figure 2. FLASH370i ISR Interface Registers**

The single-bit bypass register is used for implementing the bypass mode. This is a requirement of the IEEE 1149.1 specification and helps the entire system perform operations more quickly than otherwise would be possible.

## FLASH370i ISR Kit

The FLASH370i ISR kit comes complete with the following items: the ISR User's Guide, the ISR PC programming cable, the ISR software, and all ISR application notes.

The ISR User's Guide contains the documentation of how to use the ISR software for the chosen programming method. Also included in the ISR User's Guide is a list of error messages that the ISR software generates if problems are encountered.

The ISR programming cable provided in the FLASH370i ISR kit is used to facilitate programming from a PC. The ISR programming cable is connected between the parallel port of the PC and the connector at the edge of the printed circuit board which contains the FLASH370i devices.

The ISR software is used for programming FLASH370i devices in all serial programming methods: PC, ATE, and embedded controller.

## Other FLASH370i ISR Application Notes

Additionally, the FLASH370i ISR kit contains this application note and all other ISR application notes to assist designers in utilizing the FLASH370i CPLDs properly. A brief introduction to each of these application notes is provided here to help direct designers to where additional design information can be found.

"Designing with FLASH370i for PC Cable Programming." This application note provides detailed design information on utilizing single-function mode and dual-function mode FLASH370i devices. Utilizing a dual-function mode device in the single-function mode, guidelines for routing the ISRVPP signal on the printed circuit board, and other board design issues are only part of the PC cable programming application note.

"Cascading FLASH370i Devices." FLASH370i devices can be cascaded not only with each other, but with other devices that support the IEEE 1149.1 JTAG interface. This is independent of whether or not the other devices are programmable. This application note provides insight into how simple and complex cascading is accomplished.

"Designing with FLASH370i for Board Tester Programming." This application note is similar to the PC cable programming application note. This board tester application note discusses the additional design issues related to programming a FLASH370i ISR device using an ATE.

"Designing with FLASH370i for Embedded Controller Programming." This application note is also similar to the PC cable programming application note. This embedded controller programming application note discusses the specific design issues for using one popular microcontroller to reconfigure FLASH370i devices, and how this information applies to other microcontrollers and microprocessors is also presented.

"The Impact of Routability on Using FLASH370i ISR Devices for Debugging and Making Field Upgrades." The intentions for utilizing a reprogrammable device that is permanently connected within an electronic system are sometimes quite large. Being successful when utilizing a reprogrammable device begins with a firm understanding of the features and capabilities of the device's architecture. With this understanding, being able to complete operations like field upgrades without encountering unexpected problems (for example not being able to maintain the previous pin-out) is more likely. This application note provides additional insight into the FLASH370i CPLD architecture to assist designers in gaining the necessary expertise to use ISR effectively.

## Conclusion

This application note provides the introductory information to begin designing with the FLASH370i family of In System Reprogrammable (ISR) CPLDs. The FLASH370i ISR CPLD family is a superset replacement for the popular FLASH370 CPLD family that maintains all of the features and advantages of the FLASH370 CPLD architecture. The FLASH370i family additionally provides the ability for the device to be reprogrammed in the system. More detailed information for designing with the FLASH370i CPLD family is available with the FLASH370i ISR kit, and the appropriate information should be reviewed prior to designing with a FLASH370i device.

CYPRESS

# Designing With FLASH370i™ for PC Cable Programming

## Introduction

This application note presents how to design with the Cypress In System Reprogrammable (ISR™) family of complex PLDs, the FLASH370i™ family, for programming from a PC with the ISR programming cable. The main issues addressed are those related to programming and reprogramming the devices in-system (i.e., while they are soldered onto a printed circuit board). Primary among these is interfacing to the pins used to program the devices, both when those pins are used only for programming and when they are used both for programming and for I/O in normal operating mode.

To address this issue, we categorize designs into three types: designs using devices with single-function pins; designs using devices with dual-function pins used in single-function mode; and designs using devices with dual-function pins used in dual-function mode. We will cover all three cases in this application note. The three cases are explained further below.

**Single-Function Pins / Single-Function Mode:** Some of the FLASH370i devices have pinout/package combinations such that the pins used for programming are single-function only; i.e., they are used as programming pins only. When the device is operating (i.e., not being programmed) they are not in use; they are just extra pins.

**Dual-Function Pins / Dual-Function Mode:** The rest of the devices in the FLASH370i family have pinout/package combinations such that the pins used for programming have dual functionality. They are used as programming pins when the device is being programmed, and they are used as I/Os when the device is in normal operating mode. To use both of these functions, you must design interface logic to isolate programming signals from other devices on the board.

**Dual-Function Pins / Single-Function Mode:** Alternatively, the designer can decide to use these dual-function pins as programming pins only and not connect them as I/Os for normal operation. The design is simpler in this case as no special interface logic is required.

All FLASH370i devices can also be cascaded into a single chain for programming purposes so that one cable and one connector can be used to program all of the FLASH370i devices on a board. We will show a simple example of this in this application note, but this topic is covered in depth in an application note called "Cascading FLASH370i Devices."

In addition to the topic of interfacing to the programming pins, several other topics are covered in this application note. These include:

- handling the 12V signal on the board;
- the state of the FLASH370i devices' I/Os at power-up (and why you need to know about this);
- the state of the ISR programming cable's pins when not programming and the state of the FLASH370i's ISR programming pins when no cable is attached;
- and the structure of the ISR software programming configuration file.

Throughout this application note, assume that the ISR devices are programmed in system by means of the ISR cable that connects the board to a PC. This ISR programming cable is provided by Cypress, and the details of the cable and the connector on the board to which it interfaces are explained in detail in this application note. There are other ways to program or reprogram the parts in-system as well, and many of the topics covered and solutions shown in this application note also apply to those methods.

## The ISR Programming Cable

The pins on a FLASH370i device used for programming are: ISRVPP, SDI, SDO, SMODE, and SCLK. Their names and functions are defined below.

**ISRVPP In System Reprogramming Voltage**

During programming, this pin supplies the device with the voltage needed to program it, which is 12.0V ± 0.6V. During normal operation this pin must be between 0V and 5V.

**SDI Serial Data Input**

During programming, this pin is the serial input to the device.

**SDO Serial Data Output**

During programming, this pin is the serial output from the device.

**SCLK Serial Clock**

During programming, this pin is the clock input. SDI and SMODE are sampled on the rising edge of SCLK, while SDO changes following the falling edge of SCLK.

**SMODE Serial Mode Control**

During programming, this is the mode select control input that directs the TAP (test access port) controller state machine contained within the ISR interface.

The FLASH370i devices are programmed using a PC as shown in *Figure 1*: the ISR programming cable connects the parallel port of the PC to a cable header on the board on which the FLASH370i devices are soldered. The header on the board connects to the traces that go to the ISRVPP, SDI, SDO, SCLK, and SMODE pins on the FLASH370i devices themselves. The ISR software runs on the PC and drives these pins on the board, through the cable, header, and traces, to program the devices with the appropriate JEDEC files. The cable has a DC/DC converter built into it that receives 5V from the board and supplies the 12V programming voltage to the ISRVPP pin.

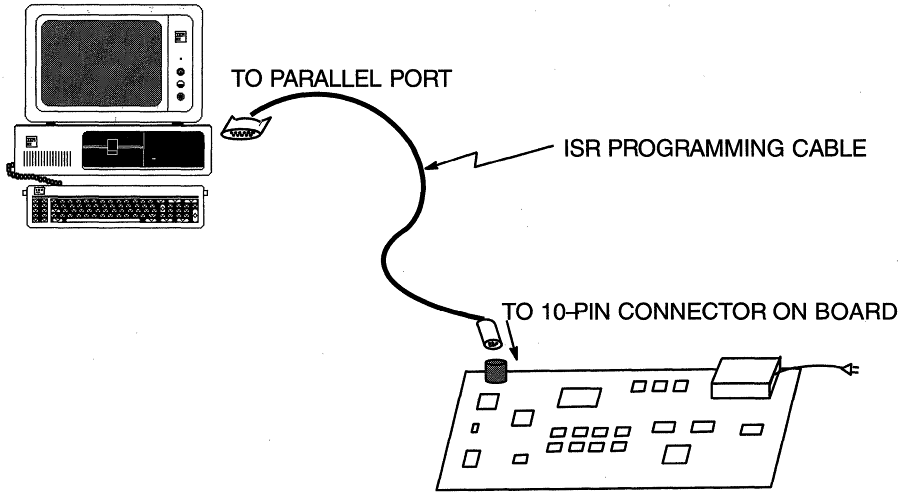


**Figure 1. ISR Programming Cable**

A 10-pin, 2 x 5, boxed header connector is used on the board for the ISR programming cable to plug into. This boxed header connector has a small opening in the box on one side (the key) that allows the ISR programming cable to be plugged in one way only. The pins are on 0.100" centers. The length of each pin is 0.230", and the cross-section is 0.025" x 0.025". This boxed header connector is available as a straight-pin connector and as a right-angle connector.

Additionally, an open header can be used. Part numbers for two compatible connectors are:

DIGI-KEY part # S2012-05-ND
(straight–pin connector)

DIGI-KEY part # S2112-05-ND
(right–angle connector)

The ISR programming cable provided by Cypress has a female end which plugs into these connectors. The position of the signal pins on the connector is shown in *Figure 2.*

| | | | | |
|---|---|---|---|---|
| SMODE | SCLK | SDI | NC | GND |
| GND | ISRVPP | ISR* | VCC | SDO |

**Figure 2. Layout of Connector for Cable on Board, Top View**

To program a FLASH370i device using the ISR programming cable described here, all you need to do is route the ISRVPP, SDI, SDO, SCLK, and SMODE pins from the cable connector to the ISRVPP, SDI, SDO, SCLK, and SMODE pins of the FLASH370i device, respectively.

In addition to these programming pins, there is an additional signal available from the cable called ISR*. The purpose of this signal is to allow the user to know the state of the signals coming from the cable. If ISR* is a logic ‘0’, it indicates that ISRVPP is 12V and one or more FLASH370i devices on the board are being programmed; if ISR* is a logic ‘1’, it indicates ISRVPP is not 12V and no FLASH370i devices on the board are being programmed. This is particularly useful in the “Dual-Function Pins/Dual-Function Mode” designs mentioned in the introduction and explained in detail later in this application note. The logic ‘0’ and logic ‘1’ levels on ISR* are 0V and 5V, respectively. The logic ‘0’ level is driven actively while the logic ‘1’ level is not driven but pulled up using a 20-kΩ pull-up resistor to the VCC pin of the ISR connector.

There are three other connection points on the cable and cable header: VCC, GND, and NC. VCC is the pin through which the VCC plane on the board containing the FLASH370i devices supplies 5V to the DC/DC converter in the ISR cable. This is necessary for the ISR programming cable to be able to generate the 12V voltage level on ISRVPP needed to program the FLASH370i devices. GND provides a common ground reference between the board and the ISR programming cable. NC is a no connect that is not used.

## Designs That Use Devices With Single-Function Programming Pins

Single-function programming pins refers to the case where the pins used for programming the device are only dedicated to that function and are extras—not in use—when the device is in normal operating mode. Two members of the FLASH370i family currently have this feature, the 64-macrocell CY7C373i in the 100-pin TQFP package and the 128-macrocell CY7C374i in the 100-pin TQFP package. These two devices, which have identical pinout, have 64 I/O pins and 6 input-only pins for a total of 70 I/Os. Therefore, in the 100-pin package, there are still plenty of pins to accommodate the five ISR pins on separate pins and still have ample power and ground connections. As the pinout diagram of these two devices in *Figure 3* shows, pins 88, 75, 50, 1, and 26 are, ISRVPP, SDI, SDO, SCLK, and SMODE, respectively.

Designing with these devices is the easiest of the three cases. You simply connect these ISR pins to the corresponding pins on the ISR programming cable connector, and you now have access to in system reprogramming.

There are other issues related to this, such as the best way to route the 12V signal to the ISRVPP pin, the state of the ISR programming pins when the cable is disconnected, and the format of the ISR software configuration file. These issues apply to all three types of designs (single-function pins, dual-function pins used in single-function mode, and dual-function pins used in dual-function mode), and will be treated in a later section.

## Designs That Use Devices With Dual-Function Programming Pins

Dual-function programming pins refers to the case where the ISR pins on a FLASH370i device are used for programming the device when it is in programming mode and are used as normal I/Os in normal operating mode. The devices that have this characteristic are the CY7C371i and CY7C372i in 44-pin packages, the CY7C373i and CY7C374i in 84-pin packages, and the CY7C375i in 160-pin packages-in other words, all of the devices in the family except the two mentioned in the single-function section above.

For example, look at the CY7C373i and CY7C374i 84-pin PLCC pinout in *Figure 4*. These are similar to the devices described in the single-function section above, but with 64 and 128 macrocells respectively, and with 64 I/O pins. In this case, the programming functions share pins with normal I/O functions. Pin 72 is SDI in programming mode and I/O54 in normal operating mode; pin 51 is SDO in programming mode and I/O38 in normal operating mode; pin 14 is SCLK in programming mode and I/O10 in normal operating mode; and pin 35 is SMODE in programming mode and I/O26 in normal operating mode. Pin 83 is ISRVPP and differentiates the function of these other four ISR interface pins.

There are two ways to design with devices that have dual-function programming pins. First, you could use the dual-function pins as single function pins. That is, you could decide to use only the programming function of the pins and not use those pins as I/Os in your design. The other way to use them, of course, is to use them as true dual-function pins, where you use them both as programming pins in programming mode and as I/Os in normal operating mode.

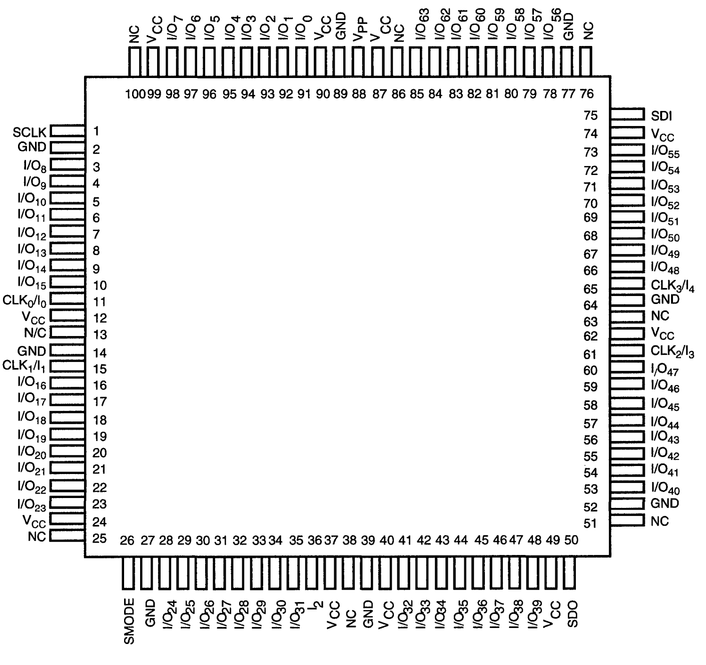


**Figure 3. Pinout of CY7C373/4i in 100-Pin TQFP**

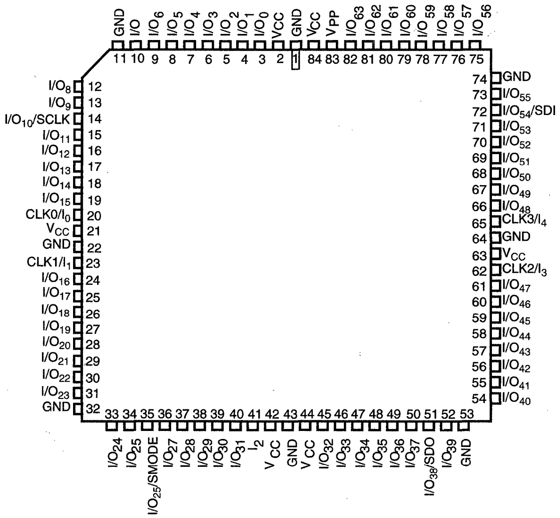


**Figure 4. Pinout of CY7C373/4i in 84-Pin PLCC**

### Devices With Dual-Function Programming Pins Used in Single-Function Mode

To use the FLASH370i devices in this way, with the dual-function pins used as programming pins only, you need the total number of I/Os used in your design to be equal to or less than (n–4), where n is the total number of input and I/O pins available on the device. This way is the preferred method of design. It is much easier and will save both time and components over implementing the kind of logic described in the next section for dual-function pins used in dual-function mode.

To design with the dual-function pins used in single-function mode, all you really need to do is make sure no I/Os get assigned to those dual function pins. Two ways to do this are described below.

First, if you are using the Cypress *Warp*™ VHDL compiler, you can use a simple synthesis directive called "pin_avoid" to make sure the compiler does not assign signals to whatever pins you specify. In this case, of course, you would specify the dual function pins. An example of the exact text to include in your VHDL code appears in *Figure 5.* This example assumes you are using the CY7C373i or CY7C374i in the 84-pin PLCC package where pins 14, 35, 51, 72, and 83 are the ISR pins.

If you prefer, or if the software you are using does not have a capability similar to the "pin_avoid" directive in *Warp*, you can also ensure the dual-function pins do not get used as I/Os in normal operating mode by explicitly assigning all of the signals to pins in your design. You just need to make sure you assign all of the signals to pins other than the dual-function pins. An example showing how to do this in *Warp* using the "pin_numbers" directive is shown in *Figure 6.* Again, this example assumes you are using the CY7C373i or CY7C374i in the 84-pin PLCC package, so pins 14, 35, 51, 72, and 83 are not being used. Notice that none of the signals used in the example in *Figure 6* are assigned to these pins.

```
-- example of using "pin_avoid" for single-function mode of
-- dual-function devices
entity cpuctl is port (
     a           :     in       bit_vector (31 downto 0);
     rd, wr      :     out      bit;
     hold        :     buffer   bit;
     status      :     out      bit_vector (7 downto 0));
attribute pin_avoid of cpuctl:entity is "14 35 51 72 83";
end cpuctl;
-- architecture would follow
```

**Figure 5. VHDL Code Fragment Showing pin_avoid Attribute**

```
-- example of explicit pin assignments that avoid ISR pins
-- to facilitate single-function mode of dual-function devices
entity cpuctl is port (
     a            :     in        bit_vector (15 downto 0);
     rd, wr       :     out       bit;
     hold         :     buffer    bit;
     status       :     out       bit_vector (7 downto 0));
-- assign pins below and avoid pins 14, 35, 51, 72, and 83
attribute pin_numbers of cpuctl:entity is
"a(15):12 a(14):13 a(13):15 a(12):16 a(11):17 a(10):18 a(9):19 " &
"a(8):24  a(7):25  a(6):26  a(5):27  a(4):28  a(3):29  a(2):30 " &
"a(1):31 a(0):33 rd:36 wr:37 hold:38 status(7):54 status(6):55 " &
"status(5):56 status(4):57 status(3):58 status(2):59 status(1):60 " &
"status(0):61";
end cpuctl;
-- architecture would follow
```

**Figure 6. VHDL Code Fragment Showing pin_numbers Attribute**

### Devices With Dual-Function Programming Pins Used in Dual-Function Mode

There are cases where you may need or want to take advantage of the dual functionality of the dual-function programming pins. For example, you may not have enough I/O pins for your design if you do not use the dual-function ISR programming pins as I/Os when the device is in normal operation. Other times, you may want to use the dual-function ISR pins as I/Os in normal operation because their physical position makes your board layout easier. If you want to do this in your design, you can do it fairly easily; it simply requires a little bit of extra logic and possibly some additional small components. This section shows you how to do this.

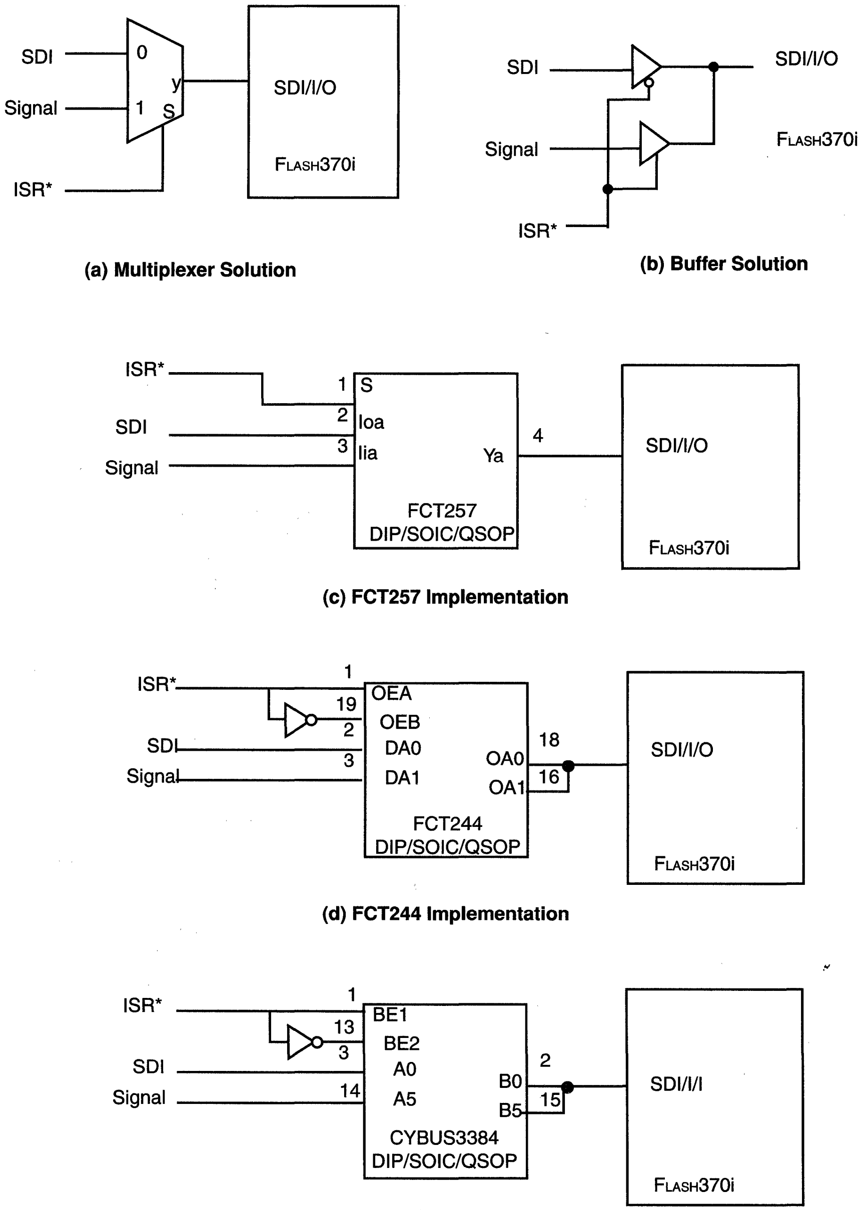
The SDI, SCLK, and SMODE programming pins are all inputs to the device during programming, and they always share pins with bidirectional I/Os when they are dual-function pins. The SDO programming pin, on the other hand, is an output pin from the device during programming. It, too, always shares a pin with a bidirectional I/O when it is a dual-function pin. These I/O pins, in turn, can be used as input only, output only, or bidirectional I/Os in any design, based upon the functionality that is described for these pins in the programmable logic chip's design description. The result is that there are six different cases to consider: you can have an ISR input programming pin (SDI, SCLK, SMODE) sharing a pin with a signal that is an input, an output, or an I/O; and you can have an ISR output programming pin (SDO) sharing a pin with a signal that is an input, an output, or an I/O. We next look at each of these six cases individually.

This approach can be more time-consuming than using the "pin_avoid" directive, especially if your design has a large number of I/Os. When you do this, you also need to take some device-specific resource information into account, such as block-reset and preset or half-block output-enable signal allocation. Since the compiler can account for all of this for you automatically, it is usually easier to just use the "pin_avoid" directive.

What you are trying to accomplish in all of these cases is fundamentally the same. You are trying to isolate the programming signals from the normal operating signals on the board. You do not want the programming signal to drive or affect anything else on the board when you are programming the FLASH370i device, and you do not want the normal operating signal to drive, affect, or be affected by the programming logic when the FLASH370i device is operating normally in the system. The basic strategy in all of the cases listed above is to use three-state buffers or multiplexers on these signals, and to have those buffers or multiplexers controlled by the ISR* signal from the programming cable. The ISR* signal, recall, is a signal from the programming cable that is a logic '0' when ISRVPP is 12V (when the FLASH370i device is being programmed), and it is a logic '1' when ISRVPP is not 12V (when the FLASH370i device is not being programmed).

First, consider the case of the ISR programming pins that are inputs to the device during programming, SDI, SCLK, and SMODE. When one of these device pins is being used as an input during normal operating mode, you simply have to select between one of two inputs based on whether you are in programming mode or in operating mode. This is implemented very easily by using a 2:1 multiplexer where ISR* is the select line, as shown in *Figure 7*(a). Alternatively, you could implement this by having two three-state buffers whose inputs are SDI (or SMODE or SCLK) and *signal*, whose outputs are tied together and to the SDI / I/O pin, and whose enable lines are controlled by opposite values of ISR*. This is shown in *Figure 7(b)*. One way you could implement this logic is with FCT-family devices. For example, you could use one of the four 2:1 multiplexers in a CY74FCT257 to implement the logic shown in *Figure 7(a)*. Alternatively, you could use a pair of transceivers or pass-transistors from a CY74FCT244 or CYBUS3384 to implement the logic shown in *Figure 7(b)*. The connections for the FCT257, FCT244, and CYBUS3384 are shown in *Figure 7(c)*, *(d)*, and *(e)*, respectively.

The inverter shown in *Figure 7(e)* can be eliminated by implementing an inversion within the CYBUS3384 device. This requires using only an external resistor and a few additional connections. An inversion of the connection to pin BE1* is accomplished by connecting +5V to pin A1 and connection one end of a resistor to GND and the other end to pin B1. B1 will then be the inverse of the input connected to BE1*, which is ISR*. By implementing this inversion, the inverter in *Figure 7(e)* can be removed, and pin B1 can be connected to pin BE2*. The BE1*, A0, A5, B0, and B5 pin connections remain the same.



**(a) Multiplexer Solution**

**(b) Buffer Solution**

**(c) FCT257 Implementation**

**(d) FCT244 Implementation**

**(e) CYBUS3384 Implementation**

**Figure 7. Design for Dual-Function Pins: SDI/SCLK/SMODE used with Input**

The FCT devices shown are just one possible way of implementing this logic, of course. There are others, including using extra pins and gates from an ASIC, FPGA, CPLD, or PAL® device already on the board. Regardless of whether the buffer or multiplexer is in an FCT device, ASIC, FPGA, or other device, there will be some additional propagation delay for the normal operating signal due to the presence of that logic. This must be accounted for in your design, and using the CYBUS3384 provides the smallest extra delay. The issue of extra delay holds true for the other cases presented next.

In the case of the SDI, SCLK, or SMODE sharing a pin with an I/O that is used only as an output during normal operating mode, the logic is slightly different. Much like the case above, you can just use a pair of three-state buffers or pass-transistors to separate the signals that are used for the two different functions. In this case, however, instead of tying the two outputs together, you tie the output of one buffer both to the dual-function pin of the FLASH370i device and to the input of the other buffer. The input to the first buffer is the programming function signal, and the output from the other buffer is the normal operation output *Signal*. The first buffer is enabled when ISR* is asserted and is disabled otherwise, and the second buffer is enabled when ISR* is deasserted and is disabled otherwise. This is shown in *Figure 8*. Thus, when the device is being programmed, SDI (or SMODE or SCLK) is driving the SDI / I/O pin and *Signal* is in the high-impedance state, and when the device is not being programmed, the SDI / I/O pin is not driven as an input allowing the FLASH370i output to drive *Signal*. Because *Signal* is in the high-impedance state during programming, you may need to have a pull-up or pull-down resistor on *Signal* depending on how you use it on your board.

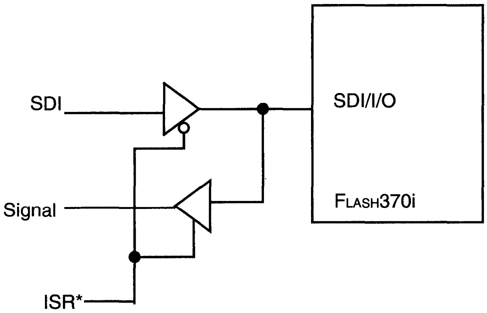


**Figure 8. Design for Dual-Function Pins: SDI/SCLK/SMODE used with an Output**

You can also use a CYBUS3384 as an alternative to the buffers, just as in solution 7(e) in the previous case. This would be the most flexible solution because it would work for all configurations—the pin used as an I, O, or I/O—and allows you to decide later exactly how to use that pin.

Another alternative exists for the case when SDI (or SMODE or SCLK) has dual functionality with an I/O pin that is used as an output during normal operating mode. You can simply connect *Signal* and SDI (or SMODE or SCLK) together directly without any isolating gates. Since *Signal* is an output from the FLASH370i in normal operation, no damage should result from *Signal* being driven identically to SDI during ISR programming. The only precaution you must take is to ensure the circuitry that is being driven is not affected logically by the values on SDI, such as a state machine being put into the wrong state. If that circuity can ignore SDI's values during programming, then this would be the preferred solution. It does not add any extra delay to *Signal's* path and does not require any additional devices or logic, thus allowing you to use the three ISR dual-function pins, SDI, SMODE, and SCLK "for free."

In the case of the SDI, SCLK, or SMODE sharing a pin with an I/O that really is used as a bidirectional I/O, the logic needed is a little more complicated. As seen in *Figure 9*, part of the logic is a combination of the two solutions for the two individual cases above in the way it uses ISR* to separate the programming function of SDI (or SMODE or SCLK) from the input and output functions of *Signal* during normal operating mode. There is more than just this logic required, however. It is also necessary to use an extra pair of buffers to separate the input and output functionality of *Signal* itself. This is required to keep from unintentionally building a feedback loop and is accomplished with the help of an extra signal that indicates the direction of the I/O pin. In this example, we assume we have a signal called *dir*, and that *dir* = '1' when the I/O pin is being used as an input and *dir* = '0' when the I/O pin is being used as an output.

To understand why this is necessary, consider just combining the logic from *Figure 7(b)* and *Figure 8*. The result would be the logic shown in *Figure 10*, which is different from *Figure 9* in that buffers *b4* and *b5* were eliminated and intermediate signals *w*, *x*, and *y* are now all simply connected together and to the SDI / I/O pin. In the logic of *Figure 10*, when the FLASH370i device is in normal operation mode and ISR* is HIGH, buffers *b2* and *b3* would both be enabled. If *Signal* were an input at that time, it would drive the input to buffer *b2*, whose output would drive the input to buffer *b3*. The output of buffer *b3* would be driving the input of *b2* again, resulting in a feedback loop that could produce undesired affects. The same thing would happen if *Signal* were an output at the time.

Buffers *b4* and *b5* in *Figure 9* prevent this. In the logic of *Figure 9*, when *Signal* is an output from the FLASH370i device, *b5* is enabled and *b4* is disabled; when *Signal* is an input to the device, *b4* is enabled and *b5* is disabled. In both cases, both the function and value at the pin of the device and the function and value of *Signal* are the same, correct, and only driven by one source. There is no dangerous self-driving feedback system like there is in *Figure 10.*

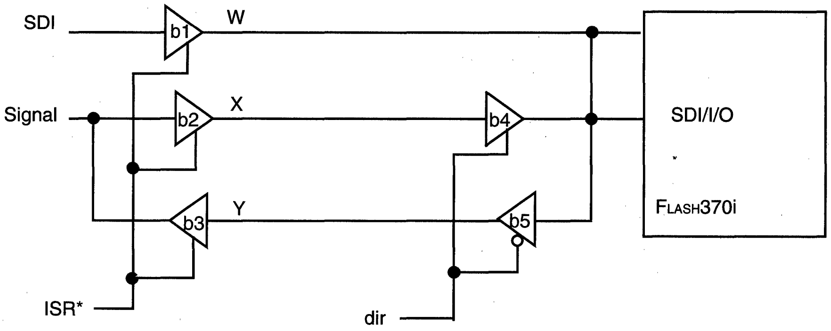


**Figure 9. Design for Dual-Function Pins: SDI/SCLK/SMODE used with an I/O**

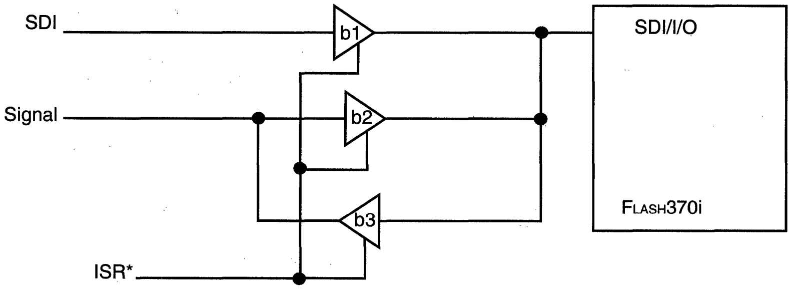


**Figure 10. SDI/SCLK/SMODE used with an I/O: Example of Incorrect Solution**

The limitation of this solution is that it requires the extra signal *dir*. This signal may be already available; in fact, it may be an input to the FLASH370i device itself for use as the $\overline{OE}$-control on the pin in question. If it is not already available, you will need to generate it using other logic on the board. If you cannot do it using other logic on your board, you should certainly be able to generate it using logic inside the FLASH370i device itself, because, as pointed out above, it should be the same signal as the $\overline{OE}$ used on that pin internally. To get the signal out of the FLASH370i, however, requires an additional pin, so if you are using the logic in *Figure 9* to save a pin, having to use a pin on the device to generate *dir* will not gain you anything. If generating one *dir* will help you save two or three pins by allowing you to use two or three of SDI, SCLK, and SMODE as dual-function pins, then you will still have a net savings of one or two pins and it may be worth it.

As was mentioned in the case where the SDI (or SMODE or SCLK) dual-function pin was being used with an input-only pin or with an output-only pin, you can also use the CYBUS3384 solution of *Figure 7(e)* when trying to use the SDI (or SMODE or SCLK) dual-function pin as a bidirectional I/O pin in normal operating mode.

The logic for using the dual-functionality of the SDO / I/O pin is essentially the same as is shown in the above three cases. The only difference is that SDO is an output during programming mode instead of an input. Therefore, the only difference in the logic is the orientation of some of the buffers. The solutions for the SDO case are presented without further explanation. The logic diagram for the case where SDO is connected to an I/O used only as an input is shown in *Figure 11*; the logic diagram for the case where SDO is connected to an I/O used only as an output is shown in *Figure 12*; and, the logic diagram for the case where SDO is connected to an I/O really used as a bidirectional pin is shown in *Figure 13*. You can alternatively use the CYBUS3384 solution presented in *Figure 7(e)* in each of these three cases.

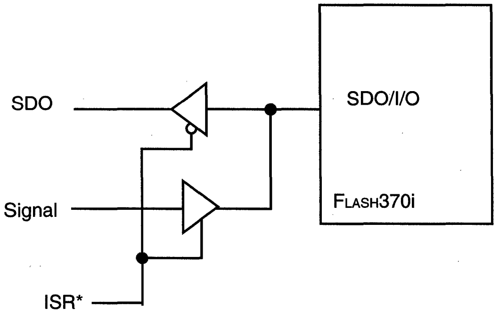


**Figure 11. Design for Dual-Function Pins: SDO used with an Input**

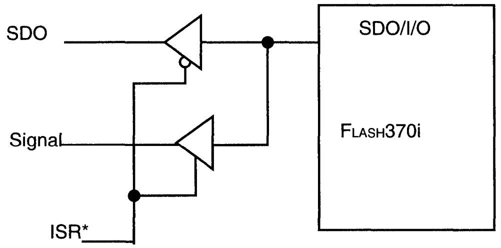


**Figure 12. Design for Dual-Function Pins: SDO used with an Output**

To summarize this section, there are many reasons to use the members of the FLASH370i family that have dual-function ISR programming pins and many different ways to accomplish it. The easiest way is to use the dual-function device in the single-function mode. This uses the dual-function pins as programming pins only, and is easily accomplished using the pin_avoid and pin_numbers directives in your *Warp* design file. There are also going to be cases where you will want to use the dual-functionality, most likely because you need some or all of the four ISR programming pins as inputs, outputs, or I/Os during normal operation to get all the signals you need into and out of the device for your design. The circuits needed to share these pins are relatively straightforward and require nothing or only buffers or pass transistors. These are circuits you can implement using FCT or other logic, or you may be able to implement them using extra gates and pins of an ASIC, FPGA, or another PLD you already have on the board.

## Simple Cascading and the ISR Software Configuration File

Until now, we have talked about programming just a single FLASH370i device in the system. You can cascade an unlimited number of FLASH370i devices in a system. That is, you can daisy-chain the devices together and connect their programming pins in such a way that all of the devices can be programmed from a single connection to the ISR programming cable. To do this, you simply tie all of the SCLK, SMODE, and ISRVPP pins of each device to those same pins, respectively, on all of the other devices, and then connect them to the corresponding pins of the ISR cable connector. You then connect the SDI pin from the cable connector to the SDI pin of the first device in the chain, then connect the SDO output of that device to the SDI input of the next device in the chain, then connect the SDO output of that device to the SDI input of the next device in the chain, and so forth, until you finally connect the SDO output of the last device in the chain to the SDO pin of the cable connector (see *Figure 14*).

We have taken for granted so far that there is ISR programming software running on the PC that accesses the FLASH370i JEDEC files and drives the ISR programming cable to program the FLASH370i devices. This ISR software, which is provided by Cypress, reads a simple configuration file provided by the user. This configuration file is where the user specifies what the JEDEC files are, what the FLASH370i devices to be programmed are, and what other operations, if any, are to be performed. For example, to program a single CY7C374i with a JEDEC file called sarctl.jed, the configuration file would just be the single line shown below:

```
CY7C374i  p c:\sarctl.jed;
```

The first field specifies the FLASH370i device; the second field specifies the operation; and the third field, if necessary, specifies the full path of the name of the file to be used.

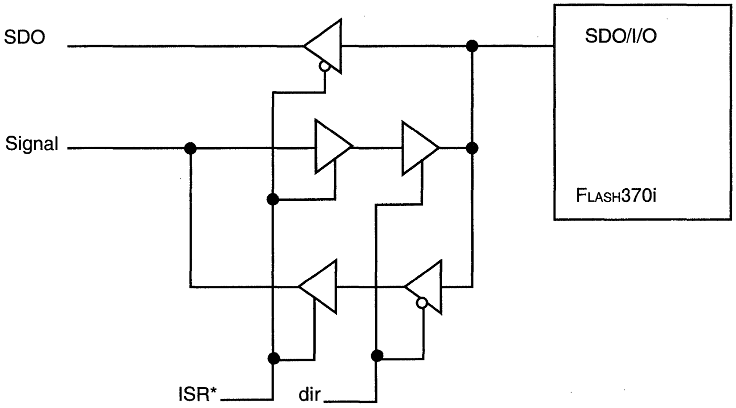


**Figure 13. Design for Dual-Function Pins: SDO used with an I/O**

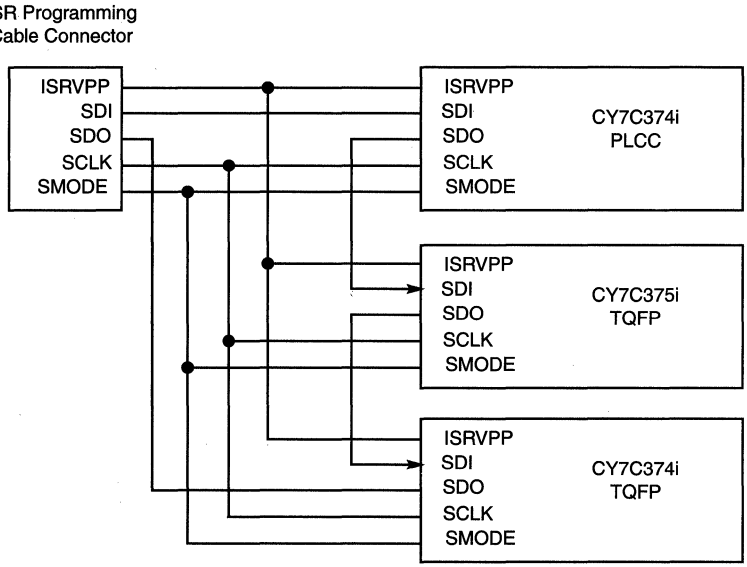


**Figure 14. Simple Cascading Example**

As previously mentioned, multiple FLASH370i devices can be chained together for ISR programming purposes. The configuration file in this case contains all of the devices in that chain, which may also include non-Cypress devices. The first line of such a configuration file corresponds to the first device in the chain, and the last line to the last device in the chain. As is the case in a single-device configuration file like the one described before, the first field in each line specifies the device type; the second field specifies the operation; and the third field, if necessary, specifies the full path of the file name to be used. The full list of operation codes that can be used in the second field along with their meaning is shown below:

| | |
|---|---|
| p | program device with listed JEDEC file |
| v | verify device contents against listed JEDEC file |
| r | read device and create listed JEDEC file |
| e | erase device |
| d | get silicon ID |
| u | get user code |
| n | no operation |
| s | program security bit |
| i | initiate power-on-reset |

A non-Cypress device can be included in the chain with FLASH370i devices if the device has a JTAG interface, contains the standard JTAG TAP controller state machine, and supports a bypass instruction which uses the specified code of all '1's. In the case where the chain of cascaded devices includes a non-Cypress device, the line in the configuration file for that device has the key word "NONCYPRESS" as the first entry and an integer that specifies the number of bits in that device's instruction register in the second field. The JTAG specification defines the bypass instruction to be all '1's, so the ISR programming software will insert the number of '1's necessary to fill the non-Cypress device's instruction register in the appropriate place in the bitstream that is sent.

An example of a multiple-device, multiple-operation configuration file is shown here:

```
CY7C375i    v   c:\dramctl.jed;
CY7C374i    n;
CY7C373i    r   c:\cpuctl.jed;
CY7C374i    p   c:\sarctl.jed;
NONCYPRESS      4;
CY7C372i    u;
```

This configuration file specifies that there are six devices in the cascaded chain, that a CY7C375i is the first in the chain, that a CY7C372i is the last in the chain, and that there is one non-Cypress device in the chain. The first device, a CY7C375i, is to have its JEDEC file read and verified against the file c:\dramctl.jed; the second device, a CY7C374i, is to be passed over—no operation is to be performed; the third device, a CY7C373i, is to have its JEDEC file read and stored in the file c:\cpuctl.jed; the fourth device, a CY7C374i, is to be programmed with the file c:\sarctl.jed; the fifth device is the non-Cypress device, and it has a four-bit instruction register; and the sixth device is a CY7C372i, and the user code already programmed into the device is to be read.

For more detailed information on cascading FLASH370i devices with other JTAG devices for ISR programming and system diagnostics, refer to the application note titled "Cascading FLASH370i Devices."

## Other Considerations

In addition to the topics covered above, there are several other miscellaneous board-level design issues you should consider when using FLASH370i devices. These include the state of the FLASH370i devices at power-up (and why you need to know about this); the state of the FLASH370i programming pins when the ISR programming cable is not attached; and handling the 12V ISR programming signal on the board. We address these considerations here.

### State of the FLASH370i Device I/Os at Power-Up

When FLASH370i devices are shipped from Cypress, the devices have already been programmed, erased, and programmed again as part of the testing process. They will not, therefore, be blank when they first come out of the tube. They will, however, be programmed such that all of the I/Os are three-stated. Furthermore, the I/Os (except SDO) are also all three-stated during device programming, that is, when ISRVPP is at 12V. This allows you to solder FLASH370i devices directly on your board without having to erase them first, and it will allow you to power-up your board and program the FLASH370i devices on it without having to worry about their initial, non-blank state causing any problems.

The most likely way people would want to use these parts is to take FLASH370i devices directly from the tube, solder them onto their board, turn on the power to the board, and then program the devices for the first time using the ISR programming cable connected to a PC. Since many of the I/Os on the FLASH370i device(s) to be programmed will undoubtedly be inputs, other devices on the board could be driving those pins immediately upon powering up the system. By having all of the FLASH370i I/Os initially programmed to be three-stated, and by having them also be guaranteed to be three-stated during ISR programming, you are assured that the FLASH370i device will not be also trying to drive those pins. This prevents bus contention that could otherwise arise, and it prevents the damage to a FLASH370i device or other devices on your board that could result from it. When the 12V ISRVPP signal goes away, the FLASH370i device will then start driving some of its output pins, based upon the FLASH370i device being programmed according to your design. At this point, it is no different from powering up a board with preprogrammed non-ISR PAL devices, PLDs or CPLDs on it.

### State of the FLASH370i's Programming Pins When the ISR Programming Cable is Not Attached

It is likely that the ISR programming cable will not always be plugged into the connector on your board, so it is important to understand that this will not be a problem when the board is powered up and expected to be running. The reason it will not be a problem is that the FLASH370i devices have been designed with bus-hold structures on every input, input/clock, and I/O pin, including the programming pins whether they are single-function or dual-function. The exception to this is the SDO pin on single-function devices, which is an output only. This eliminates the need to use external pull-up resistors or any other technique for handling the case where the ISR programming pins are left floating due to the ISR programming cable being disconnected.

Bus-hold structures enable an I/O pin to maintain its most recent logic value even when it is three-stated, whether that value was being driven in as an input pin or driven out as an output pin. This is done with an extremely weak latch connected to the pin. Since the ISR programming pins have these bus-hold structures, if the ISR programming cable is disconnected when the board is powered on, the ISR programming pins will all maintain a logic '0' or logic '1' value even though they are no longer being driven. These pins will not be floating between these logic levels, and therefore, they will not be subject to oscillation and will not be sourcing or sinking any more current than the bus-hold currents specified in the data sheet. If the ISR programming cable is disconnected when the board is powered-down, or if the board is powered down and then back up after the cable has been disconnected, there is still no problem. The FLASH370i bus-hold structures have been designed to always power-up with a logic '1' level maintained on the pins to emulate an internal pull-up, but this does not interfere with the capability of maintaining the last state when not driving or not being driven.

The advantages of the bus-hold structures apply both to the case of ISR programming pins used as single-function pins and as dual-function pins. In the single-function case, they can be taken advantage of exactly as described above. In the dual-function case, they also behave as described above, and in addition they do not interfere with the normal function of the pin. Once the device is no longer being programmed, the normal function of the pin becomes its main use. If, in that normal function, the I/O is first three-stated, the bus-hold structure operates as described before; if, in that normal function, the I/O is first driven as an output, the bus-hold structure does not interfere; and if, in that normal function, the I/O is first driven as an input, the bus-hold structure holds the last value until a new one is driven.

The bus-hold structures are also useful on normal pins (i.e., non-programming pins). Just as the bus-hold eliminates the need for external pull-up resistors, or any other method for dealing with floating ISR programming pins, they also eliminate the need for dealing with floating non-programming, general-purpose input and I/O pins. This can be especially useful during the debugging of a design where you may want to leave all unused pins on a CPLD unconnected in case you need to add more signals as the design evolves. You can take advantage of the bus-hold structures to make that as easy as possible. You simply do not use those pins when describing and compiling your design, and you then leave those unused pins unconnected on your board. The bus-hold structures then act as pull-ups on these pins, but they are better because there is no external resistor connecting the pin to VCC or GND. This most importantly saves you time. When you need to add a signal to a formerly unused pin, you simply make that connection on your board; you do not also have to break the connection to VCC or GND like you would have to if you used an external pull-up.

Note that the ISR* pin from the ISR programming cable connector does not necessarily connect to a FLASH370i I/O pin. Since it does not, you must use a pull-up on the ISR* signal on your board so that it is not left floating when the ISR programming cable is disconnected.

### Handling the 12V Signal on the Board

There are two requirements on the ISR programming voltage that necessitate special handling. The first is that its voltage must be in the range 11.4V ≤ ISRVPP ≤ 12.6V during programming, and the second is that its maximum current is 40 mA per FLASH370i device during programming. The 5V/12V DC/DC converter and other components in the ISR programming cable described in this application note have been chosen to ensure that these specifications are met. Because of this, if you are using the ISR programming cable for programming the FLASH370i devices, it would be a good idea to just use the 12V supplied by the ISR programming cable for ISRVPP even if 12V is available on the board on which the devices are being used or from the backplane to which that board is connected.

Because of the higher than usual current and voltage requirements on the ISRVPP signal, the trace on the printed-circuit board connecting the ISRVPP pin from the ISR programming cable to the ISRVPP pin on the FLASH370i device(s) also deserves special attention. First, to handle the current, the trace should be double the width of the standard traces. Second, the trace should be kept as short as possible. In general, this means the connector for the ISR programming cable should be place as close as possible to the FLASH370i devices on the board. Since the connector is small, it is much easier to move the connector to be close to the devices than change the whole board layout to place the devices close to the chosen spot for the connector.

## Conclusion

In-system reprogrammability (ISR) in a CPLD has several benefits. It allows engineering development and debugging without having to socket the CPLDs and without having to remove them and reprogram them in a device programmer. This saves time regardless of the package type you decide to use. ISR is especially valuable when you decide to use fine-pitch packages like TQFPs. Like before, it allows you to use them without sockets, which means, again, no handling of devices for reprogramming. Not only does that save time, but in this case, it also avoids the higher potential for bending leads on very fine-leaded devices. Also, by allowing you to solder TQFP packages directly onto a board without sockets, it helps you avoid spending time simply checking device-to-socket-lead connections during debugging. ISR also allows for designs which can be reconfigured in the field, either by a software update or by an input from the system they are in. The superior routability and flexible architecture of the Cypress FLASH370i CPLDs enhance the value of all of these benefits greatly by allowing you to actually make design changes during prototyping, debugging, or field operation and still successfully route to the already-defined pinout, even on designs that are utilizing most or all of many of the device's resources.

This application note shows how to take advantage of the in-system reprogrammability of the FLASH370i family by using a cable connected to the parallel port of a PC for programming the CPLDs on a board. This is typically the way these parts are used during development and debugging, which, as described above, is an area where the FLASH370i architecture and routability are particularly useful. This application note explains all of the details of the programming cable and the signals it uses, and it also covers many design techniques and considerations that show how to most easily use the desired capabilities of these parts. These include logic designs for using the dual-function pins on FLASH370i devices whose programming signals share pins with I/O signals, tips for handling the 12-volt programming signal on your board, and details of the ISR programming software configuration file.

CYPRESS

# Using Hierarchy in VHDL Design

## Introduction

Hierarchical design methodology has been commonly used for quite some time by system designers and software developers. There are two primary advantages to using this methodology. First, it allows commonly-used building blocks to be created separately and saved for later use without having to redesign or reverify them. Second, it allows for more readable design files by keeping the top-level design file as a simple integration of smaller building blocks, either user-defined or from a vendor-supplied library. In system design, these building blocks normally take the form of schematic symbols instantiated into a schematic drawing, while in software they are functions or procedures that are called from the main program.

VHDL includes a set of features specifically designed to make hierarchical design both simple and powerful. This note will first describe these features and then walk through a simple example of how they might be used. It assumes that the reader is familiar with how to create a VHDL design unit consisting of an entity-architecture pair.

## Key Concepts

In order to construct a hierarchical design in VHDL, the designer must understand the concepts of components, packages and libraries.

Component – A component is a VHDL design unit that may be instantiated in other VHDL design units. Before it can be instantiated, it must be declared using the COMPONENT declaration which specifies the name of the component and lists its local signal names.

Package – A package is a collection of VHDL declarations that can be used by other VHDL descriptions. For the purpose of creating hierarchical designs, a package consists of one or more components. However, a package may also include other types of declarations.

Library – A library is a logical storage facility for design units. Before a component can be instantiated in a higher-level design unit, its package must be compiled into a library that is visible to that design unit, usually the current work library.

## Simple Example

Consider the following example. A designer discovers that for a specific type of circuit design he commonly needs an unusual type of counter. ("Commonly," in this reference, means that this counter is likely to be used either multiple times in a particular design or across multiple designs. Both are cases where hierarchical design simplifies things.) This counter is a simple four-bit counter, but it must output a terminal count indication (tc) and roll over to zero when it reaches 1110 rather than 1111.

A design file that would accomplish this is shown in Appendix A. (The reader should understand the contents of the entity-architecture pair—they will not be discussed further.) In order to use this counter in other VHDL design units, it is declared as a component within a package at the top of the file. The component declaration simply names the design unit and lists its signal names. When this file is compiled, the package is placed into the current library and the component it contains may then be instantiated into other designs compiled into that library. If this were a standalone design, the entire package declaration could be omitted.

Now, suppose this design consists of two of these counters with their outputs multiplexed as in *Figure 1.* We can then instantiate our counter twice as in the design in Appendix B. All that is necessary is the statement:

```
use work.cnt_pkg.all;
```

at the top of the file, which makes any components in the cnt_pkg visible within the current design unit, as long as the package was compiled into the work library. The counters are then instantiated by giving them unique labels and listing the signals connected to the port map in the same order as the component declaration.

We could also have created the mux as a separate component and instantiated it, but it is simpler to use the if-then-else structure.

### Multiple Components

For further illustration, assume our complete design includes two types of counters, one that rolls over at 1110 and one that rolls over at 1011, as shown in *Figure 2.* We could simply repeat the above procedure and create another design file with another component and package and then use both of these packages in our top-level design file.

However, it may be easier to keep track of things if we keep similar counter designs together in a single package as in Appendix C. This file contains both entity-architecture pairs and two components in a single package. As before, when this file is compiled, the package is added to the current library and its components are made visible with a single-use clause as in Appendix D.

### Configurable Components using Generics

When multiple components that have the same basic architecture but differ in one or more parameters are needed (such as the two counters in the previous example) VHDL generics allow a more compact approach. Generics are a means by which parameters may be passed to a component when it is instantiated allowing a configurable component.

In Appendix E a component is created that is the same basic counter, but allows the terminal count to be configured using a generic. Instead of hard-coding this value, a bit_vector is used in the architecture. This bit_vector is then declared in the entity and component declarations. Generics may also be of other types such as integers and a component may contain multiple generics (although our example contains only one).

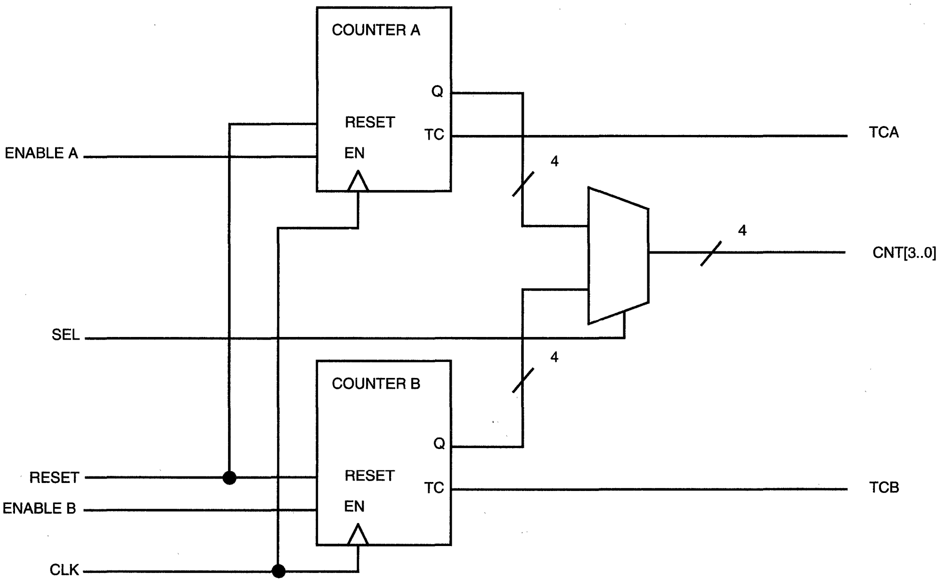


**Figure 1. Multiplexed Dual Counter Design**

Appendix F is the top-level design unit of the same design from *Figure 2*, but this time it is using the component with the generic rather than two different components. When the component is instantiated, it is configured by passing it the specific bit_vector in the generic map.

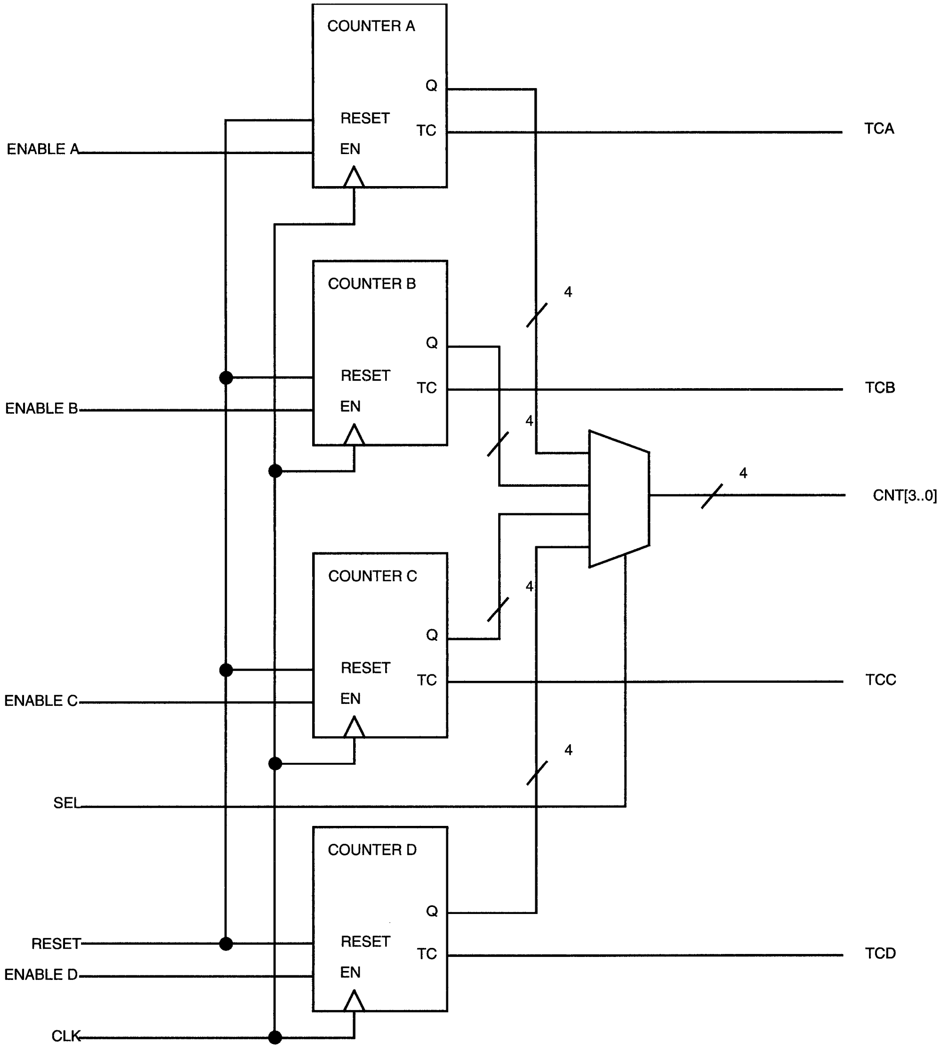


**Figure 2. Multiplexed Quad Counter Design**

## Appendix A. Counter with Terminal Count and Rollover Selection

```
package cnt_pkg is
  component count15 port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
  end component;
end cnt_pkg;
use work.bit_arith.all;
entity count15 is port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
end count15;
architecture one of count15 is
begin
process (cnt) begin
    if cnt="1110" then
        tc<='1';
    else
        tc<='0';
    end if;
end process;
process(clk,reset) begin
    if reset='1' then
        cnt<="0000";
    elsif (clk'event and clk='1') then
        if cnt="1110" and enable='1' then
            cnt<="0000";
        elsif enable='1' then
            cnt<=cnt+1;
        else
            cnt<=cnt;
        end if;
    end if;
end process;
end one;
```

## Appendix B. Instantiation of Counter from Appendix A

```
use work.cnt_pkg.all;
entity muxcntr is port(
    clk, enablea, enableb, reset, sel:in bit;
    cnt:out bit_vector (3 downto 0);
    tca, tcb:out bit);
end muxcntr;
architecture one of muxcntr is
signal muxina, muxinb:bit_vector(3 downto 0);
begin
cntra:count15 port map(clk, enablea, reset, muxina, tca);
cntrb:count15 port map(clk, enableb, reset, muxinb, tcb);
process (sel) begin
    if sel='1' then
        cnt<=muxina;
    else
        cnt<=muxinb;
    end if;
end process;
end one;
```

## Appendix C. Multiple Counters in a Single Package

```
package cnt_pkg is
  component count15 port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
  end component;
  component count12 port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
  end component;
end cnt_pkg;
use work.bit_arith.all;
entity count15 is port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
end count15;
architecture one of count15 is
begin
process (cnt) begin
    if cnt="1110" then
        tc<='1';
    else
        tc<='0';
    end if;
end process;
process(clk,reset) begin
    if reset='1' then
        cnt<="0000";
    elsif (clk'event and clk='1') then
        if cnt="1110" and enable='1' then
            cnt<="0000";
        elsif enable='1' then
            cnt<=cnt+1;
        else
            cnt<=cnt;
        end if;
    end if;
```

## Appendix C. Multiple Counters in a Single Package (continued)

```
end process;
end one;

use work.bit_arith.all;

entity count12 is port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
end count12;

architecture one of count12 is
begin

process (cnt) begin
    if cnt="1011" then
        tc<='1';
    else
        tc<='0';
    end if;
end process;

process(clk,reset) begin
    if reset='1' then
        cnt<="0000";
    elsif (clk'event and clk='1') then
        if cnt="1011" and enable='1' then
            cnt<="0000";
        elsif enable='1' then
            cnt<=cnt+1;
        else
            cnt<=cnt;
        end if;
end if;

end process;
end one;
```

## Appendix D. Instantiation of Counters in Appendix C

```
use work.cnt_pkg.all;
entity muxcntr is port(
    clk, enablea, enableb, enablec, enabled, reset:in bit;
    sel:in bit_vector (1 downto 0);
    cnt:out bit_vector (3 downto 0);
    tca, tcb, tcc, tcd:out bit);
end muxcntr;
architecture one of muxcntr is
signal muxina, muxinb, muxinc, muxind:bit_vector(3 downto 0);
begin
cntra:count15 port map(clk, enablea, reset, muxina, tca);
cntrb:count15 port map(clk, enableb, reset, muxinb, tcb);
cntrc:count12 port map(clk, enablec, reset, muxinc, tcc);
cntrd:count12 port map(clk, enabled, reset, muxind, tcd);
process (sel)begin
    if sel="11" then
        cnt<=muxina;
    elsif sel="10" then
        cnt<=muxinb;
    elsif sel="01" then
        cnt<=muxinc;
    else
        cnt<=muxind;
    end if;
end process;
end one;
```

## Appendix E. Parametrizable Counters Using Generics

```
package cnt_pkg is
    component countg
    generic (stop:bit_vector(3 downto 0):="1111");
      port(
      clk, enable, reset:in bit;
      cnt:inout bit_vector (3 downto 0);
      tc:out bit);
    end component;
end cnt_pkg;

use work.bit_arith.all;

entity countg is
  generic (stop:bit_vector(3 downto 0):="1111");
    port(
    clk, enable, reset:in bit;
    cnt:inout bit_vector (3 downto 0);
    tc:out bit);
end countg;

architecture one of countg is
begin

process (cnt) begin
    if cnt=stop then
        tc<=’1’;
    else
        tc<=’0’;
    end if;
end process;

process(clk,reset) begin
    if reset=’1’ then
        cnt<="0000";
    elsif (clk’event and clk=’1’) then
        if cnt=stop and enable=’1’ then
            cnt<="0000";
        elsif enable=’1’ then
            cnt<=cnt+1;
        else
            cnt<=cnt;
        end if;
    end if;

end process;
end one;
```

## Appendix F. Multiplexed Quad Counter Design

```
use work.cnt_pkg.all;
entity muxcntr is port(
    clk, enablea, enableb, enablec, enabled, reset:in bit;
    sel:in bit_vector (1 downto 0);
    cnt:out bit_vector (3 downto 0);
    tca, tcb, tcc, tcd:out bit);
end muxcntr;
architecture one of muxcntr is
signal muxina, muxinb, muxinc, muxind:bit_vector(3 downto 0);
begin
cntra:countg generic map("1110") port map(clk, enablea, reset, muxina, tca);
cntrb:countg generic map("1110") port map(clk, enableb, reset, muxinb, tcb);
cntrc:countg generic map("1011") port map(clk, enablec, reset, muxinc, tcc);
cntrd:countg generic map("1011") port map(clk, enabled, reset, muxind, tcd);
process (sel) begin
    if sel="11" then
        cnt<=muxina;
    elsif sel="10" then
        cnt<=muxinb;
    elsif sel="01" then
        cnt<=muxinc;
    else
        cnt<=muxind;
    end if;
end process;
end one;
```

CYPRESS

General Information 1

Small PLDs 2

CPLDs 3

Development Tools 4

Application Notes 5

**Quality** 6

Packages 7

**Table of Contents** **Page Number**

## Quality

Quality, Reliability, and Process Flows ....... 6-1
Moisture-Sensitive Devices Handling Information ....... 6-16
Thermal Management and Component Reliability ....... 6-18
Tape and Reel Specifications ....... 6-28

# Quality, Reliability, and Process Flows

## Corporate Views on Quality and Reliability

Cypress believes in product excellence. Excellence can only be defined by how the users perceive both our product quality and reliability. If you, the user, are not satisfied with every device that is shipped, then product excellence has not been achieved.

Product excellence does not occur by following the industry norms. It begins by being better than one's competitors, with better designs, processes, controls and materials. Therefore, product quality and reliability are built into every Cypress product from the beginning.

Some of the techniques used to insure product excellence are the following:

- Product Reliability is built into every product design, starting from the initial design conception.
- Product Quality is built into every step of the manufacturing process through stringent inspections of incoming materials and conformance checks after critical process steps.
- Stringent inspections and reliability conformance checks are done on finished product to insure the finished product quality requirements are met.
- Field data test results are encouraged and tracked so that accelerated testing can be correlated to actual use experiences.

### Product Testing Categories

Three different testing categories are offered by Cypress:

1. Commercial operating range product: 0°C to +70°C.
2. Industrial operating range product: –40°C to +85°C.
3. Military SMD (Standard Military Drawing) product processed to QML Mil PRF 38535; Military operating range: –55°C to +125°C.

## Military Product Assurance

Cypress under the QML program, uses MIL-STD-883 and MIL-PRF-38535 as baseline documents to determine our Test Methods, Procedures and General Specifications for Semiconductors.

Cypress's Military components and SMD products are processed per MIL-STD-883 using methods 5004 and 5005 to baseline our screening and quality conformance procedures. Refer to *Tables 3* to *7* for the baseline flows and requirements. The processing performed by Cypress results in a product that meets the class B screening requirements as specified by these methods. Every device shipped, as a minimum, meets these requirements.

## Commercial Product Assurance

Cypress is a ISO9000 certified supplier. All commercial and industrial temp range products are manufactured using the same controlled systems as our QML military product. *Tables 1* and *2* define the 100% screening and conformance inspection for commercial and industrial temp range product.

**Table 1. Cypress Commercial and Industrial Product Screening Flows—Components**

| Screen | MIL-STD-883 Method | Product Temperature Ranges | |
|---|---|---|---|
| | | Commercial 0°C to +70°C; Industrial –40°C to +85°C | |
| | | Plastic | Hermetic |
| **Visual/Mechanical** | | | |
| • Internal Visual | 2010 | 0.4% AQL | 100% |
| • Hermeticity | | | |
| —Fine Leak | 1014, Cond A or B (sample) | Does Not Apply | LTPD = 5 |
| —Gross Leak | 1014, Cond C | Does Not Apply | 100% |
| **Final Electrical** | Per Device Specification | | |
| • Static (DC), Functional, and Switching (AC) Tests | 1. At Hot Temperature and Power Supply Extremes | 100% | 100% |
| **Cypress Quality Lot Acceptance** | | | |
| • External Visual | 2009 | Note 1 | Note 1 |
| • Final Electrical Conformance | Cypress Method 17-00064 | Note 1 | Note 1 |

**Table 2. Cypress Commercial and Industrial Product Screening Flows—Modules**

| Screen | MIL-STD-883 Method | Product Temperature Ranges |
|---|---|---|
| | | Commercial 0°C to +70°C; Industrial –40°C to +85°C |
| **Final Electrical** | Per Device Specification | |
| • Static (DC), Functional, and Switching (AC) Tests | 1. At 25°C and Power Supply Extremes | 100% |
| | 2. At Hot Temperature and Power Supply Extremes | 100% |
| **Cypress Quality Lot Acceptance** | | |
| • External Visual | 2009 | Per Cypress Module Specification |
| • Final Electrical Conformance | Cypress Method 17-00064 | Note 1 |

**Note:**

1. Lot acceptance testing is performed on every lot. AOQL (the Average Outgoing Quality Level) for 1995 was 0 PPM.

**Table 3. Cypress QML/JAN/SMD/Military Product Screening Flows for Class B**

| Screen | Screening Per Method 5004 of MIL-STD-883 | Product Temperature Ranges –55°C to +125°C | |
|---|---|---|---|
| | | QML/JAN/SMD/Military Components[2] | Military Modules |
| **Visual/Mechanical** | | | |
| • Internal Visual | Method 2010, Cond B | 100% | N/A |
| • Temperature Cycling | Method 1010, Cond C, (10 cycles) | 100% | 100% |
| • Constant Acceleration | Method 2001, Cond E (Min.), Y1 Orientation Only | 100% | N/A |
| • Hermeticity: | | | |
| —Fine Leak | Method 1014, Cond A or B | 100% | N/A |
| —Gross Leak | Method 1014, Cond C | 100% | 100% |
| **Burn-in** | | | |
| • Pre-Burn-in Electrical Parameters | Per Applicable Device Specification | 100% | 100% |
| • Burn-in Test | Method 1015, Cond D,<br>160 Hrs at 125°C Min. or<br>80 Hrs at 150°C | 100% | 100%<br>(48 Hours at 125°C) |
| • Post-Burn-in Electrical Parameters | Per Applicable Device Specification | 100% | 100% |
| • Percent Defective Allowable (PDA) | Maximum PDA, for All Lots | 5% | 5% |
| **Final Electrical Tests** | | | |
| • Static Tests | Method 5005<br>Subgroups 1, 2, and 3 | 100% Test to Applicable Device Specification | 100% Test to Applicable Device Specification |
| • Functional Tests | Method 5005<br>Subgroups 7, 8A, and 8B | 100% Test to Applicable Device Specification | 100% Test to Applicable Device Specification |
| • Switching | Method 5005<br>Subgroups 9, 10, and 11 | 100% Test to Applicable Device Specification | 100% Test to Applicable Device Specification |
| **Quality Conformance Tests** | | | |
| • Group A | Method 5005, See *Table 4* | Sample | Sample |
| • Group B | Method 5005, See *Table 5* | Sample | Sample |
| • Group C[3] | Method 5005, See *Table 6* | Sample | Sample |
| • Group D[3] | Method 5005, See *Table 7* | Sample | Sample |
| **External Visual** | Method 2009 | 100% | 100% |

**Notes:**

2. QML product is allowed a reduction in screening requirements with DESC approval per MIL-PRF-38535.
3. Group C and D end-point electrical tests for QML/SMD/Military Grade products are performed to Group A subgroups 1, 2, 3, 7, 8A, 8B, 9, 10, 11, or per JAN Slash Sheet.

**Table 4. Group A Test Descriptions**

| Sub-group | Description | Sample Size/Accept No. | |
|---|---|---|---|
| | | Components | Modules[4] |
| 1 | Static Tests at 25°C | 116/0 | 116/0 |
| 2 | Static Tests at Maximum Rated Operating Temperature | 116/0 | 116/0 |
| 3 | Static Tests at Minimum Rated Operating Temperature | 116/0 | 116/0 |
| 4 | Dynamic Tests at 25°C | 116/0 | 116/0 |
| 5 | Dynamic Tests at Maximum Rated Operating Temperature | 116/0 | 116/0 |
| 6 | Dynamic Tests at Minimum Rated Operating Temperature | 116/0 | 116/0 |
| 7 | Functional Tests at 25°C | 116/0 | 116/0 |
| 8A | Functional Tests at Maximum Temperature | 116/0 | 116/0 |
| 8B | Functional Tests at Minimum Temperature | 116/0 | 116/0 |
| 9 | Switching Tests at 25°C | 116/0 | 116/0 |
| 10 | Switching Tests at Maximum Temperature | 116/0 | 116/0 |
| 11 | Switching Tests at Minimum Temperature | 116/0 | 116/0 |

Cypress uses an LTPD sampling plan that was developed by the Military to assure product quality. Testing is performed to the subgroups found to be appropriate for the particular device type. All Military products have a Group A sample test performed on each inspection lot per MIL-PRF-38535 or MIL-STD-883 and the applicable device specification.

**Table 5. Group B Quality Tests**

| Sub-group | Description | Quantity/Accept # or LTPD | |
|---|---|---|---|
| | | Components | Modules[4] |
| 2 | Resistance to Solvents, Method 2015 | 3/0 | 3/0 |
| 3 | Solderability, Method 2003[5] | 22/0 | 3 |
| 5 | Bond Strength, Method 2011[6] | 15/0 | NA |

Group B testing is performed for each inspection lot. An inspection lot is defined as a group of material of the same device type, package type and lead finish built within a six week seal period and submitted to Group B testing at the same time.

**Table 6. Group C Quality Tests**

| Sub-group | Description | LTPD | |
|---|---|---|---|
| | | Components | Modules[4] |
| 1 | Steady State Life Test, End-Point Electricals, Method 1005, Cond D | 45/0 | 15/0 |

Group C tests for all Military Grade products are performed on one device type from one inspection lot representing each technology. Sample tests are performed per MIL-PRF-38535/MIL-STD-883 from each four calendar quarters production of devices, which is based upon the die fabrication date code.

End-point electrical tests and parameters are performed per the applicable device specification.

**Notes:**

[4] Military Grade Modules are processed to proposed JEDEC standard flows for MIL-STD-883 compliant modules. Alternate Group A method as detailed in JC-13-BP-123A.

[5] Sample size is based upon leads taken from a minimum of 3 devices.

[6] Sample size is based upon leads taken from a minimum of 4 devices.

**Table 7. Group D Quality Tests (Package Related)**

| Sub-group | Description | Quantity/Accept # or LTPD | |
|---|---|---|---|
| | | Components | Modules[7] |
| 1 | Physical Dimensions, Method 2016 | 15/0 | 15/0 |
| 2 | Lead Integrity, Seal: Fine and Gross Leak, Method 2004 and 1014 | 45/0[5] | 15/0<br>N/A for Seal |
| 3 | Thermal Shock, Temp-Cycling, Moisture Resistance, Seal: Fine and Gross Leak, Visual Examination, End-Point, Electricals, Methods 1011, 1010, 1004, and 1014 | 15/0 | 15/0<br>N/A for Moisture Resistance; N/A for Fine Leak |
| 4 | Mechanical Shock, Vibration - Variable Frequency, Constant Acceleration, Seal: Fine and Gross Leak, Visual Examination, End-Point Electricals, Methods 2002, 2007, 2001, and 1014 | 15/0 | 15/0<br>N/A for Constant Acceleration;<br>N/A for Fine Leak |
| 5 | Salt Atmosphere, Seal: Fine & Gross Leak, Visual Examination, Methods 1009 & 1014 | 15/0 | 15/0<br>N/A for Fine Leak |
| 6 | Internal Water-Vapor Content; 5000 ppm maximum @ 100°C. Method 1018 | 3(0) or 5(1) | N/A |
| 7 | Adhesion of Lead Finish,[8] Method 2025 | 15/0 | 15/0 |
| 8 | Lid Torque, Method 2024[9] | 5(0) | N/A |

Group D tests for all Military Grade procedures are performed per MIL-PRF-38535/MIL-STD-883 on each package type from each six months of production, based on the identification (or date) codes.

End-point electrical tests and parameters are performed per the applicable device specification.

## Military Modules

- Military Temperature Grade Modules are designated with an 'M' suffix only. These modules are screened to standard combined flows and tested at both military temperature extremes.
- MIL-STD-883 Equivalent Modules are processed to proposed JEDEC standard flows for MIL-STD-883 compliant modules. All MIL-STD-883 equivalent modules are assembled with fully compliant MIL-STD-883 components.

## Product Screening Summary Components

### Commercial and Industrial Product

- Screened per *Table 1* product assurance flows
- Hermetic and molded packages available
- Incoming mechanical and electrical performance guaranteed:
  - — 0.02% AQL Electrical Sample test performed on every lot prior to shipment
  - — 0.01% AQL External Visual Sample inspection
- Electrically tested to Cypress data sheet

### Ordering Information

- Order Standard Cypress part number
- Parts marked the same as ordered part number
  Ex: CY7C122–15PC, PALC22V10–25PI

## Military Product

- SMD and Military components are manufactured in compliance with paragraph 1.2.1 of MIL-STD-883. Compliant products are identified by an 'MB' suffix on the part number (CY7C122–25DMB) and the letter "C"
- QML devices are manufactured in accordance with MIL-PRF-38535. Compliant products are identified with the letter "Q."
- Military devices electrically tested to:
  - — SMD devices are electrically tested to the applicable standard military drawing specifications
    OR
  - — Cypress data sheet specifications
- All devices supplied in hermetic packages
- Quality conformance inspection: Method 5005, Groups A, B, C, and D performed as part of the standard process flow
- Burn-in performed on all devices
  - — Cypress detailed circuit specification for non-JAN devices
- Static functional and switching tests performed at 25°C as well as temperature and power supply extremes on 100% of the product in every lot

### Ordering Information

**SMD Product:**

- Order per military document
- Marked per military document
  Ex: 5962–8867001LA

**Military Product:**

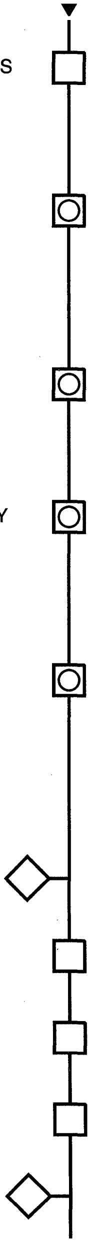
- Order per Cypress standard military part number
- Marked the same as ordered part number
  Ex: CY7C122–25DMB

**Notes:**

7. Does not apply to leadless chip carriers.
8. Based on the number of leads.
9. Applies only to packages with glass seals.

6

## Product Quality Assurance Flow—Components

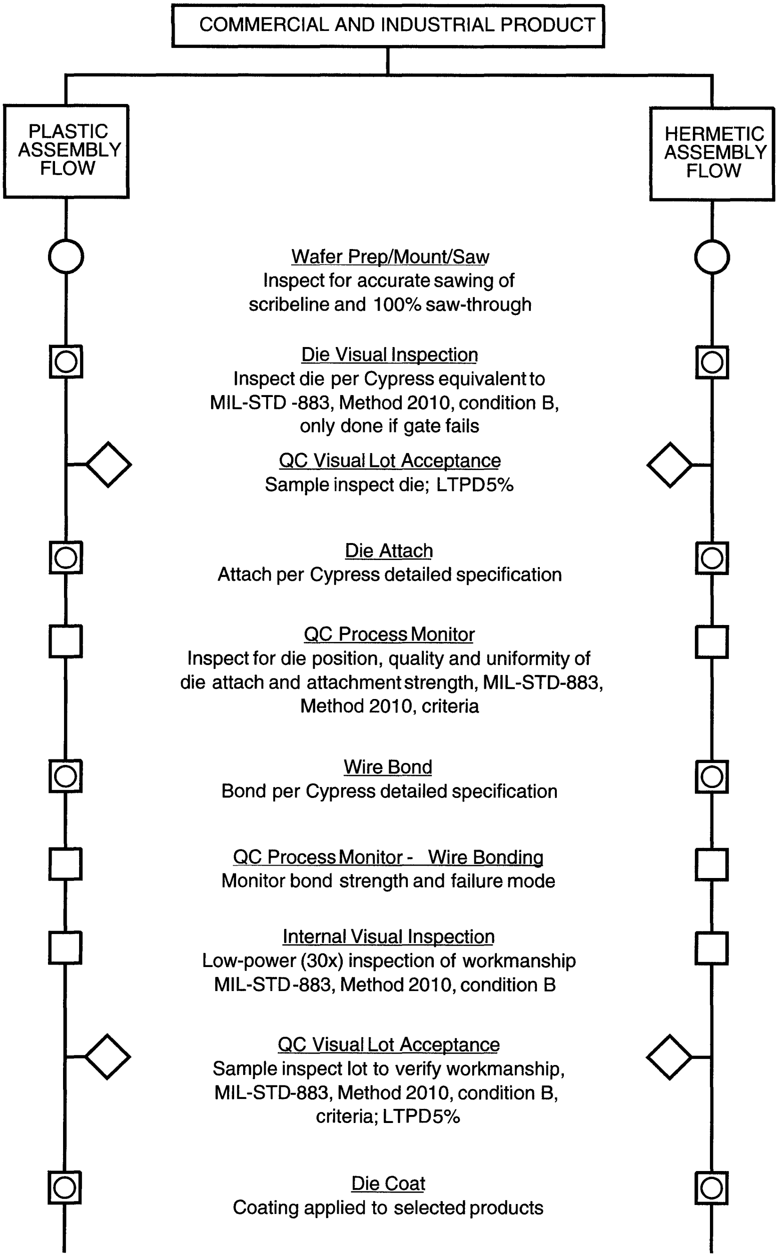


| Area | PROCESS | Process Details |
|---|---|---|
| QC | INCOMING MATERIALS INSPECTION | All incoming materials are inspected to documented procedures covering the handling, inspection, storage, and release of raw materials used in the manufacture of Cypress products. Materials inspected are: wafers, masks, leadframes, ceramic packages and/or piece parts, molding compounds, gases, chemicals, etc. |
| FAB | DIFFUSION/ION IMPLANTATION | Sheet resistance, implant dose, species and CV characteristicsare measured for all critical implants on every product run. Testwafers may be used to collect this data instead of actual production wafers. If this is done, they are processed with the standard product prior to collecting specific data. This insures accurate correlation between the actual product and the wafers used to monitor implantation. |
| FAB | OXIDATION | Sample wafers and sample sites are inspected on each run from various positions of the furnace load to inspect for oxide thickness.Theintegrity of critical oxides is monitored at electrical test. |
| FAB | PHOTOLITHOGRAPHY /ETCHING | Appearance of resist is checked by the operator after the spin operation. Also, after the film is developed, both dimensions and appearance are checked by the operator on a sample of wafers and locations upon each wafer. Final CDs and alignment are also sample inspected on several wafers and sites on each wafer on every product run. |
| FAB | METALIZATION | Film thickness is monitored on a daily basis. Step coverage cross–sectionsare performed on a periodic basis to insure coverage. |
| FAB | PASSIVATION | Film thickness is verified on a sample of wafers and locations on each run. Film stress is monitored on a weekly basis. |
| FAB | QC VISUALOF WAFERS | |
| FAB | E-TEST | Electrical test is performed for final process electrical characteristicson every wafer. |
| FAB | QC MONITOROF E-TEST DATA | Weekly review of all data trends; running averages, minimums, maximums, etc. are reviewed with the process control manager. |
| TEST | WAFERPROBE/SORT | Verify functionality, electrical characteristics, stress test devices. |
| TEST | QC CHECK PROBING AND ELECTRICAL TEST RESULTS | Pass/fail lot based on yield and correct probe placement. |

TO ASSEMBLY AND TEST

(continued)

## Product Quality Assurance Flow—Components (continued)
## Commercial and Industrial Product



COMMERCIAL AND INDUSTRIAL PRODUCT

PLASTIC ASSEMBLY FLOW

HERMETIC ASSEMBLY FLOW

Wafer Prep/Mount/Saw
Inspect for accurate sawing of scribeline and 100% saw-through

Die Visual Inspection
Inspect die per Cypress equivalent to MIL-STD -883, Method 2010, condition B, only done if gate fails

QC Visual Lot Acceptance
Sample inspect die; LTPD5%

Die Attach
Attach per Cypress detailed specification

QC Process Monitor
Inspect for die position, quality and uniformity of die attach and attachment strength, MIL-STD-883, Method 2010, criteria

Wire Bond
Bond per Cypress detailed specification

QC Process Monitor - Wire Bonding
Monitor bond strength and failure mode

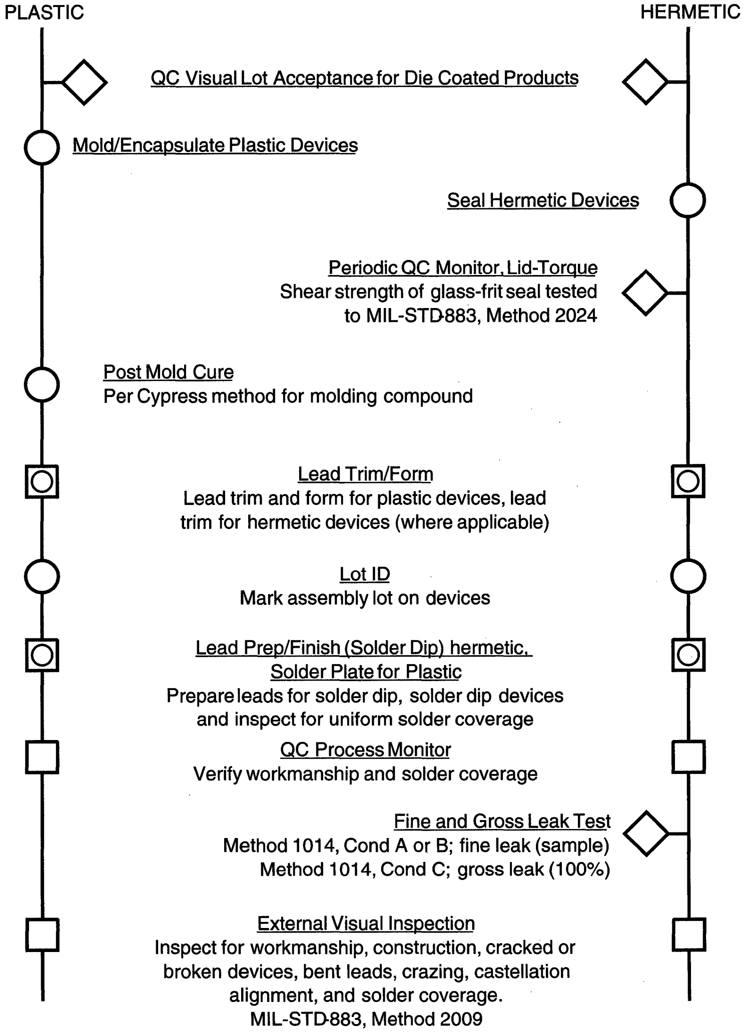
Internal Visual Inspection
Low-power (30x) inspection of workmanship MIL-STD-883, Method 2010, condition B

QC Visual Lot Acceptance
Sample inspect lot to verify workmanship, MIL-STD-883, Method 2010, condition B, criteria; LTPD5%

Die Coat
Coating applied to selected products

(continued)

## Product Quality Assurance Flow—Components (continued)
## Commercial and Industrial Product



PLASTIC

HERMETIC

QC Visual Lot Acceptance for Die Coated Products

Mold/Encapsulate Plastic Devices

Seal Hermetic Devices

Periodic QC Monitor, Lid-Torque
Shear strength of glass-frit seal tested
to MIL-STD-883, Method 2024

Post Mold Cure
Per Cypress method for molding compound

Lead Trim/Form
Lead trim and form for plastic devices, lead
trim for hermetic devices (where applicable)

Lot ID
Mark assembly lot on devices

Lead Prep/Finish (Solder Dip) hermetic,
Solder Plate for Plastic
Prepare leads for solder dip, solder dip devices
and inspect for uniform solder coverage

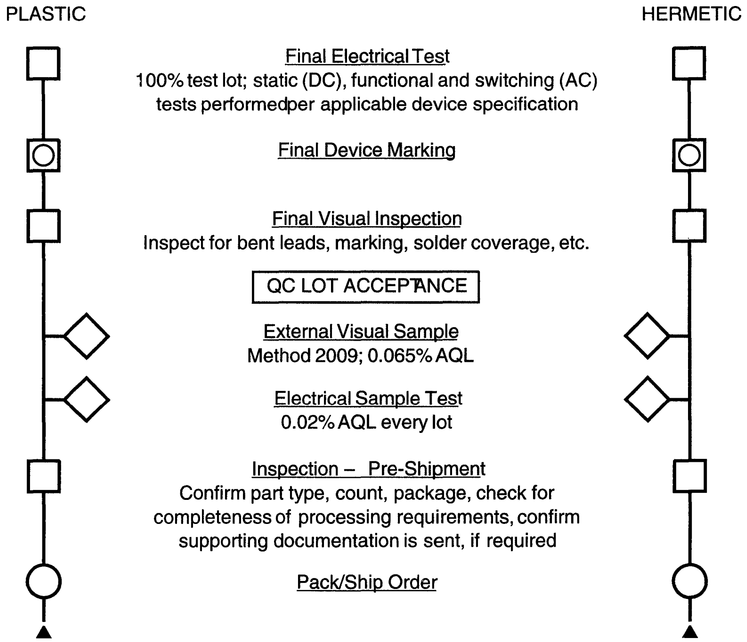
QC Process Monitor
Verify workmanship and solder coverage

Fine and Gross Leak Test
Method 1014, Cond A or B; fine leak (sample)
Method 1014, Cond C; gross leak (100%)

External Visual Inspection
Inspect for workmanship, construction, cracked or
broken devices, bent leads, crazing, castellation
alignment, and solder coverage.
MIL-STD-883, Method 2009

(continued)

## Product Quality Assurance Flow—Components (continued)
## Commercial and Industrial Product



Final Electrical Test
100% test lot; static (DC), functional and switching (AC) tests performedper applicable device specification

Final Device Marking

Final Visual Inspection
Inspect for bent leads, marking, solder coverage, etc.

QC LOT ACCEPTANCE

External Visual Sample
Method 2009; 0.065% AQL

Electrical Sample Test
0.02% AQL every lot

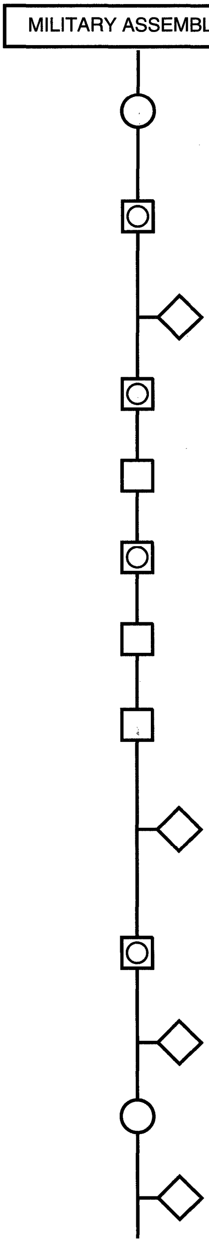
Inspection – Pre-Shipment
Confirm part type, count, package, check for completeness of processing requirements, confirm supporting documentation is sent, if required

Pack/Ship Order

**Key**

- ○ Production Process
- □ Test/Inspection
- ◙ Production Process and Test Inspection
- ◇ QC Sample Gate and Inspection

## Product Quality Assurance Flow—Components (continued)
## Military Components



MILITARY ASSEMBLY FLOW

<u>Wafer Prep/Mount/Saw</u>
Inspect for accurate sawing of scribeline and 100% saw-through

<u>Die Visual Inspection</u>
Inspect die per MIL-STD-883, Method 2010, condition B

<u>QC Visual Lot Acceptance</u>
Sample inspect die; 1.0% AQL

<u>Die Attach</u>
Attach per Cypress detailed specification

<u>Die Adherence Monitor</u>
MIL-STD-883, Method 2019 or Method 2027

<u>Wire Bond</u>
Bond per Cypress detailed specification

<u>Bond Pull Monitor</u>
MIL-STD-883, Method 2011

<u>Internal Visual Inspection</u>
Low-power and high-power inspection per
MIL-STD-883, Method 2010, condition B

<u>QC Visual Lot Acceptance</u>
Sample inspect lot per MIL-STD-883,
Method 2010, condition B, 0.4% AQL

<u>Die Coat</u>
Coating applied to selected products

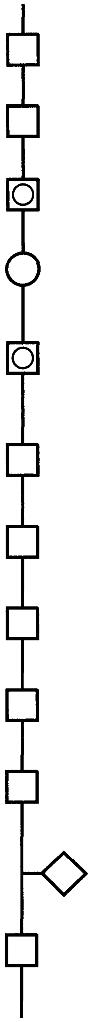
<u>QC Visual Lot Acceptance for Die Coated Products</u>

<u>Seal</u>

<u>Periodic QC Monitor, Lid-Torque</u>
Shear strength of glass

(continued)

## **Product Quality Assurance Flow—Components** (continued)
## Military Components



Temperature Cycle
Method 1010, Cond C, 10 cycles

Constant Acceleration
Method 2001, Cond E, Y1 Orientation

Lead Trim
Lead trim when applicable

Lot ID
Mark assembly lot on devices

Lead Finish
Solder dip or matte tin plate applicable devices and inspect

QC Process Monitor
Verify workmanship and lead finish coverage

External Visual Inspection
Method 2009

Pre-Burn-In Electrical Test
Method 5004, per applicable device specification

Burn-In
Method 1015, condition D

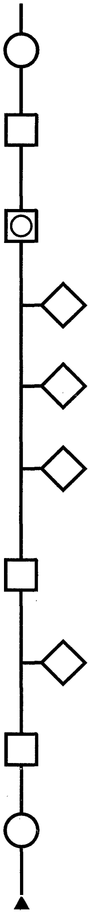
Post-Burn-In Electricals
Method 5004, per applicable device specification

PDA Calculation
Method 5004, 5%

Final Electrical Test
Method 5004; Static, functional and switching tests per applicable device specification

(continued)

## Product Quality Assurance Flow—Components (continued)
## Military Components



Lead Finish - Solder Dip
Solder dip applicable devices

Fine and Gross Leak Test
Method 1014, condition A or B, fine leak; condition C, gross leak

Final Device Marking
MIL-STD-883 or applicable device specification

Group B
Method 5005

Group A
Method 5005, per applicable device specification

Group C and D
Method 5005, in accordance with
1.2.1 of MIL-STD-883; JAN devices
in accordance with MIL-PRF-38535

External Visual
Method 2009, 100% inspection

External Visual Sample
Method 2009, 0.01% AQL

Plant Clearance

Pack/Ship Order

**Key**

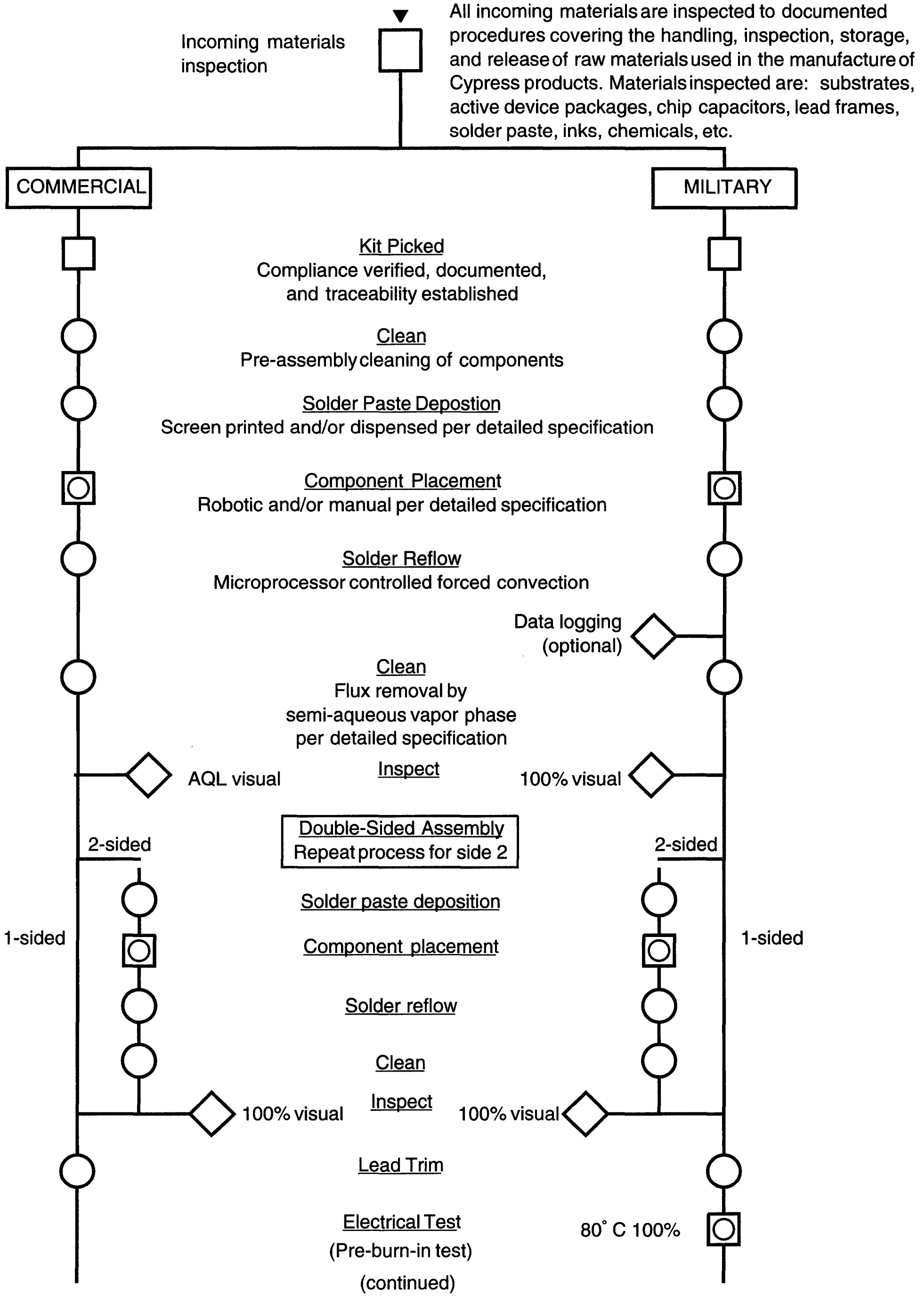
○ Production Process

□ Test/Inspection

⧇ Production Process and Test Inspection

◇ QC Sample Gate and Inspection

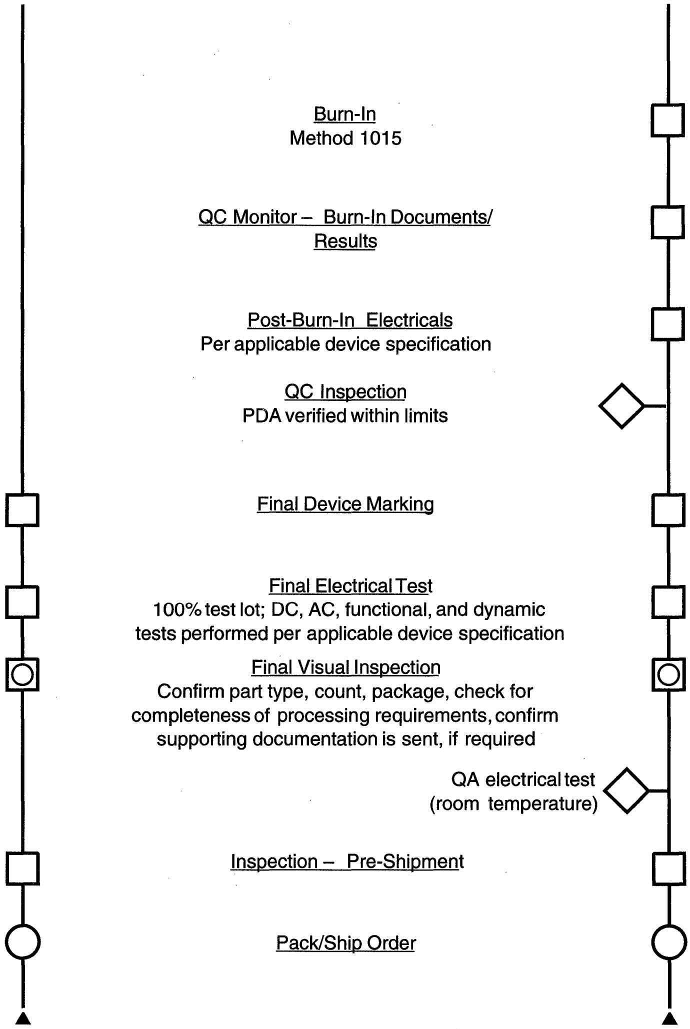
## Product Quality Assurance Flow—Modules



All incoming materials are inspected to documented procedures covering the handling, inspection, storage, and release of raw materials used in the manufacture of Cypress products. Materials inspected are: substrates, active device packages, chip capacitors, lead frames, solder paste, inks, chemicals, etc.

Incoming materials inspection
COMMERCIAL
MILITARY
Kit Picked
Compliance verified, documented, and traceability established
Clean
Pre-assembly cleaning of components
Solder Paste Depostion
Screen printed and/or dispensed per detailed specification
Component Placement
Robotic and/or manual per detailed specification
Solder Reflow
Microprocessor controlled forced convection
Data logging (optional)
Clean
Flux removal by semi-aqueous vapor phase per detailed specification
AQL visual
Inspect
100% visual
Double-Sided Assembly
Repeat process for side 2
2-sided
2-sided
1-sided
1-sided
Solder paste deposition
Component placement
Solder reflow
Clean
100% visual
Inspect
100% visual
Lead Trim
Electrical Test
(Pre-burn-in test)
80° C 100%
(continued)

## Product Quality Assurance Flow—Modules (continued)



Burn-In
Method 1015

QC Monitor – Burn-In Documents/
Results

Post-Burn-In Electricals
Per applicable device specification

QC Inspection
PDA verified within limits

Final Device Marking

Final Electrical Test
100% test lot; DC, AC, functional, and dynamic
tests performed per applicable device specification

Final Visual Inspection
Confirm part type, count, package, check for
completeness of processing requirements, confirm
supporting documentation is sent, if required

QA electrical test
(room temperature)

Inspection – Pre-Shipment

Pack/Ship Order

**Key**

○ Production Process

□ Test/Inspection

◙ Production Process and Test Inspection

◇ QC Sample gate and inspection

## Reliability Monitor Program

The Reliability Monitor Program is a documented Cypress procedure that is described in Cypress specification #25-00008, which is available to Cypress customers upon request. This specification describes a procedure that provides for periodic reliability monitors to insure that all Cypress products comply with established goals for reliability improvement and to minimize reliability risks for Cypress customers. The Reliability Monitor Program monitors our most advanced technologies and packages. Every technology produced at a given fabrication site (Tech. – Fab.) and all assembly houses are monitored at least quarterly. If failures occur, detailed failure analyses are performed and corrective actions are implemented. A summary of the Reliability Monitor Program test and sampling plan is shown below.

## Quarterly Reliability Monitor Test Matrix

| Stress | Sampling Strategy | Lots Tested per Quarter |
|---|---|---|
| HTOL | Technology–Fab Location | 8 |
| HTSSL | Technology–Fab Location | 8 |
| TEV | Technology–Fab Location | 8 |
| DRET | Technology–Fab Location | 2 |
| HAST | Technology–Fab Location | 8 |
| | Package–Assembly Location | 10 |
| TC | Technology–Fab Location | 8 |
| | Package–Assembly Location | 12 |
| PCT | Package–Assembly Location | 10 |

## Reliability Monitor Test Conditions

| Test | Abbrev. | Temp. (°C) | R.H. (%) | Bias | Sample Size | LTPD | Read Points (hrs.) |
|---|---|---|---|---|---|---|---|
| High-Temperature Operating Life | HTOL | +125 | N/A | 5.75V Dynamic | 116 | 2 | 96, 500, 1000 |
| High-Temperature Steady-State Life | HTSSL | +125 | N/A | 5.75V Static | 116 | 2 | 96, 500, 1000 |
| Data Retention for Plastic Packages | DRET | +165 | N/A | N/A | 76 | 3 | 168, 1000 |
| Data Retention for Ceramic Packages | DRET2 | +250 | N/A | N/A | 76 | 3 | 168, 500 |
| Pressure Cooker | PCT | +121 | 100 | N/A | 76 | 3 | 96, 168 |
| Highly Accelerated Stress Test | HAST | +140 | 85 | 5.5V Static | 76 | 3 | 128 |
| Temperature Cycling | TC | –65 to +150°C | N/A | N/A | 45 | 5 | 300, 1000 Cycles |
| Temperature Extreme Verification | TEV | Commercial Hot & Cold 0 to +70°C | N/A | N/A | 116 | 2 | N/A |

fax id: 6001

CYPRESS

# Moisture-Sensitive Devices Handling Information

## Cypress Dry Bag Policy

In order to insure against moisture damage, Cypress carries out dry bake and dry packing on all devices that are moisture sensitive in surface mount applications. These devices are shipped in sealed Moisture Barrier Bags (MBBs) with caution labels similar to *Figure 1* Cypress recommends that all PLCCs with pin counts of 44 and higher, all Plastic Quad Flat Packs (PQFPs), and Thin Quad Flat Packs (TQFPs) be used dry in surfacemount applications. *NOTE:* ***A package is considered dry if it has been baked for 24 continuous hours at 125 5C and has been stored at or below 20% Relative Humidity (RH) prior to the reflow-soldering process.***

## Moisture-Sensitivity

The extremely high temperatures and steep temperature gradients that are present during the reflow-soldering processes used in attaching surface mount devices to circuit boards could damage to moisture-sensitive devices if they have absorbed excessive moisture. The moisture trapped in such a device vaporizes during the reflow-soldering processes and may generate significant hydrostatic pressure within the package. In the worst case, this pressure may cause an internal or external crack in the overmold that allows flux and other contaminants to reach the die area. The final result is a cracked device that will suffer an early failure.

## Cypress Dry-Packing Process

The Cypress dry-packing process starts with baking the moisture sensitive devices at 125 5°C for 24 continuous hours in bakeable carrier trays (for TSOP, TQFP, PQFP devices) or aluminum tubes (for SOIC, SOJ, and PLCC devices). Baked devices are then vacuum sealed and dry packed in MBBs within the 24 hour factory-floor-life limit under controlled environment.

In each of the vacuum sealed MBBs, appropriate amount of desiccant packs are enclosed to keep the enclosed devices at less than 20% RH for up to 12 months from the date of seal as stated on the caution label on the bag. A reversible humidity indicator card (HIC) that has indicator grades, is also enclosed to monitor the internal humidity level. On the outside of each sealed bag a caution label similar to the one shown is attached to inform and alert the customers of the seal date information and the need for special handling precautions.

## Dry-Packed Part Handling Procedures

### Package Inspection

Upon receiving the package, check the seal date and make sure that the package has no holes, tears, or openings that may endanger the enclosed devices to humidity. Cypress recommends that the bag stay sealed until the enclosed devices are ready for use.

### Storage Condition

In line with the warning stated on the caution label, the sealed MBBs should be stored unopened in a relatively dry environment or one of no more than 90% relative humidity and 40°C.

### Expiration Date: First Seal Date Plus 12 Months

The expiration date of a sealed MBB is 12 months from the original seal date if it has been stored in an environment of less than 40°C and 90% humidity. If the expiration date has been exceeded, or if upon opening a bag within its stated expiration period, the HIC display a humidity level of over 30%, the enclosed devices “may still be used with the addition of a bake of 192 hours at 40°C with less than 5% humidity, or a bake of 24 hours at 125°C with less than 5% humidity.” After the baking, any of the following options may apply:

- Use the parts within 48 hours
- Reseal the devices in a MBB within 12 hours after baking with fresh desiccant packs and HIC
- Store the devices in a controlled cabinet with less than 20% RH.

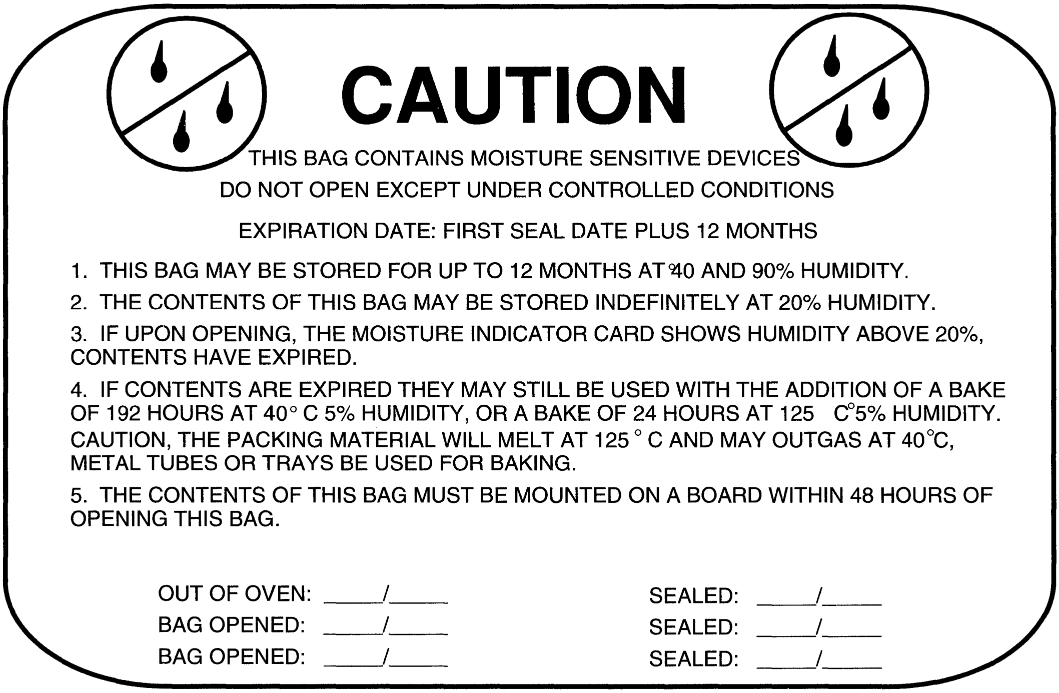
## Opening a Sealed Bag

When parts are ready to be used, open the MBB by cutting across the top within one inch of the seal area. Use the dry parts following the guide line under the factory-floor-life section to ensure they are maintained below critical moisture levels. Opened bags must be resealed immediately (the factory-floor-life is cumulative and has a maximum of 24 hours) and the information on the date the bag is opened and resealed must be filled out on the caution label. If the opened bag is not resealed immediately, and provided that the factory-floor-life has not been exceeded, or in the case of splitting the devices of a bag, and new desiccant, the same packing and sealing procedure must also be applied.

## Factory Floor Life

Cypress recommends the maximum cumulative time that devices can be exposed to the open air without requiring re-bake and resulting in moisture-related damages to be 24 hours with environment condition not worse than 40°C and 85% RH. (This includes processing time after bake but prior to seal at Cypress and any time the bag is re-opened after seal.) If the floor life is exceeded, it is recommended that the devices be dry baked as if the storage period has expired.

If at any time, the recommended storage and factory floor conditions, or handling guidelines might be violated, please consult Cypress for more information.



CAUTION

THIS BAG CONTAINS MOISTURE SENSITIVE DEVICES
DO NOT OPEN EXCEPT UNDER CONTROLLED CONDITIONS

EXPIRATION DATE: FIRST SEAL DATE PLUS 12 MONTHS

1. THIS BAG MAY BE STORED FOR UP TO 12 MONTHS AT °40 AND 90% HUMIDITY.

2. THE CONTENTS OF THIS BAG MAY BE STORED INDEFINITELY AT 20% HUMIDITY.

3. IF UPON OPENING, THE MOISTURE INDICATOR CARD SHOWS HUMIDITY ABOVE 20%, CONTENTS HAVE EXPIRED.

4. IF CONTENTS ARE EXPIRED THEY MAY STILL BE USED WITH THE ADDITION OF A BAKE OF 192 HOURS AT 40° C 5% HUMIDITY, OR A BAKE OF 24 HOURS AT 125 C°5% HUMIDITY. CAUTION, THE PACKING MATERIAL WILL MELT AT 125 ° C AND MAY OUTGAS AT 40°C, METAL TUBES OR TRAYS BE USED FOR BAKING.

5. THE CONTENTS OF THIS BAG MUST BE MOUNTED ON A BOARD WITHIN 48 HOURS OF OPENING THIS BAG.

OUT OF OVEN: _____/_____ SEALED: _____/_____
BAG OPENED: _____/_____ SEALED: _____/_____
BAG OPENED: _____/_____ SEALED: _____/_____

**Figure 1. Caution Label**

Document #: 38-00501

# Thermal Management and Component Reliability

One of the key variables determining the long-term reliability of an integrated circuit is the junction temperature of the device during operation. Long-term reliability of the semiconductor chip degrades proportionally with increasing temperatures following an exponential function described by the Arrhenius equation of the kinetics of chemical reactions. The slope of the logarithmic plots is given by the activation energy of the failure mechanisms causing thermally activated wear out of the device (see *Figure 1*).

Typical activation energies for commonly observed failure mechanisms in CMOS devices are shown in *Table 1.*

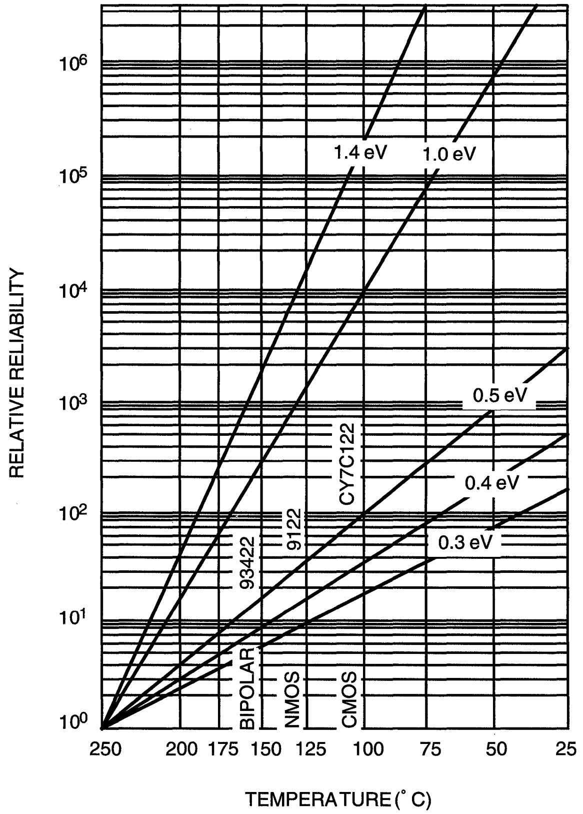


**Figure 1. Arrhenius plot, which assumes a failure rate proportional to EXP ($-E_A/kT$) where $E_A$ is the activation energy for the particular failure mechanism**

**Table 1. Failure Mechanisms and Activation Energies in CMOS Devices**

| Failure Mode | Approximate Activation Energy (Eq) |
|---|---|
| Oxide Defects | 0.3 eV |
| Silicon Defects | 0.3 eV |
| Electromigration | 0.6 eV |
| Contact Metallurgy | 0.9 eV |
| Surface Charge | 0.5-1.0 eV |
| Slow Trapping | 1.0 eV |
| Plastic Chemistry | 1.0 eV |
| Polarization | 1.0 eV |
| Microcracks | 1.3 eV |
| Contamination | 1.4 eV |

To reduce thermally activated reliability failures, Cypress Semiconductor has optimized both their low-power-generating CMOS device fabrication process and their high-heat-dissipation packaging capabilities. *Table 2* demonstrates this optimized thermal performance by comparing bipolar, NMOS, and Cypress high-speed 1K SRAM CMOS devices in their respective plastic packaging environments under standard operating conditions.

**Table 2. Thermal Performance of Fast 1K SRAMs in Plastic Packages**

| Technology | Bipolar | NMOS | Cypress CMOS |
|---|---|---|---|
| Device Number | 93422 | 9122 | 7C122 |
| Speed (ns) | 30 | 25 | 25 |
| $I_{CC}$ (mA) | 150 | 110 | 60 |
| $V_{CC}$ (V) | 5.0 | 5.0 | 5.0 |
| $P_{MAX}$ (mW) | 750 | 550 | 300 |
| Package RTH (JA) (°C/W) | 120 | 120 | 70 |
| Junction Temperature (°C) at Datasheet $P_{MAX}$[1] | 160 | 136 | 91 |

During its normal operation, the Cypress 7C122 device experiences a 91°C junction temperature, whereas competitive devices in their respective packaging environments see a 45°C and 69°C higher junction temperature. In terms of relative reliability life expectancy, assuming a 1.0-eV activation energy failure mechanism, this translates into an improvement in excess of two orders of magnitude (100x) over the bipolar 93422 device, and more than one order of magnitude (30x) over the NMOS 9122 device.

## Thermal Performance Data of Cypress Component Packages

The thermal performance of a semiconductor device in its package is determined by many factors, including package design and construction, packaging materials, chip size, chip thickness, chip attachment process and materials, package size, etc.

### Thermal Resistance ($\theta_{JA}$, $\theta_{JC}$)

Thermal resistance is a measure of the ability of a package to transfer the heat generated by the device inside it to the ambient.

For a packaged semiconductor device, heat generated near the junction of the powered chip causes the junction temperature to rise above the ambient temperature. The total thermal resistance is defined as

$$\theta_{JA} = \frac{T_J - T_A}{P}$$

and $\theta_{JA}$ physically represents the temperature differential between the die junction and the surrounding ambient at a power dissipation of 1 watt.

The junction temperature is given by the equation

$$T_J = T_A + P[\theta_{JA}] = T_A + P[\theta_{JC} + \theta_{CA}]$$

where

$$\theta_{JC} = \frac{T_J - T_C}{P} \quad \text{and} \quad \theta_{CA} = \frac{T_C - T_A}{P}$$

$T_A$ = Ambient temperature at which the device is operated; Most common standard temperature of operation is room temperature to 70°C.

$T_J$ = Junction temperature of the IC chip.

$T_C$ = Temperature of the case (package).

P = Power at which the device operates.

$\theta_{JC}$ = Junction-to-case thermal resistance. This is mainly a function of the thermal properties of the materials constituting the package.

$\theta_{JA}$ = Junction-to-ambient thermal resistance. The junction-to-ambient environment is a still-air environment.

$\theta_{CA}$ = Case-to-ambient thermal resistance. This is mainly dependent on the surface area available for convection and radiation and the ambient conditions among other factors. This can be controlled at the user end by using heat sinks providing greater surface area and better conduction path or by air or liquid cooling.

## Thermal Resistance: Finite Element Model

$\theta_{JC}$ and $\theta_{JA}$ values given in the following figures and listed in the following tables have been obtained by simulation using the Finite element software ANSYS[2]. SDRC-IDEAS Pre and Post processor software[3] was used to create the finite element model of the packages and the ANSYS input data required for analysis.

**Note:**

1. $T_{ambient}$ = 70°C
2. *ANSYS Finite Element Software User Guides*
3. *SDRC-IDEAS Pre and Post Processor User Guide*

SEMI Standard (Semiconductor Equipment and Materials International) method SEMI G30-88[4] states "heat sink" mounting technique to be the "reference" method for $\theta_{JC}$ estimation of ceramic packages. Accordingly, $\theta_{JC}$ of packages has been obtained by applying the boundary conditions that correspond to the heat sink mounted on the packages in the simulation.

For $\theta_{JA}$ evaluation, SEMI standard specification SEMI G38-87 suggests using a package-mounting arrangement that approximates the application environment. So, in evaluating the $\theta_{JA}$, package on-board configuration is assumed.

## Model Description

- One quarter of the package is mounted on a FR-4 PC board.
- Leads have been modeled as a continuous metallic plane, and equivalent thermal properties have been used to account for the plastic (or the glass in the case of ceramic packages) that fills the space between the leads.
- 1W power dissipation over the entire chip is assumed.
- 70°C ambient condition is considered.

## Comparison of Simulation Data with Measured Data

In the case of ceramic packages, it is not unusual to see significant differences in $\theta_{JC}$ values when a heat sink is used in the place of fluid bath.[5] However, SEMI G30-88 test method recommends the heat sink configuration for $\theta_{JC}$ evaluation.

$\theta_{JA}$ values from simulation compare within 12 percent of the measured values. $\theta_{JA}$ values obtained from simulation seem to be *conservative* with an accuracy of about +12 percent.

Measured values given in *Table 3* used the Temperature Sensitive Parameter method described in MIL STD 883C, method 1012.1. The junction-to-ambient measurement was made in a still-air environment where the device was inserted into a low-cost standard-device socket and mounted on a standard 0.062" G10 PC board.

**Table 3. 24-Lead Ceramic and Plastic DIPs**

| Package | Cavity/PAD Size (mils) | $\theta_{JA}$ (°C/W) | | |
|---|---|---|---|---|
| | | Measured | Simulation | % Diff. |
| 24LCDIP[6] | 170 x 270 | 64 | 67 | 5 |
| 24LPDIP[7] | 160 x 210 | 72 | 82 | 12 |

## Thermal Resistance of Packages with Forced Convection Air Flow

One of the methods adopted to cool the packages on PC boards at the system level is to used forced air (fans) specified in linear feet per minute or LFM. This helps reduce the device operating temperature by lowering the case to ambient thermal resistance. Available surface area of the package and the orientation of the package with respect to the air flow affect the reduction of thermal resistance that can be achieved. A general rule of thumb is:

- For plastic packages:
  - — 200 LFM air flow can reduce $\theta_{JA}$ by 20 to 25%
  - — 500 LFM air flow can reduce $\theta_{JA}$ by 30 to 40%
- For ceramic packages:
  - — 200 LFM air flow can reduce $\theta_{JA}$ by 25 to 30%
  - — 500 LFM air flow can reduce $\theta_{JA}$ by 35 to 45%

If $\theta_{JA}$ for a package in still air (no air flow) is known, approximate values of thermal resistance at 200 LFM and 500 LFM can be estimated. For estimation, the factors given in *Table 4* can be used as a guideline.

**Table 4. Factors for Estimating Thermal Resistance**

| Package Type | Air Flow Rate (LFM) | Multiplication Factor |
|---|---|---|
| Plastic | 200 | 0.77 |
| Plastic | 500 | 0.66 |
| Ceramic | 200 | 0.72 |
| Ceramic | 500 | 0.60 |

Example:

$\theta_{JA}$ for a plastic package in still air is given to be 80°C/W. Using the multiplication factor from *Table 4*:

- $\theta_{JA}$ at 200 LFM is (80 x 0.77) = 61.6°C/W
- $\theta_{JA}$ at 500 LFM is (80 x 0.66) = 52.8°C/W

$\theta_{JA}$ for a ceramic package in still air is given to be 70°C/W. Using *Table 4*:

- $\theta_{JA}$ at 200 LFM is (70 x 0.72) = 50.4°C/W
- $\theta_{JA}$ at 500 LFM is (70x0.60) = 42.0°C/W

## Presentation of Data

The following figures and tables present the data taken using the aforementioned procedures. The thermal resistance values of Cypress standard packages are graphically illustrated in *Figure 2* through *23.* Each envelope represents a spread of typical Cypress integrated circuit chip sizes (upper boundary=5000 mils$^2$, lower boundary = 100,000 mils$^2$) in their thermally optimized packaging environments. These graphs should be used in conjunction with *Table 10*, which lists the die sizes of Cypress devices.

*Table 5* through *9* give the thermal resistance values for other package types not included in the graphs. The letter in the header (*D, P, J,* etc.) of these tables refer to the package designators as detailed in the Package Diagrams section of this catalog. The numeric values given in the table (e.g., 20.3) refer to the lead count (20) and package width in inches (3). If no decimal appears, the reader must refer to the package diagrams.

**Notes:**

[4] *SEMI International Standards*, Vol. 4, Packaging Handbook, 1989.

[5] "Thermal resistance measurements and finite calculations for ceramic hermetic packages." James N. Sweet et. al., *SEMI-Therm*, 1990.

[6] 24LCDIP = 24-lead cerDIP

[7] 24LPDIP = 24-lead plastic DIP

## Packaging Materials

Cypress plastic packages incorporate:

- High thermal conductivity copper lead frame
- Molding compound with high thermal conductivity
- Gold bond wires

Cypress cerDIP packages incorporate

- High conductivity alumina substrates
- Silver-filled glass as die attach material
- Alloy 42-lead frame
- Aluminum bond wires
- Silver-filled conductive epoxy as die attach material

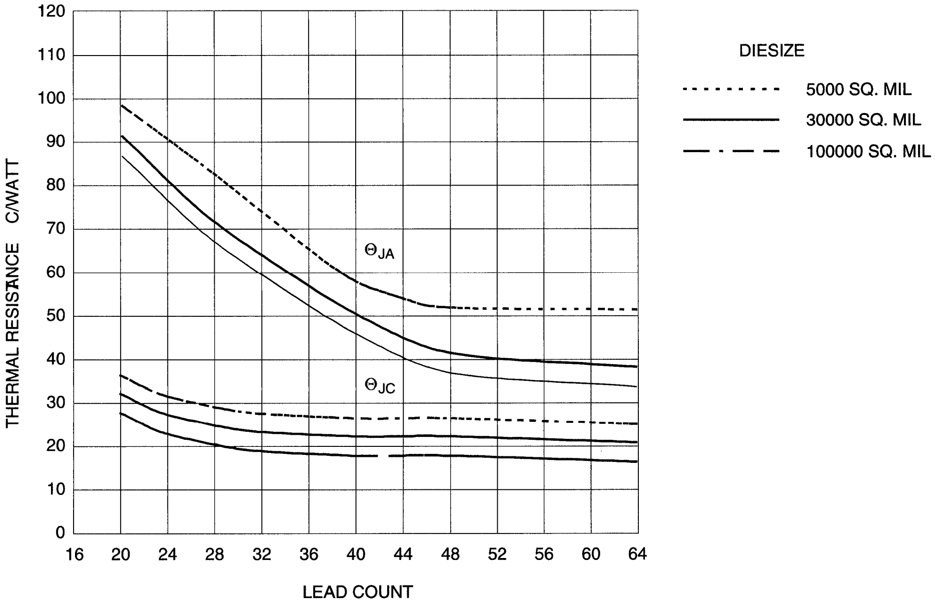


**Figure 2. Thermal Resistance of Cypress Plastic DIPs (Package type “P”)**

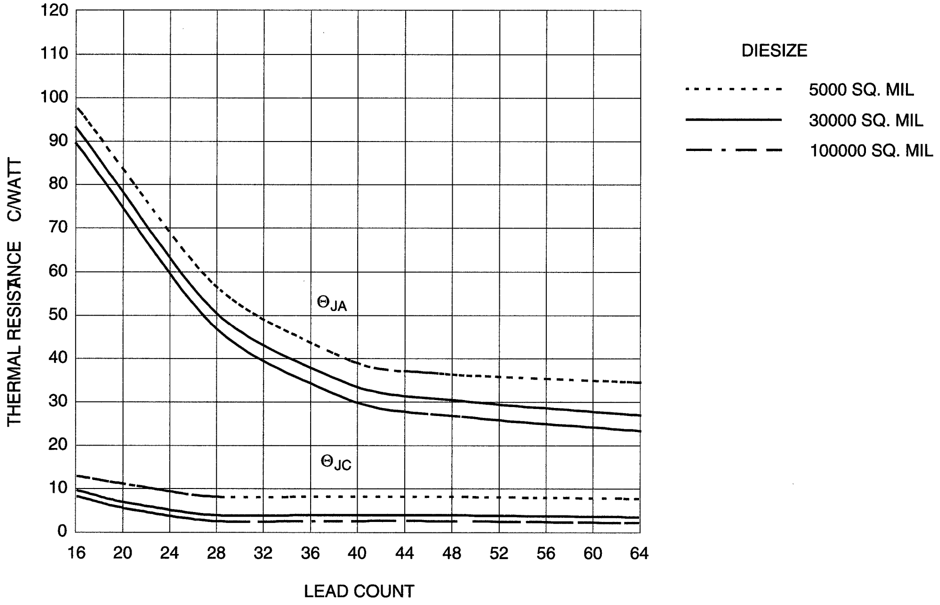


**Figure 3. Thermal Resistance of Cypress Ceramic DIPs (Package type "D" and "W")**

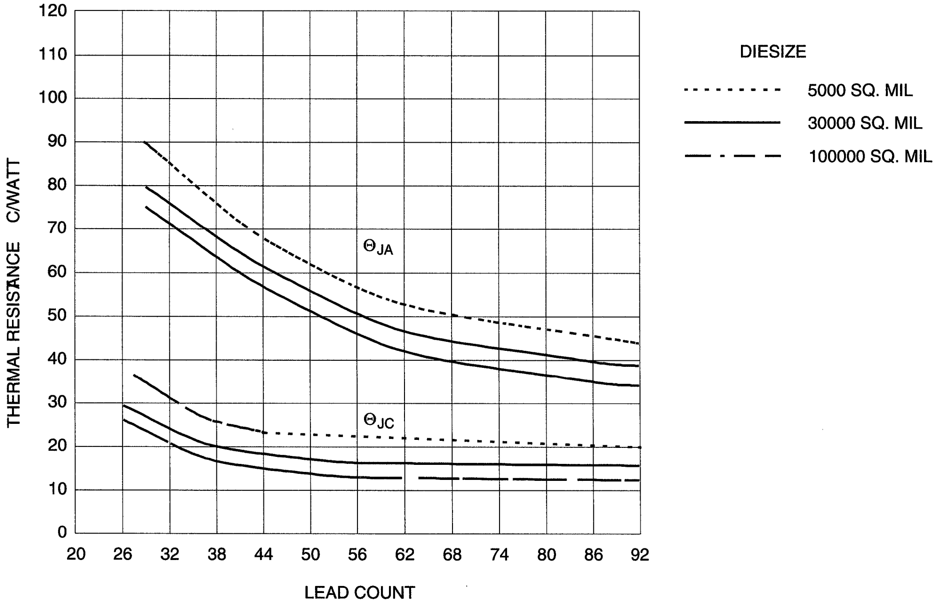


**Figure 4. Thermal Resistance of Cypress PLCCs (Package type "J")**

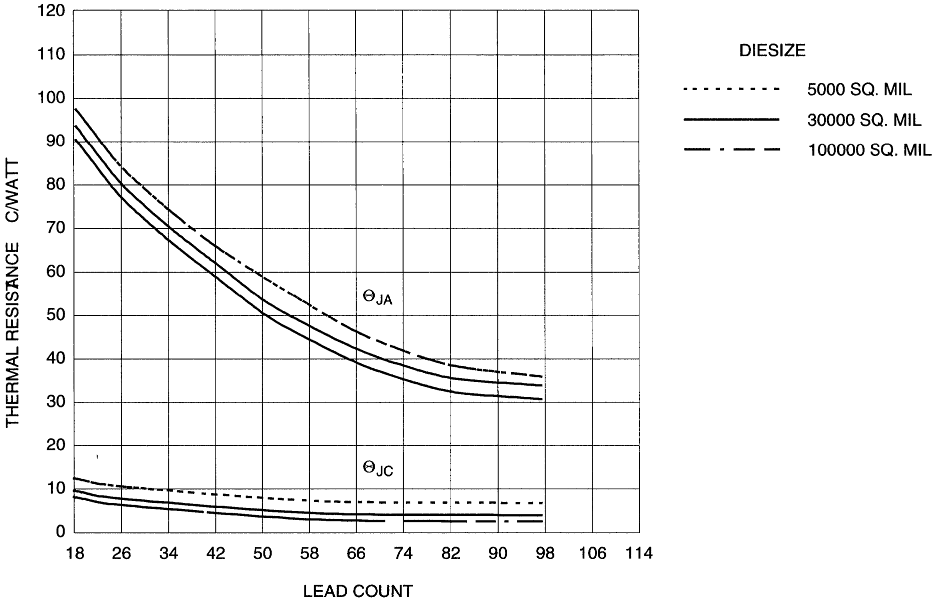


**Figure 5. Thermal Resistance of Cypress LCCs (Package type "L" and "Q")**

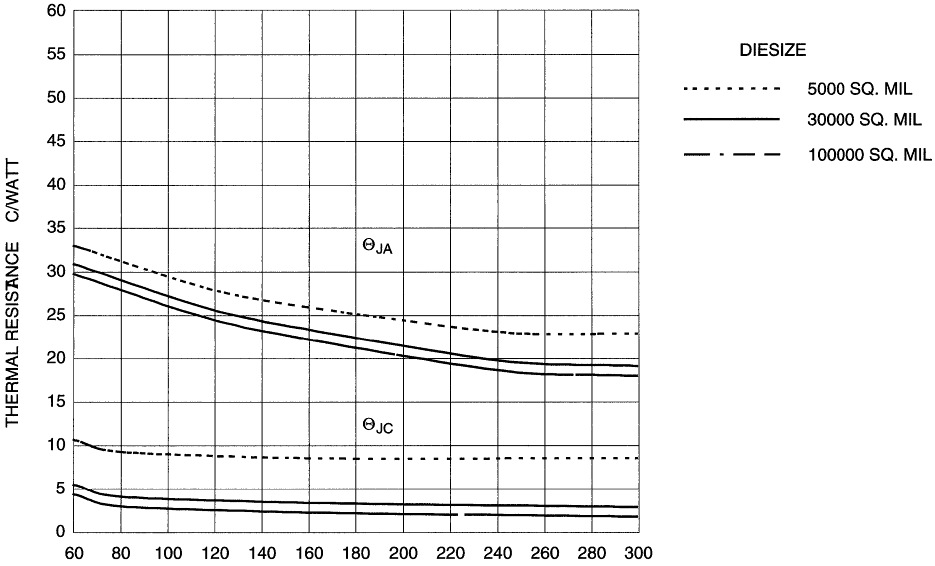


**Figure 6. Thermal Resistance of Cypress Ceramic PGAs**

Table 5. Plastic Surface Mount SOIC, SOJ[8,9]

| Package Type "S" and "V" | Paddle Size (mil) | LF Material | Die Size (mil) | Die Area (sq. mil) | $\theta_{JC}$ (°C/W) | $\theta_{JA}$ (°C/W still air) |
|---|---|---|---|---|---|---|
| 16 | 140 x 170 | Copper | 98 x 84 | 8,232 | 19.0 | 120 |
| 18 | 140 x 170 | Copper | 98 x 84 | 8,232 | 18.0 | 116 |
| 20 | 180 x 250 | Copper | 145 x 213 | 30,885 | 17.0 | 105 |
| 24 | 180 x 250 | Copper | 145 x 213 | 30,885 | 15.4 | 88 |
| 24 | 170 x 500 | Copper | 141 x 459 | 64,719 | 14.9 | 85 |
| 28 | 170 x 500 | Copper | 145 x 213 | 30,885 | 16.7 | 84 |
| 28 | 170 x 500 | Copper | 141 x 459 | 64,719 | 14.4 | 80 |

Table 6. Plastic Quad Flatpacks

| Package Type "N" | LF Material | Paddle Size (mil) | Die Size (mil) | $\theta_{JC}$ (°C/W) | $\theta_{JA}$ (°C/W still air) |
|---|---|---|---|---|---|
| 100 | Copper | 310 x 310 | 235 x 235 | 17 | 51 |
| 144 | Copper | 310 x 310 | 235 x 235 | 18 | 41 |
| 160 | Copper | 310 x 310 | 230 x 230 | 18 | 40 |
| 184 | Copper | 460 x 460 | 322 x 311 | 15 | 38.5 |
| 208 | Copper | 400 x 400 | 290 x 320 | 16 | 39 |

Table 7. Ceramic Quad Flatpacks

| Package Type "H" and "Y" | Cavity Size (mil) | LF Material | Die Size (mil) | Die Area (sq. mil) | $\theta_{JC}$ (°C/W) | $\theta_{JA}$ (°C/W still air) |
|---|---|---|---|---|---|---|
| 28 | 250 x 250 | Alloy 42 | 123 x 162 | 19,926 | 9.2 | 96 |
| 28 | 250 x 250 | Alloy 42 | 150 x 180 | 27,000 | 8.9 | 93 |
| 32 | 316 x 317 | Alloy 42 | 198 x 240 | 47,520 | 7.5 | 72 |
| 44 | 400 x 400 | Alloy 42 | 310 x 250 | 77,500 | 5.9 | 55 |
| 52 | 400 x 400 | Alloy 42 | 250 x 310 | 77,500 | 5.9 | 55 |
| 68 | 400 x 400 | Alloy 42 | 310 x 250 | 77,500 | 5.4 | 33 |
| 84 | 450 x 450 | Alloy 42 | 310 x 250 | 77,500 | 5.4 | 29 |

Table 8. Cerpacks

| Package Type "K" and "T" | Cavity Size (mil) | Leadframe Material | Die Size (mil) | Die Area (sq. mil) | $\theta_{JC}$ (°C/W) | $\theta_{JA}$ (°C/W still air) |
|---|---|---|---|---|---|---|
| 16 | 140 x 200 | Alloy 42 | 100 x 118 | 11,800 | 10 | 107 |
| 18 | 140 x 200 | Alloy 42 | 100 x 118 | 11,800 | 10 | 104 |
| 20 | 180 x 265 | Alloy 42 | 128 x 170 | 21,760 | 9 | 102 |
| 24 | 170 x 270 | Alloy 42 | 128 x 170 | 21,760 | 10 | 102 |
| 28 | 210 x 210 | Alloy 42 | 150 x 180 | 27,000 | 9 | 98 |
| 32 | 210 x 550 | Alloy 42 | 141 x 459 | 64,719 | 7 | 81 |

**Note:**

[8] The data in *Table 6* was simulated for SOIC packaging.

[9] SOICs and SOJs have very similar thermal resistance characteristics. The thermal resistance values given above apply to SOJ packages also.

**Table 9. Miscellaneous Packaging**

| Package Type | Cavity Size (mil) | Leadframe Material | Die Size (mil) | Die Area (sq. mil) | $\theta_{JC}$ (°C/W) | $\theta_{JA}$ (°C/W still air) |
|---|---|---|---|---|---|---|
| 24 VDIP[10] | 500 x 275 | Alloy 42 | 145 x 213 | 30,885 | 6 | 57 |
| 68 CPGA[11] | 350 x 350 | Kovar Pins | 323 x 273 | 88,179 | 3 | 28 |

**Note:**

[10] VDIP = "PV" package.

[11] CPGA = "G" package.

**Table 10. Die Sizes of Cypress Devices**

| Part Number | Size (mil$^2$) |
|---|---|
| **SRAMs** | |
| CY2147 | 10132 |
| CY2148 | 9983 |
| CY2149 | 9983 |
| CY27LS03 | 4130 |
| CY27S03A | 4130 |
| CY27S07A | 4130 |
| CY6116 | 20007 |
| CY6116A | 20007 |
| CY6117 | 20007 |
| CY6117A | 20007 |
| CY74S189 | 4130 |
| CY7B134 | 76152 |
| CY7B1342 | 76152 |
| CY7B135 | 76152 |
| CY7B138 | 76152 |
| CY7B139 | 76152 |
| CY7B144 | 76152 |
| CY7B145 | 76152 |
| CY7B160 | 27244 |
| CY7B161 | 27244 |
| CY7B162 | 27244 |
| CY7B164 | 27244 |
| CY7B166 | 27244 |
| CY7B173 | 102200 |
| CY7B174 | 102200 |
| CY7B180 | 54600 |
| CY7B181 | 54600 |
| CY7B185 | 27244 |
| CY7B186 | 27244 |
| CY7B191 | 73152 |
| CY7B192 | 73152 |
| CY7B194 | 73152 |
| CY7C122 | 6300 |
| CY7C123 | 6300 |
| CY7C128 | 20007 |
| CY7C128A | 17400 |
| CY7C130 | 36636 |
| CY7C131 | 36636 |
| CY7C132 | 36636 |
| CY7C136 | 36636 |
| CY7C140 | 36636 |
| CY7C141 | 36636 |
| CY7C142 | 36636 |
| CY7C146 | 36636 |
| CY7C147 | 10132 |
| CY7C148 | 9983 |
| CY7C149 | 9983 |
| CY7C150 | 6634 |
| CY7C157 | 86460 |
| CY7C161A | 30885 |
| CY7C162A | 30885 |
| CY7C164A | 30885 |
| CY7C166A | 30885 |
| CY7C167 | 21228 |
| CY7C167A | 21228 |
| CY7C168 | 21228 |
| CY7C168A | 21228 |
| CY7C169 | 21228 |
| CY7C169A | 21228 |
| CY7C170 | 21228 |
| CY7C170A | 21228 |
| CY7C171 | 21228 |
| CY7C171A | 21228 |
| CY7C172 | 21228 |
| CY7C172A | 21228 |
| CY7C183 | 65636 |
| CY7C184 | 65636 |
| CY7C185 | 30885 |
| CY7C186 | 30885 |
| CY7C187 | 30885 |
| CY7C189 | 4130 |

**Table 10. Die Sizes of Cypress Devices** (continued)

| Part Number | Size (mil$^2$) |
|---|---|
| CY7C190 | 4130 |
| CY7C191 | 68150 |
| CY7C192 | 68150 |
| CY7C194 | 68150 |
| CY7C196 | 68150 |
| CY7C197 | 68150 |
| CY7C198 | 68150 |
| CY7C199 | 68150 |
| CY7C191 (RAM2.5) | 51590 |
| CY7C192 (RAM2.5) | 51590 |
| CY7C194 (RAM2.5) | 51590 |
| CY7C196 (RAM2.5) | 51590 |
| CY7C197 (RAM2.5) | 51590 |
| CY7C198 (RAM2.5) | 51590 |
| CY7C199 (RAM2.5) | 51590 |
| CY7C9122 | 6300 |
| CY93422A | 6300 |
| **PROMs** | |
| CY7C225 | 11815 |
| CY7C235 | 13900 |
| CY7C245 | 19321 |
| CY7C245A | 9394 |
| CY7C251 | 49536 |
| CY7C254 | 49536 |
| CY7C261 | 28290 |
| CY7C263 | 28290 |
| CY7C264 | 28290 |
| CY7C265 | 28290 |
| CY7C266 | 28290 |
| CY7C268 | 29400 |
| CY7C269 | 29400 |
| CY7C271 | 38750 |
| CY7C274 | 38750 |
| CY7C277 | 38750 |
| CY7C279 | 38750 |
| CY7C281 | 13900 |
| CY7C282 | 13900 |
| CY7C285 | 43875 |
| CY7C286 | 43875 |
| CY7C287 | 43875 |
| CY7C289 | 43875 |
| CY7C291 | 19182 |
| CY7C291A | 9394 |
| CY7C292 | 19321 |

**Table 10. Die Sizes of Cypress Devices** (continued)

| Part Number | Size (mil$^2$) |
|---|---|
| CY7C292A | 9394 |
| CY7C293A | 9394 |
| **PLDs** | |
| CY7C330 | 20088 |
| CY7C331 | 16536 |
| CY7C332 | 19116 |
| CY7C335 | 23111 |
| CY7C341 | 136320 |
| CY7C342 | 83475 |
| CY7C342B | 49104 |
| CY7C343 | 43953 |
| CY7C344 | 21977 |
| CY7C361 | 25872 |
| PAL16L8 | 13552 |
| PAL16R4 | 13552 |
| PAL16R6 | 13552 |
| PAL16R8 | 13552 |
| PAL22V10C | 18834 |
| PAL22VP10C | 18834 |
| PALC16L8 | 9700 |
| PALC16R4 | 9700 |
| PALC16R6 | 9700 |
| PALC16R8 | 9700 |
| PALC22V10 | 19926 |
| PALC22V10B | 13284 |
| PALC22V10D | 12954 |
| PLD20G10C | 18834 |
| PLDC18G8 | 7744 |
| PLDC20G10 | 19926 |
| PLDC20G10B | 13284 |
| PLDC20RA10 | 13284 |
| **FIFOs** | |
| CY3341 | 8064 |
| CY7C401 | 8064 |
| CY7C402 | 8064 |
| CY7C403 | 8064 |
| CY7C404 | 8064 |
| CY7C408A | 16268 |
| CY7C409A | 16268 |
| CY7C420 | 41019 |
| CY7C421 | 41019 |
| CY7C424 | 41019 |
| CY7C425 | 41019 |
| CY7C428 | 41019 |

**Table 10. Die Sizes of Cypress Devices** (continued)

| Part Number | Size (mil$^2$) |
|---|---|
| CY7C429 | 41019 |
| CY7C432 | 50040 |
| CY7C433 | 50040 |
| CY7C439 | 47160 |
| CY7C441 | 44756 |
| CY7C443 | 44756 |
| CY7C451 | 44756 |
| CY7C453 | 44756 |
| CY7C460 | 89445 |
| CY7C462 | 89445 |
| CY7C464 | 89445 |
| CY7C470 | 89445 |
| CY7C472 | 89445 |
| CY7C474 | 89445 |
| **Logic** | |
| CY2909A | 7968 |
| CY2910A | 21750 |
| CY2911A | 7968 |
| CY7C2901 | 11800 |
| CY7C510 | 30704 |
| CY7C516 | 29000 |
| CY7C517 | 29000 |
| CY7C901 | 11800 |
| CY7C909 | 7968 |
| CY7C910 | 21750 |
| CY7C9101 | 36108 |
| CY7C911 | 7968 |
| **ECL** | |
| CY100E301L | 14875 |
| CY100E302L | 14875 |
| CY100E422 | 6960 |
| CY100E474 | 10830 |
| CY100E494 | 29575 |
| CY10E301L | 14875 |
| CY10E302L | 14875 |
| CY10E422 | 6960 |
| CY10E474 | 10830 |
| CY10E494 | 29575 |
| **Bus Interface** | |
| CY7C964 | 21460 |
| VAC068 | 101060 |
| VIC068A | 103620 |
| VIC64 | 103620 |

Document #: 38-00190

# Tape and Reel Specifications

## Description

Surface-mounted devices are packaged in embossed tape and wound onto reels for shipment in compliance with Electronics Industries Association Standard EIA-481 Rev. A.

## Specifications

### Cover Tape

- The cover tape may not extend past the edge of the carrier tapes
- The cover tape shall not cover any part of any sprocket hole.
- The seal of the cover tape to the carrier tape is uniform, with the seal extending over 100% of the length of each pocket, on each side.
- The force to peel back the cover tape from the carrier tape shall be: 20 gms minimal, 70 gms nominal, 100 gms maximal, at a pull-back speed of 300 ± 10 mm/min.

### Loading the Reel

Empty pockets are not permitted between the first and last filled pockets on the tape.

The surface-mount devices are placed in the carrier tape with the leads down, as shown in *Figure 1*.

### Leaders and Trailers

The carrier tape and the cover tape may not be spliced. Both tapes must be one single uninterrupted piece from end to end.

Both ends of the tape must have empty pockets meeting the following minimum requirements:

- Trailer end (inside hub of reel) is 300 mm minimum
- Leader end (outside of reel) is 500 mm min., 560 mm max.
- Unfilled leader and trailer pockets are sealed
- Leaders and trailers are taped to tape and hub respectively using masking tape

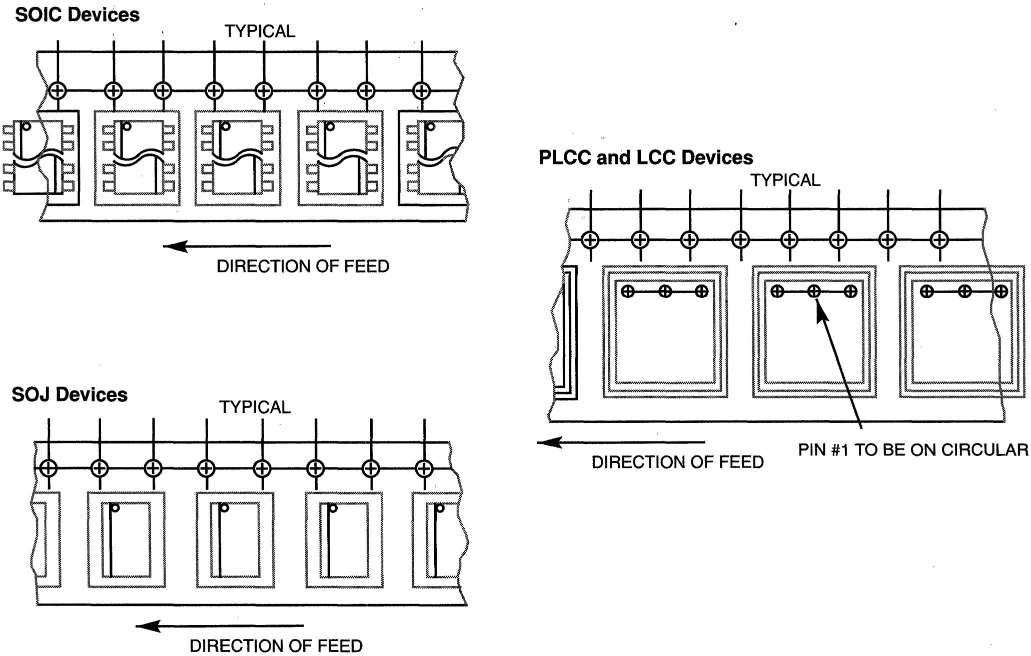


**Figure 1. Part Orientation in Carrier Tape**

### Packaging

- Full reels contain a standard number of units (refer to *Table 1*). Each reel has a 3” hub unless otherwise noted in table.
- Reels may contain up to 4 mark lots. However, in case low yield is encountered an additional lot may be added.
- Each reel is packed in an anti-static bag and then in its own individual box.
- Labels are placed on each reel as shown in *Figure 2*. The information on the label consists of a minimum of the following information, which complies with EIA 556, “Shipping and Receiving Transaction Bar Code Label Standard”:
  - — Barcoded Information:
    Customer PO number
    Quantity
    Date code
  - — Human Readable Only:
    Package count (number of reels per order)
    Description
    “Cypress-San Jose”
    Cypress p/n
    Cypress CS number (if applicable)
    Customer p/n
- Each box will contain an identical label plus an ESD warning label.

## Ordering Information

CY7Cxxx-yyzzz

xxx = part type

yy = speed

zzz = package, temperature, and options

SCT = soic, commercial temperature range
SIT = soic, industrial temperature range
SCR = soic, commercial temperature plus burn-in
SIR = soic, industrial temperature plus burn-in
VCT = soj, commercial temperature range
VIT = soj, industrial temperature range
VCR = soj, commercial temperature plus burn-in
VIR = soj, industrial temperature plus burn-in
JCT = plcc, commercial temperature range
JIT = plcc, industrial temperature range
JCR = plcc, commercial temperature range plus burn-in
JIR = plcc, industrial temperature range plus burn-in

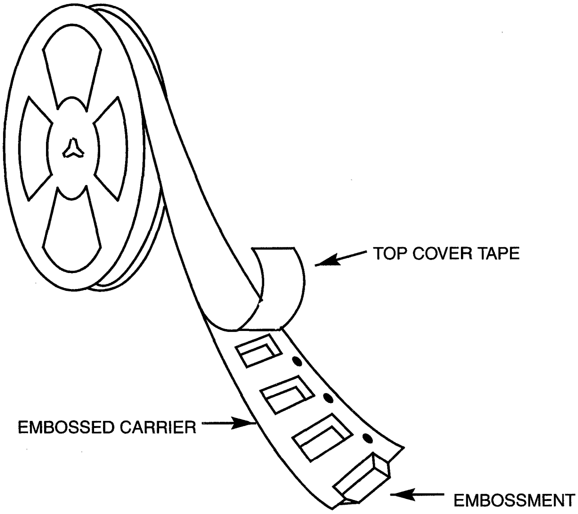
Notes:

1. The T or R suffix will not be marked on the device. Units will be marked the same as parts in a tube.
2. Order releases must be in full-reel multiples as listed in *Table 1.*

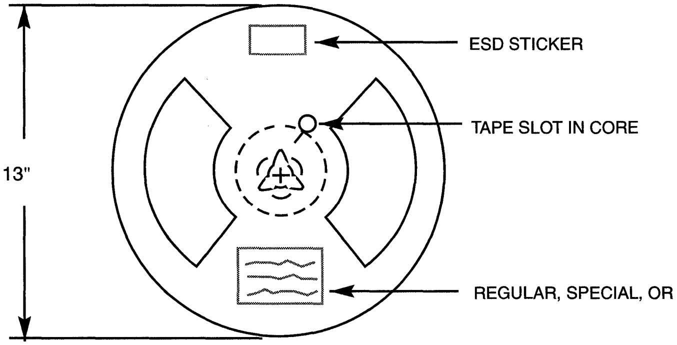
**Table 1. Parts Per Reel and Tape Specifications**

| Package Type | Terminals | Carrier Width (mm) | Part Pitch (mm) | Parts Per Full Reel | Minimum Partial Quantity |
|---|---|---|---|---|---|
| PLCC | 20 | 16 | 12 | 1,000 | 250 |
| | 28 | 24 | 16 | 750 | 188 |
| | 32R | 24 | 16 | 750 | 188 |
| | 44 | 32 | 24 | 500 | 125 |
| | 52 | 32 | 24 | 350 (Uses 7” hub) | 125 |
| | 68 | 44 | 32 | 250 | 63 |
| | 84 | 44 | 32 | 250 | 63 |
| SOIC | 16 | 16 | 8 | 2,500 | 625 |
| | 20 | 24 | 12 | 1,000 | 250 |
| | 24 | 24 | 12 | 1,000 | 250 |
| | 28 | 24 | 12 | 1,000 | 250 |
| SOJ | 20 | 24 | 12 | 1,000 | 250 |
| | 24 | 24 | 12 | 1,000 | 250 |
| | 28 | 24 | 12 | 1,000 | 250 |
| | 32L (300 mil) | 24 | 16 | 1,000 | 250 |
| | 32 (400 mil) | 32 | 16 | 750 | 188 |
| TSOP | 28L | 24 | 12 | 1,500 | 375 |
| | 32 | 32 | 16 | 1,500 | 375 |
| SSOP | 20 (150 mil) | 16 | 8 | 2,000 | 500 |
| | 24 (150 mil) | 16 | 8 | 2,000 | 500 |
| | 48 (300 mil) | 32 | 16 | 1,000 | 250 |
| | 56 (300 mil) | 32 | 16 | 1,000 | 250 |
| TSSOP | 48 | 24 | 12 | 2,000 | 500 |
| | 56 | 24 | 12 | 2,000 | 500 |

6



**Tape and Reel Shipping Medium**



**Label Placement**

**Figure 2. Shipping Medium and Label Placement**

CYPRESS

GENERAL INFORMATION 1

SMALL PLDS 2

CPLDs 3

DEVELOPMENT TOOLS 4

APPLICATION NOTES 5

QUALITY 6

**PACKAGES** 7

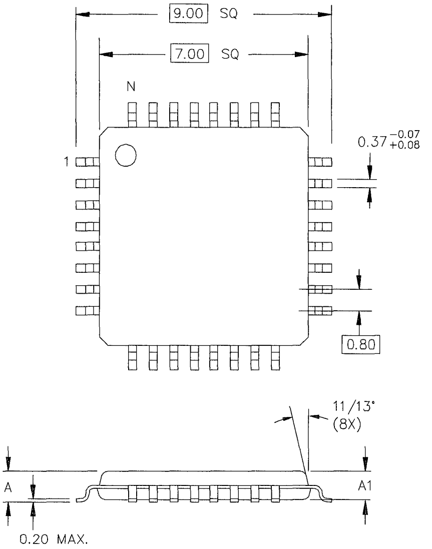
**Table of Contents** **Page Number**

**Packages**

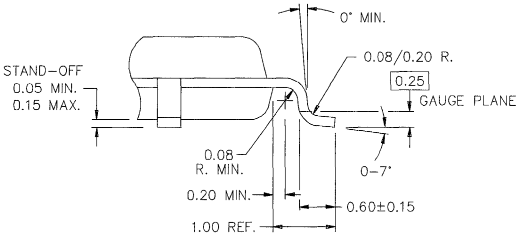
Thin Quad Flat Packs ........ 7-1
Plastic Pin Grid Arrays ........ 7-8
Ceramic Dual-In-Line Packages ........ 7-9
Ceramic Flatpacks ........ 7-20
Ceramic Pin Grid Arrays ........ 7-24
Ceramic Windowed J-Leaded Chip Carriers ........ 7-30
Plastic Leaded Chip Carriers ........ 7-36
Cerpacks ........ 7-40
Ceramic Leadless Chip Carriers ........ 7-44
Plastic Quad Flatpacks ........ 7-51
Shrunk Small Outline Packages ........ 7-62
Plastic Dual-In-Line Packages ........ 7-65
Quarter Size Outline Packages ........ 7-71
Ceramic Windowed Leadless Chip Carriers ........ 7-73
Ceramic Windowed Pin Grid Arrays ........ 7-75
Reverse Thin Small Outline Package ........ 7-77
Plastic Small Outline ICs ........ 7-78
Windowed Cerpacks ........ 7-85
Ceramic Quad Flatpacks ........ 7-87
Plastic Small Outline J-Bend ........ 7-90
Ceramic Windowed Dual-In-Line Packages ........ 7-94
Ceramic J-Leaded Chip Carriers ........ 7-98
Thin Small Outline Packages ........ 7-103

## Thin Quad Flat Packs

### 32-Lead Plastic Thin Quad Flat Pack (TQFP) A32

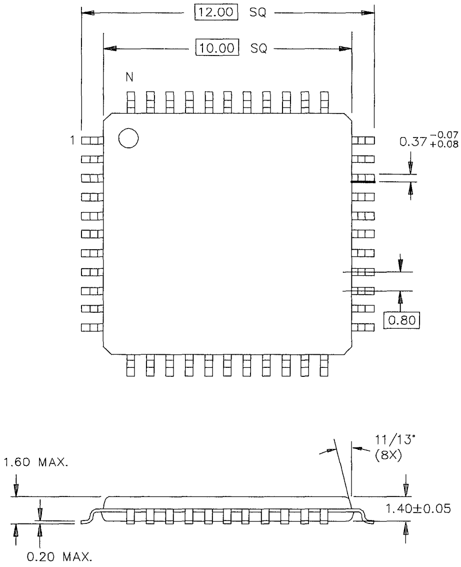


DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.080 MAX.

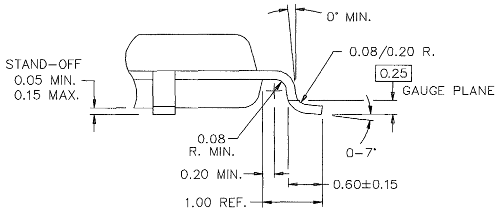


| DIM. A | DIM. A1 |
|---|---|
| 1.60 MAX. | 1.40±0.05 PKG. THICK |
| 1.20 MAX. | 1.00±0.05 PKG. THICK |

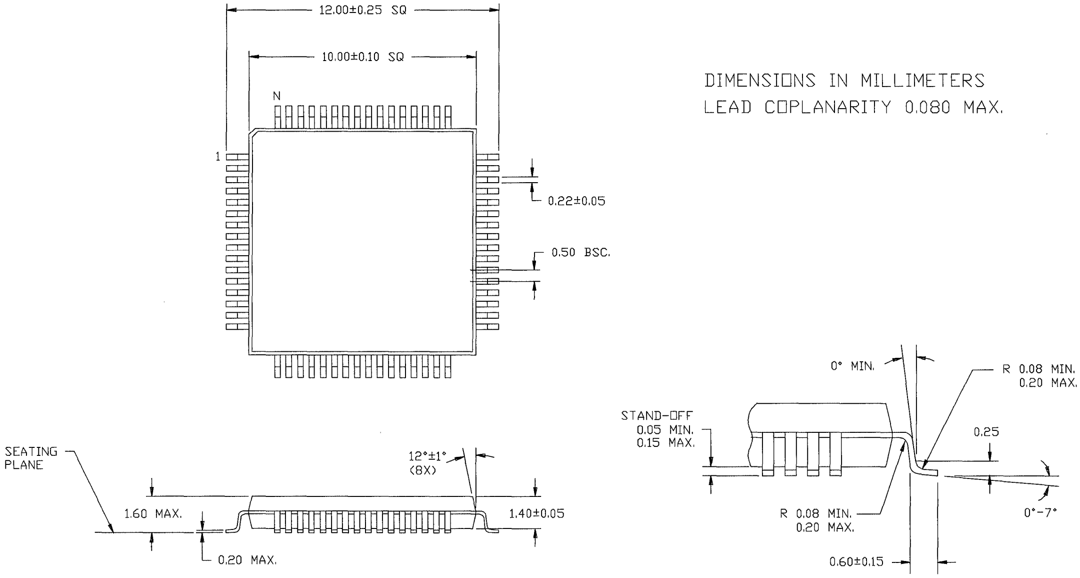
### 44-Lead Thin Plastic Quad Flat Pack A44



DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.080 MAX.

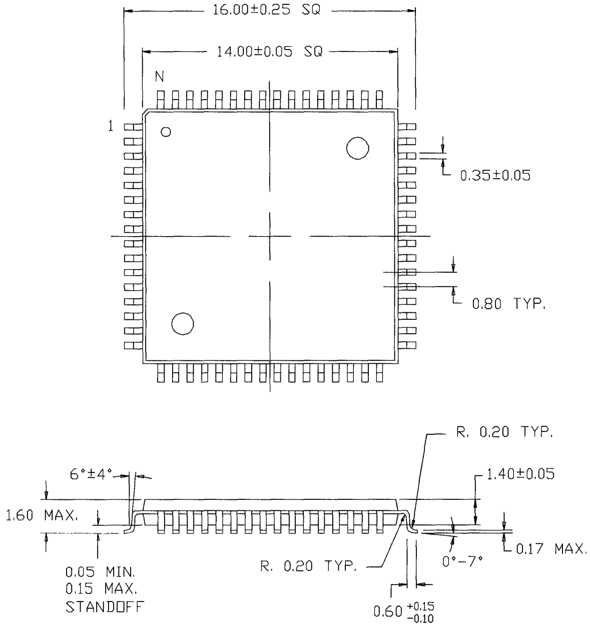


## 64-Pin Thin Quad Flat Pack A64



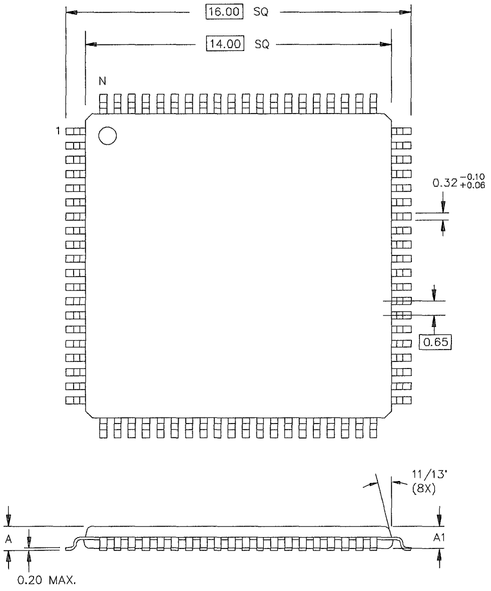
DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.080 MAX.

### 64-Lead Thin Plastic Quad Flat Pack A65

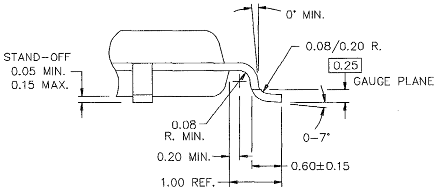


DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.100 MAX.

## 80-Pin Thin Plastic Quad Flat Pack A80

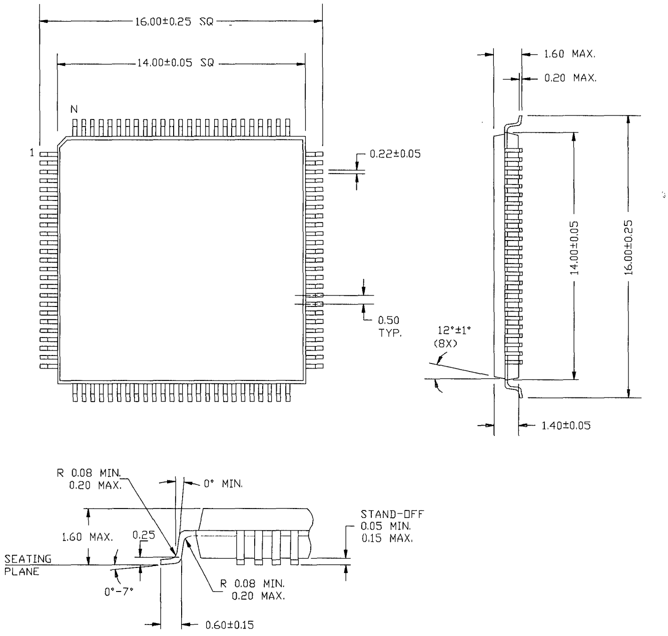


DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.080 MAX.

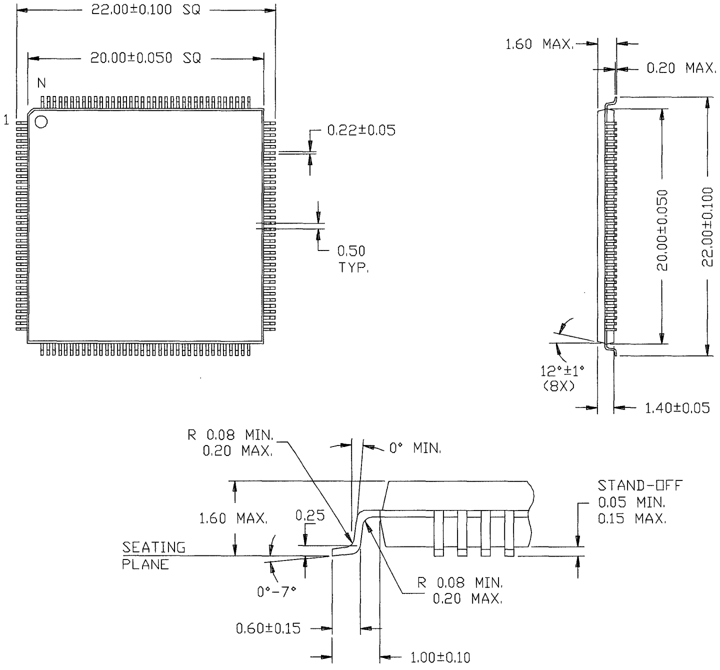


| DIM. A | DIM. A1 |
|---|---|
| 1.60 MAX. | 1.40±0.05 PKG. THICK |
| 1.20 MAX. | 1.00±0.05 PKG. THICK |

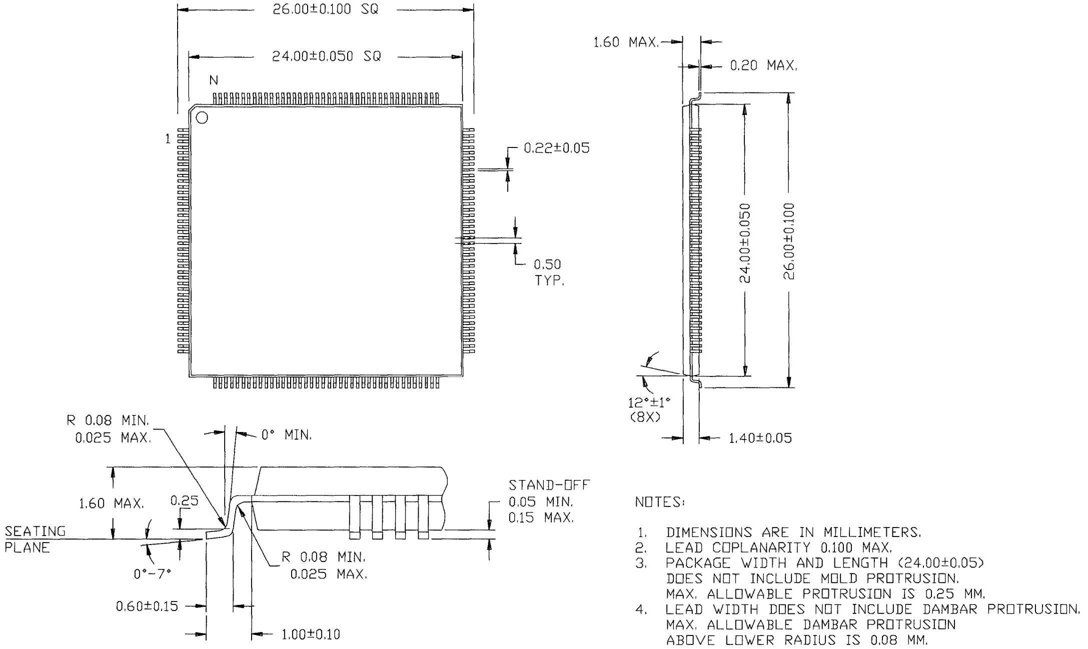
### 100-Pin Plastic Thin Quad Flat Pack (TQFP) A100



### 144-Pin Plastic Thin Quad Flat Pack (TQFP) A144



## 160-Pin Plastic Thin Quad Flat Pack (TQFP) A160

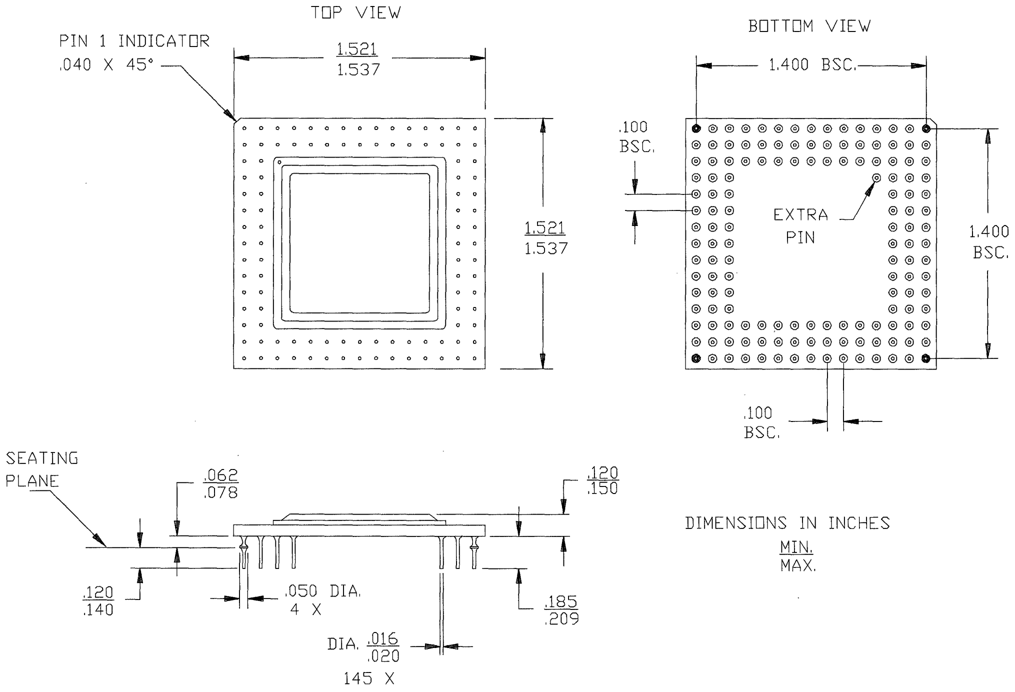


NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.
2. LEAD COPLANARITY 0.100 MAX.
3. PACKAGE WIDTH AND LENGTH (24.00±0.05) DOES NOT INCLUDE MOLD PROTRUSION. MAX. ALLOWABLE PROTRUSION IS 0.25 MM.
4. LEAD WIDTH DOES NOT INCLUDE DAMBAR PROTRUSION. MAX. ALLOWABLE DAMBAR PROTRUSION ABOVE LOWER RADIUS IS 0.08 MM.

## Plastic Pin Grid Arrays

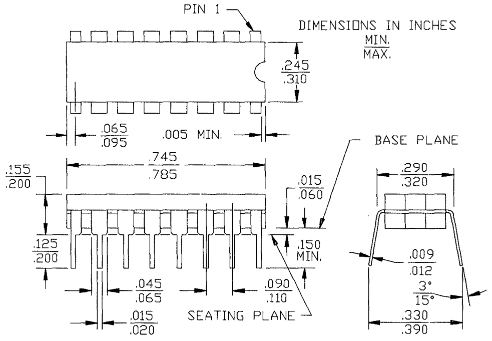
### 145-Pin Plastic Grid Array (Cavity Up) B144



## Ceramic Dual-In-Line Packages

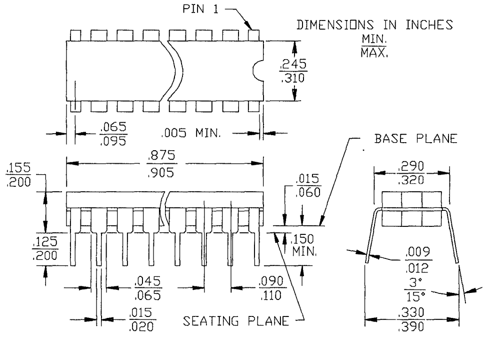
### 16-Lead (300-Mil) CerDIP D2

MIL-STD-1835 D-2 Config. A



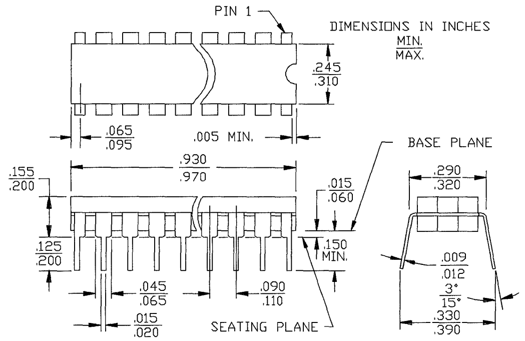
### 18-Lead (300-Mil) CerDIP D4

MIL-STD-1835 D-6 Config. A



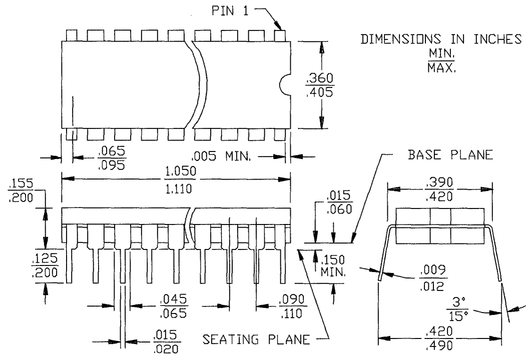
### 20-Lead (300-Mil) CerDIP D6

MIL-STD-1835 D-8 Config. A

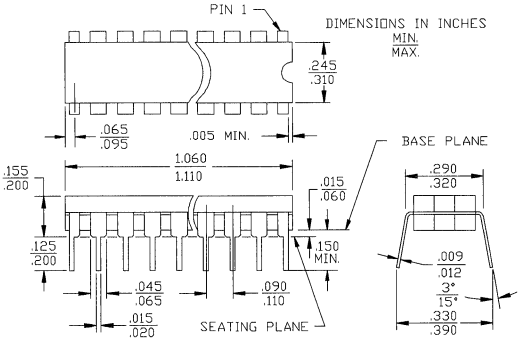


### 22-Lead (400-Mil) CerDIP D8

MIL-STD-1835 D-7 Config. A

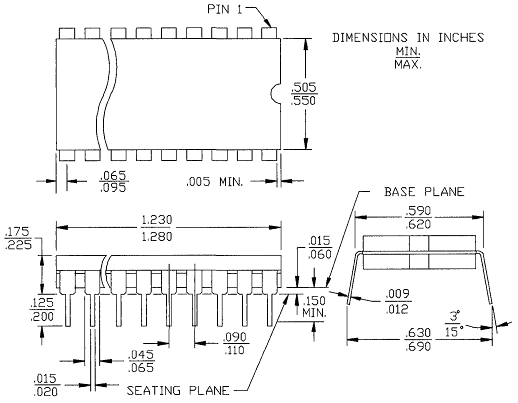


### 22-Lead (300-Mil) CerDIP D10



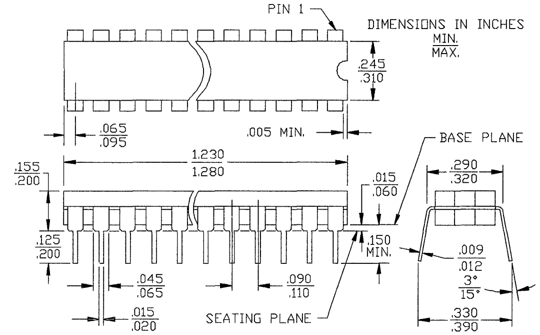
### 24-Lead (600-Mil) CerDIP D12

MIL-STD-1835 D-3 Config. A



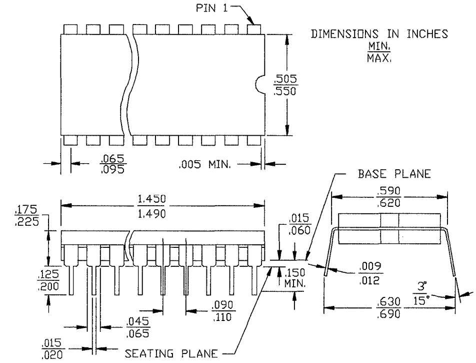
### 24-Lead (300-Mil) CerDIP D14

MIL-STD-1835 D-9 Config.A



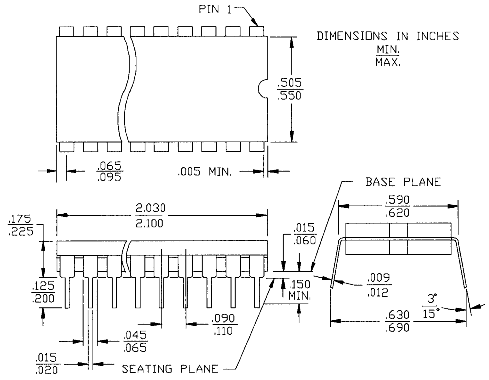
### 28-Lead (600-Mil) CerDIP D16

MIL-STD-1835 D-10 Config .A

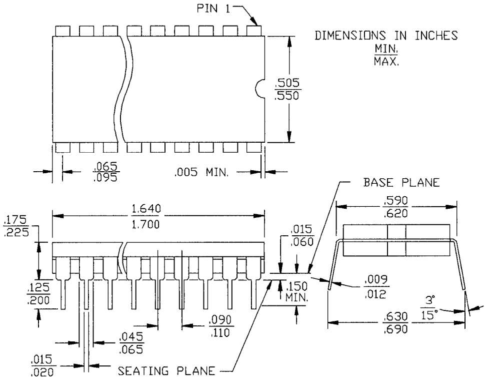


### 40-Lead (600-Mil) CerDIP D18

MIL-STD-1835 D-5 Config. A

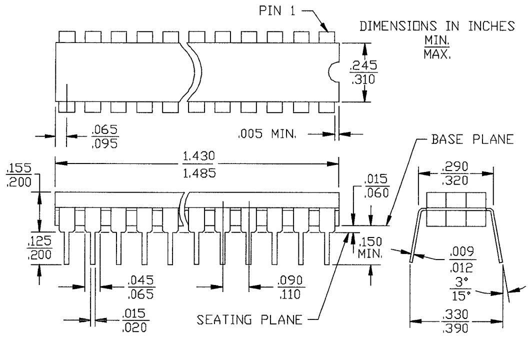


### 32-Lead (600-Mil) CerDIP D20



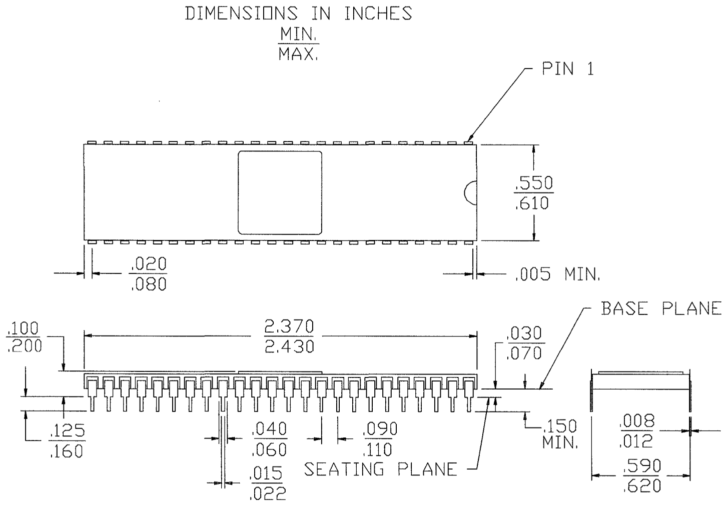
## 28-Lead (300-Mil) CerDIP D22

MIL-STD-1835 D-15 Config. A

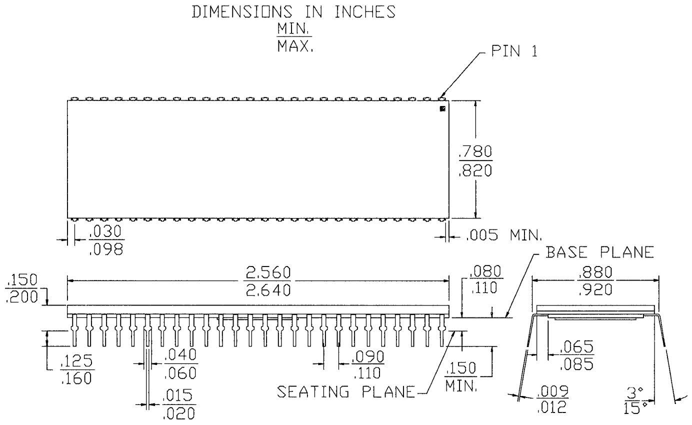


## 48-Lead (600-Mil) Sidebraze DIP D26

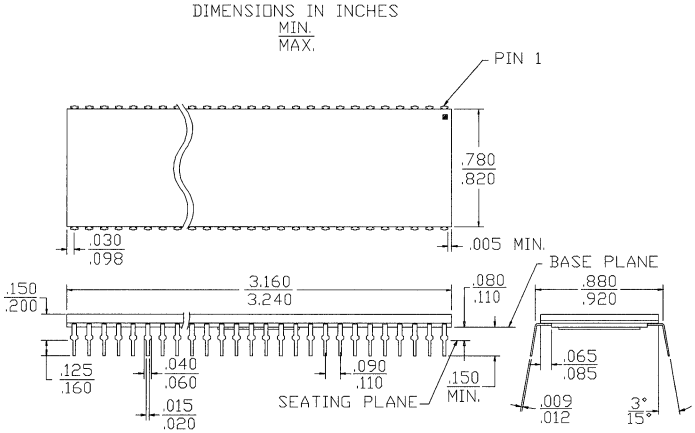
MIL-STD-1835 D-14 Config. C



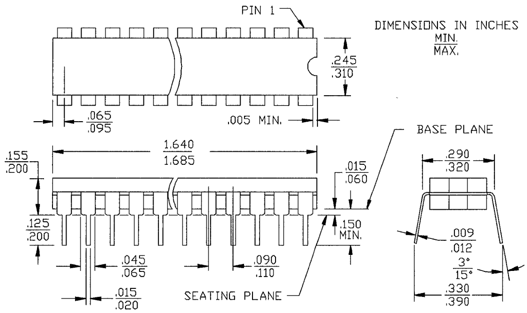
### 52-Lead (900-Mil) Bottombraze DIP D28



### 64-Lead (900-Mil) Bottombraze DIP D30

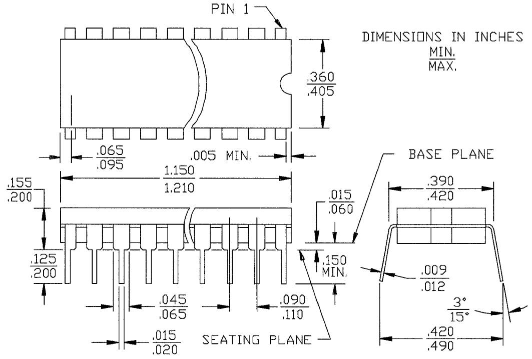


### 32-Lead (300-Mil) CerDIP D32

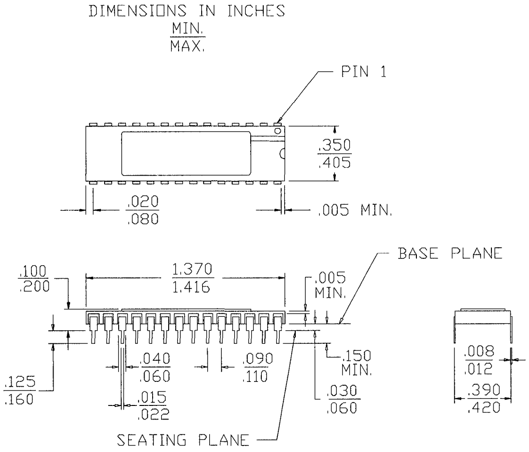


### 24-Pin (400-Mil) Sidebraze DIP D40

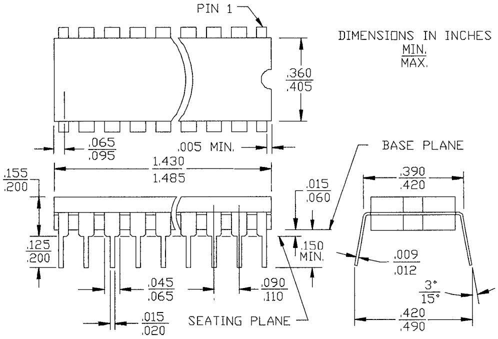
MIL-STD-1835 D-11 Config. A



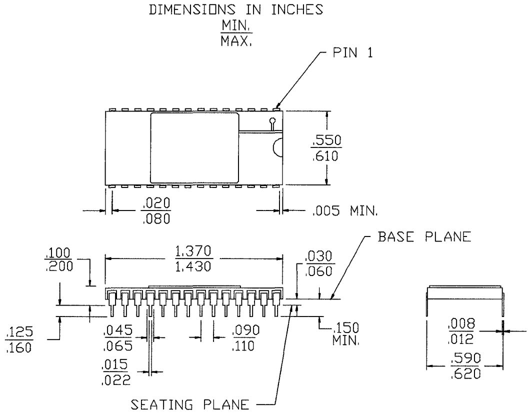
### 28-Lead (400-Mil) Sidebraze DIP D41



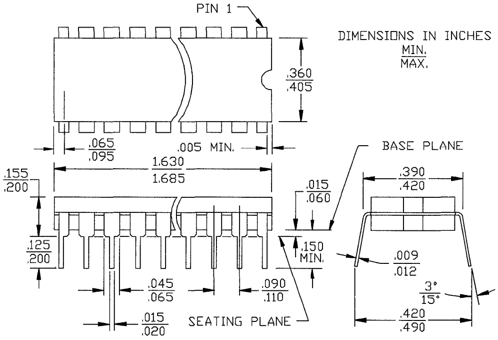
### 28-Lead (400-Mil) CerDIP D42



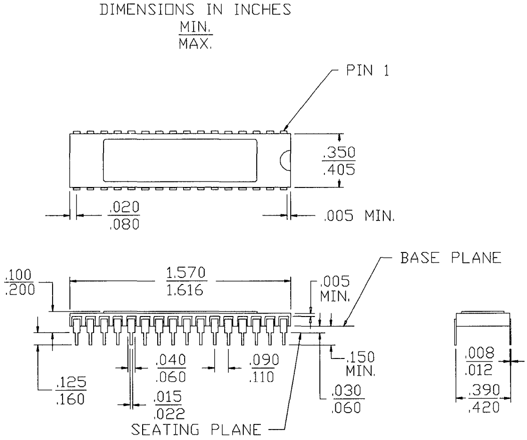
## 28-Lead (600-Mil) Sidebraze DIP D43



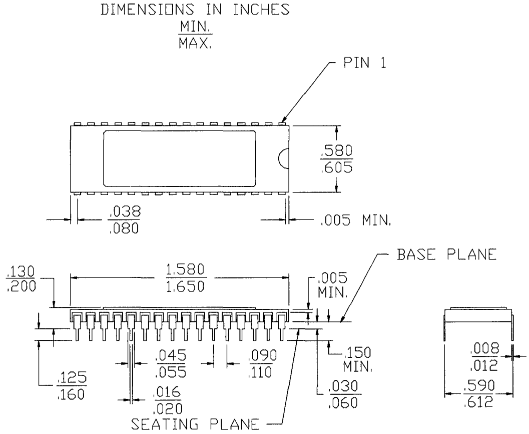
## 32-Lead (400-Mil) CerDIP D44



### 32-Lead (400-Mil) Sidebraze DIP D46



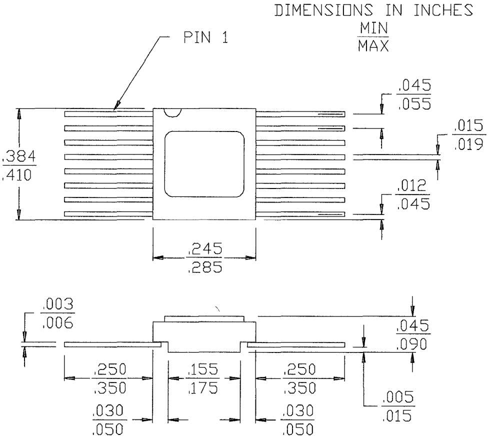
### 32-Lead (600-Mil) Sidebraze DIP D50



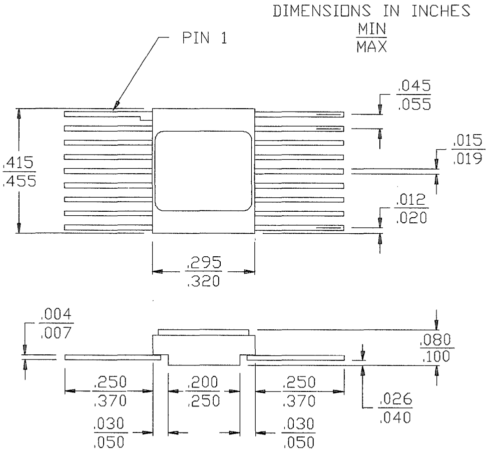
## Ceramic Flatpacks

### 16-Lead Rectangular Flatpack F69

MIL-STD-1835 F-5 Config. B

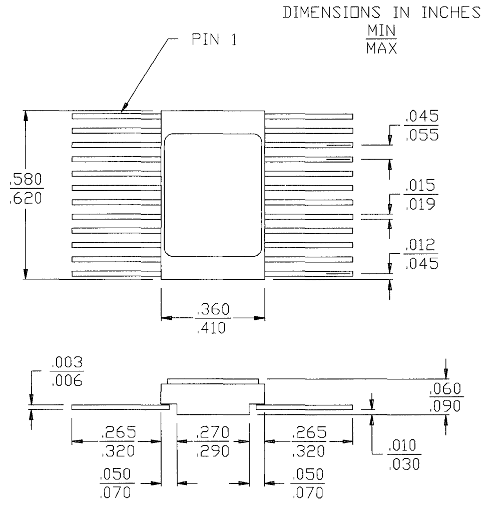


### 18-Lead Rectangular Flatpack F70

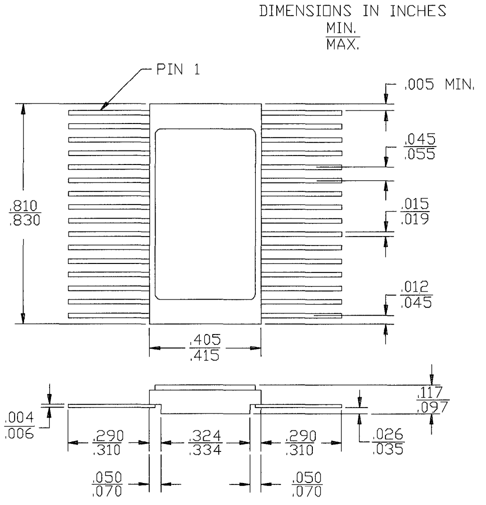


### 24-Lead Rectangular Flatpack F73

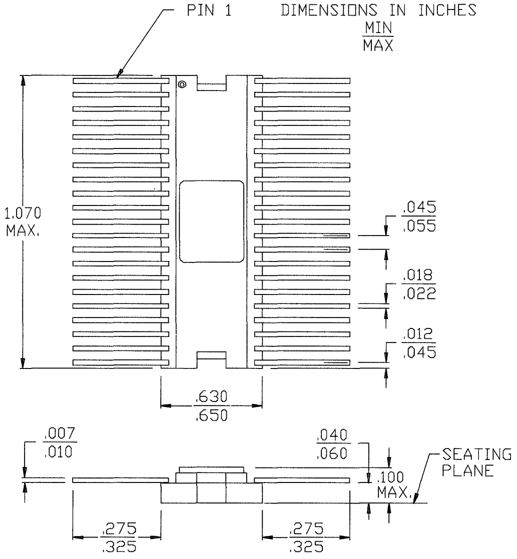
MIL-STD-1835 F-6 Config. B



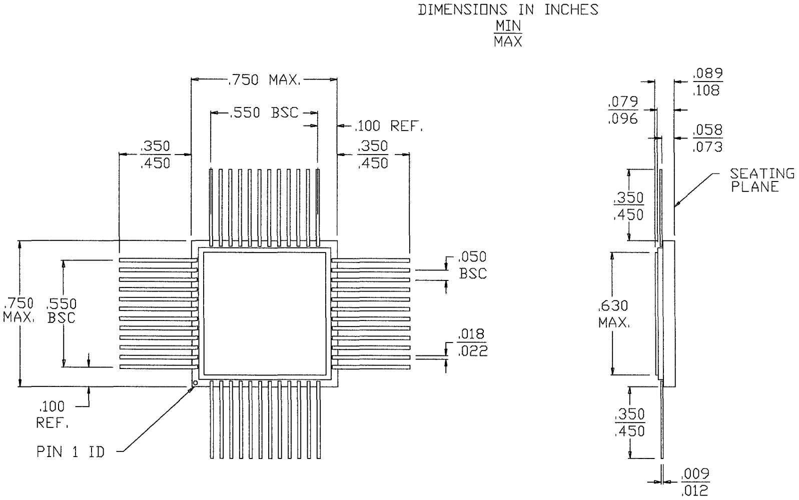
### 32-Lead Rectangular Flatpack F75



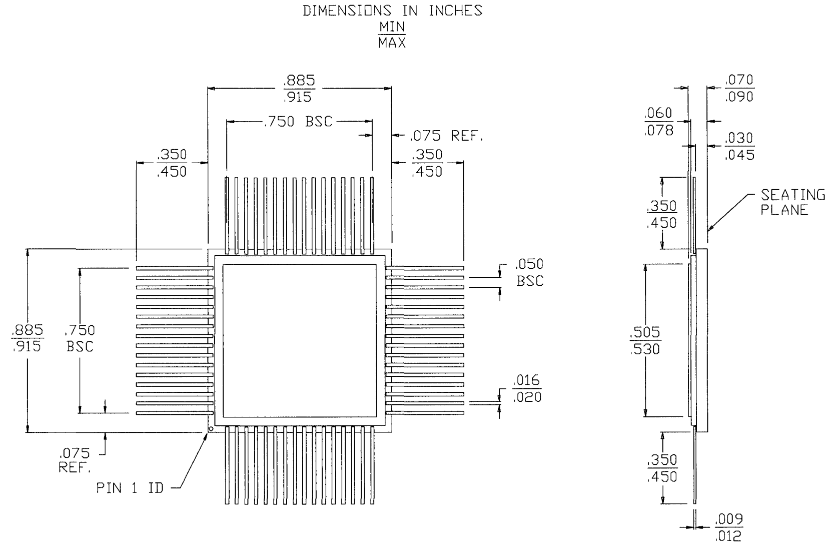
### 42-Lead Rectangular Flatpack F76



### 48-Lead Quad Flatpack F78

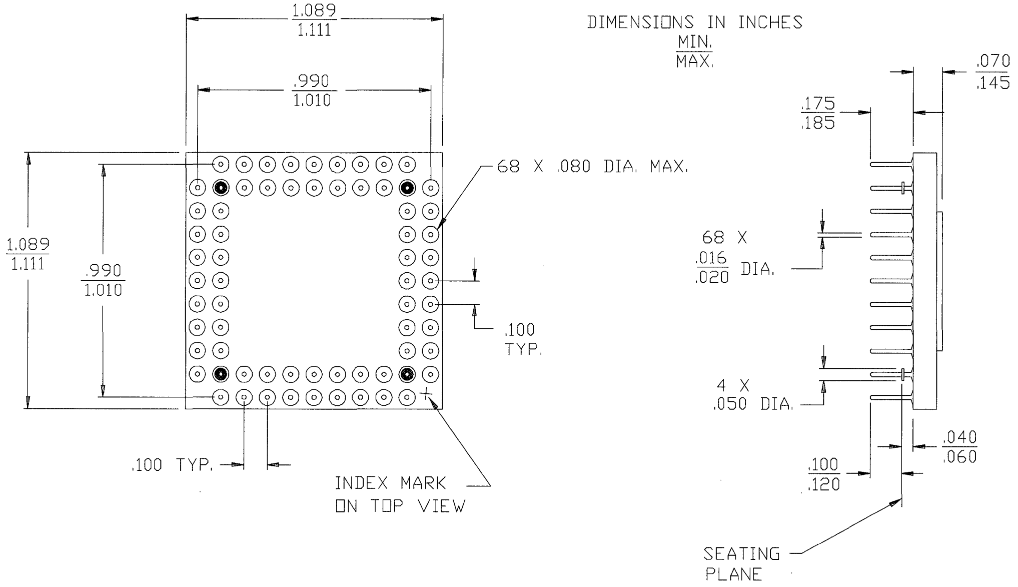


## 64-Lead Quad Flatpack F90

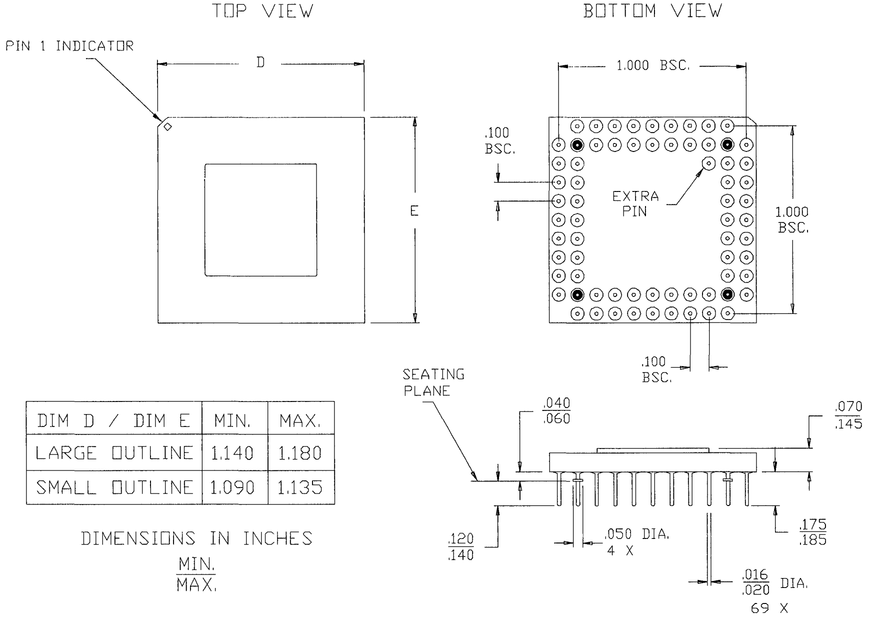


## Ceramic Pin Grid Arrays

### 68-Pin Grid Array (Cavity Up) G68

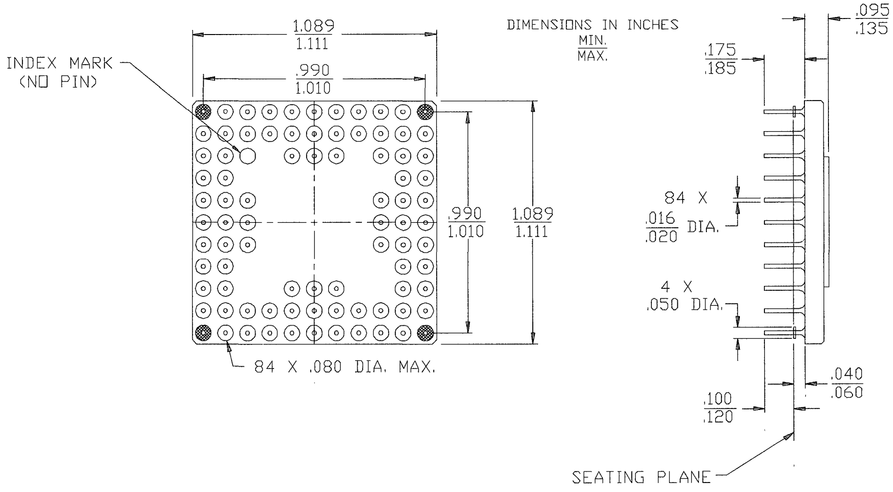


### 69-Pin Grid Array (Cavity Up) G69

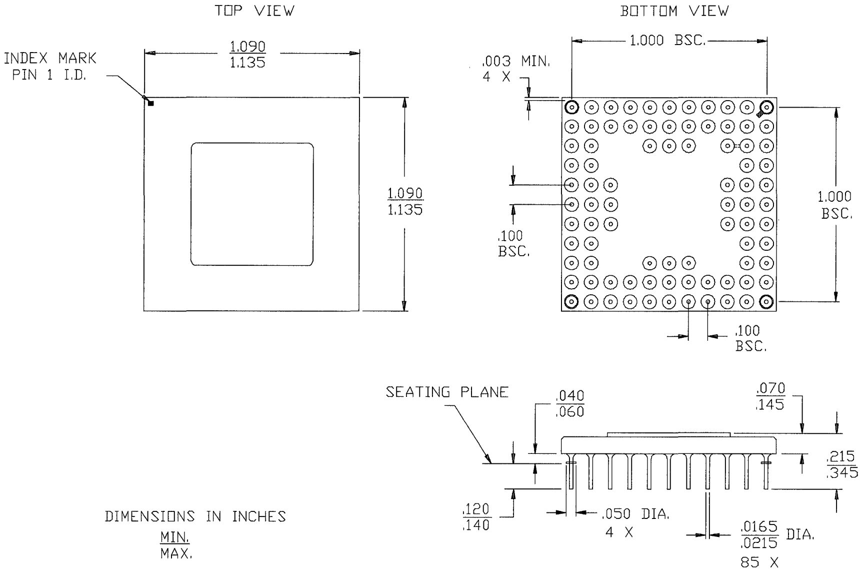


| DIM D / DIM E | MIN. | MAX. |
|---|---|---|
| LARGE OUTLINE | 1.140 | 1.180 |
| SMALL OUTLINE | 1.090 | 1.135 |

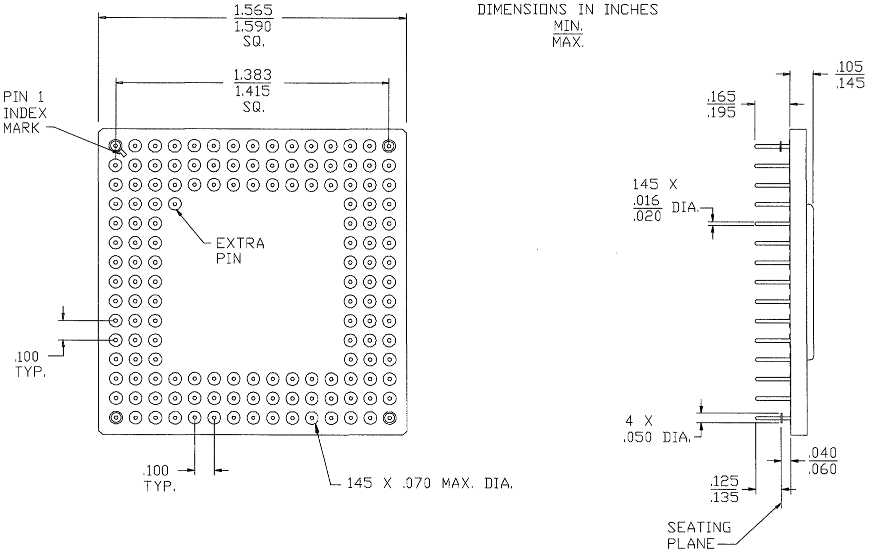
### 84-Pin Grid Array (Cavity Up) G84



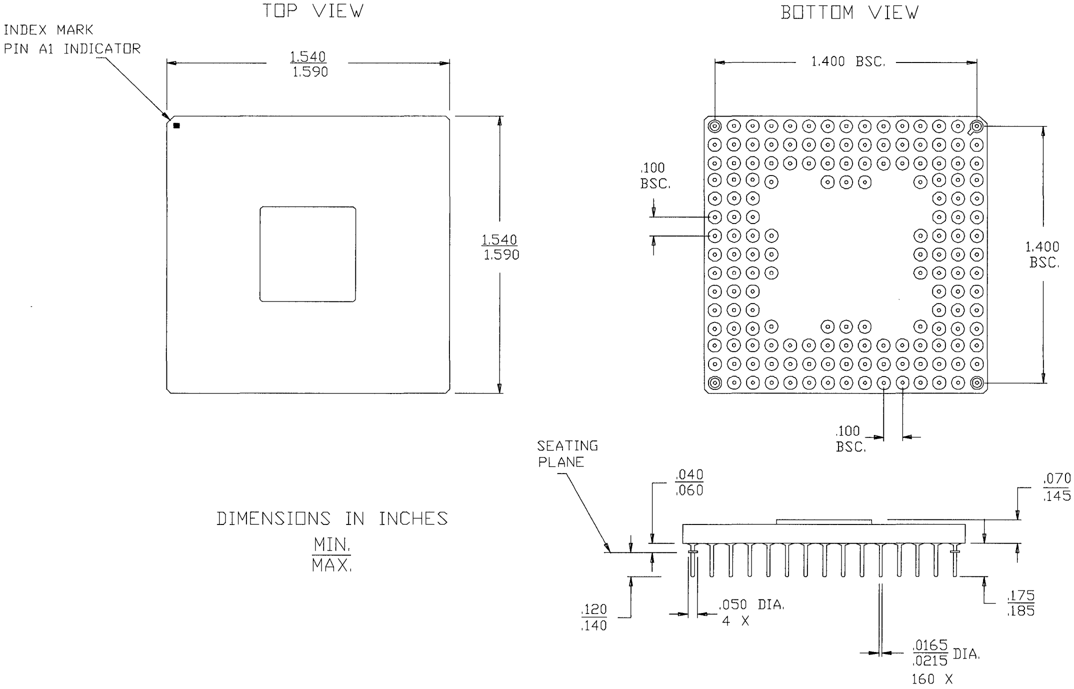
## 85-Pin Grid Array G85



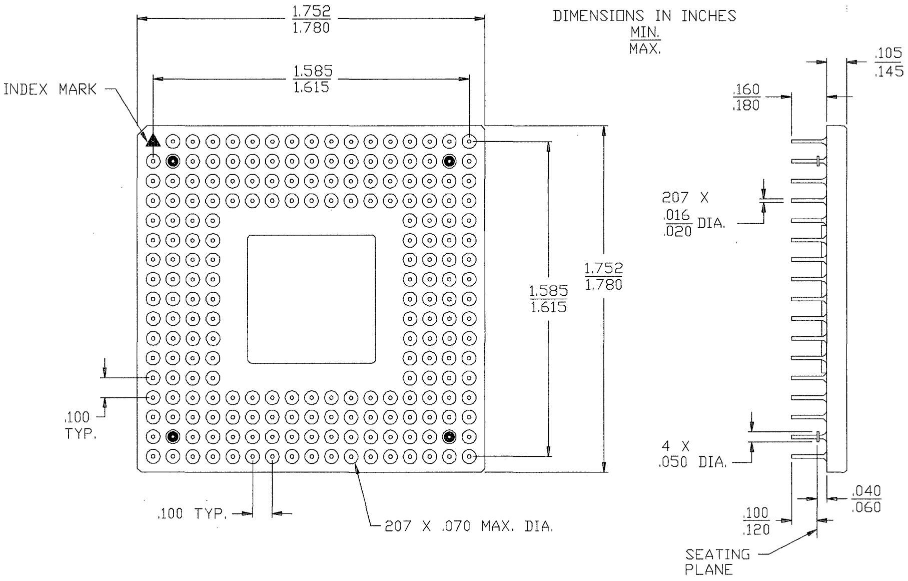
## 145-Pin Grid Array (Cavity Up) G145



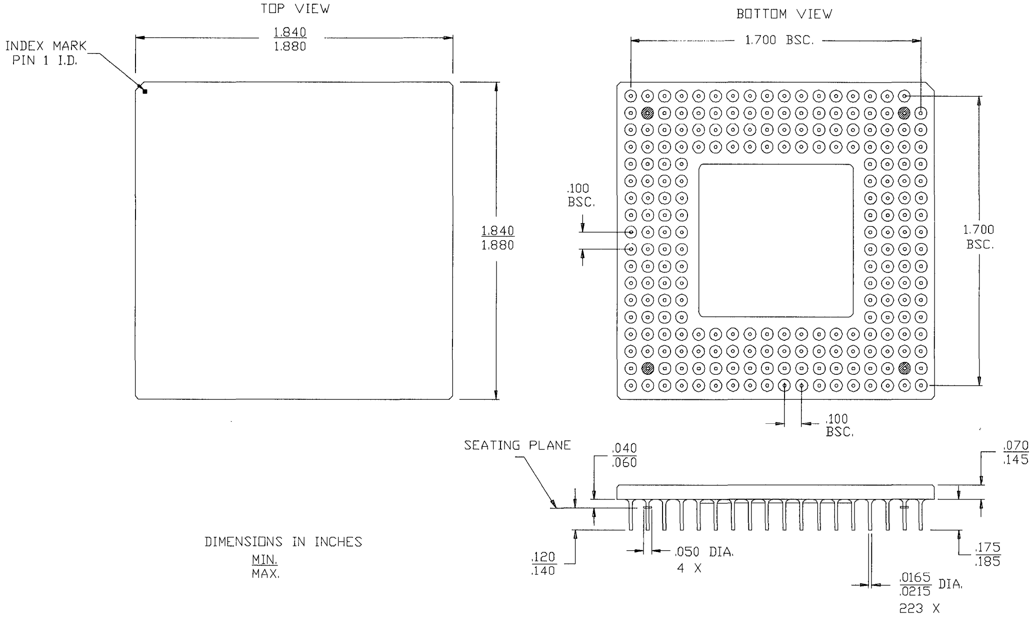
## 160-Pin PGA G160



### 207-Pin Grid Array (Cavity Down) G207

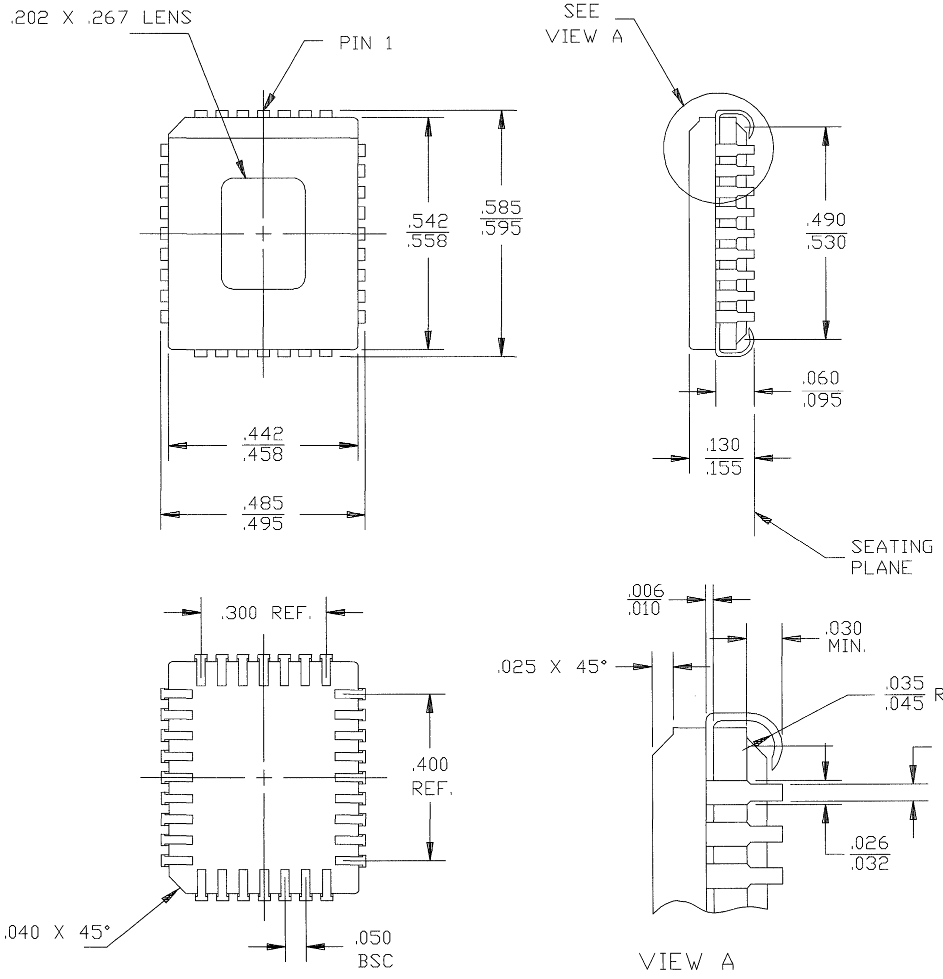


## 223-Pin Grid Array (Cavity Down) G223

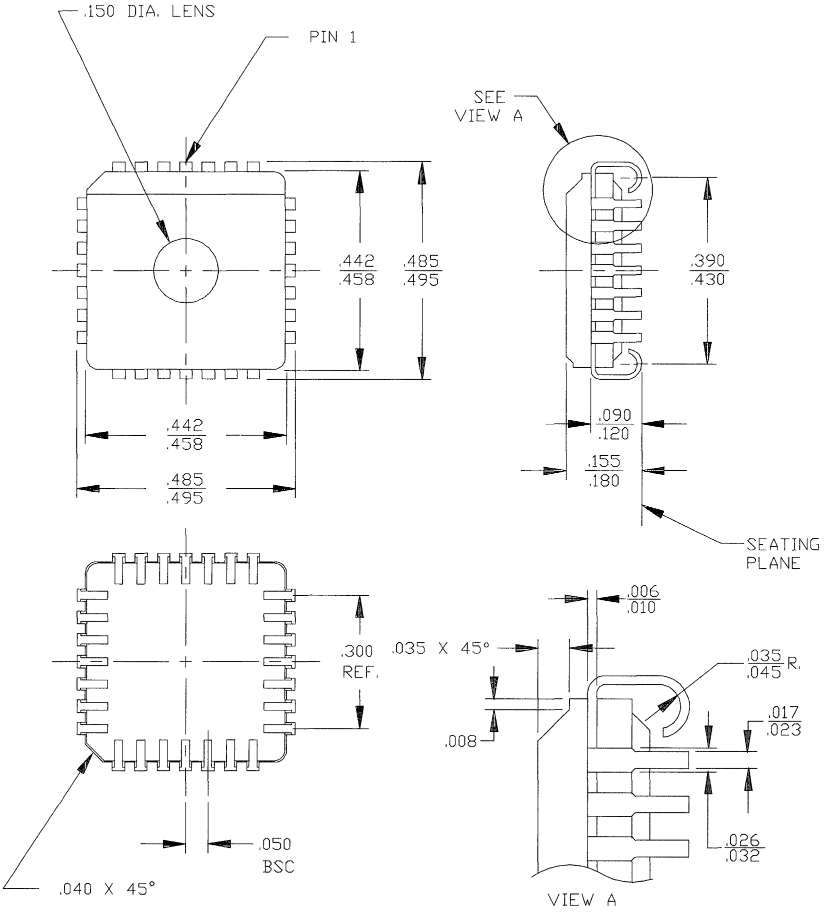


## Ceramic Windowed J-Leaded Chip Carriers

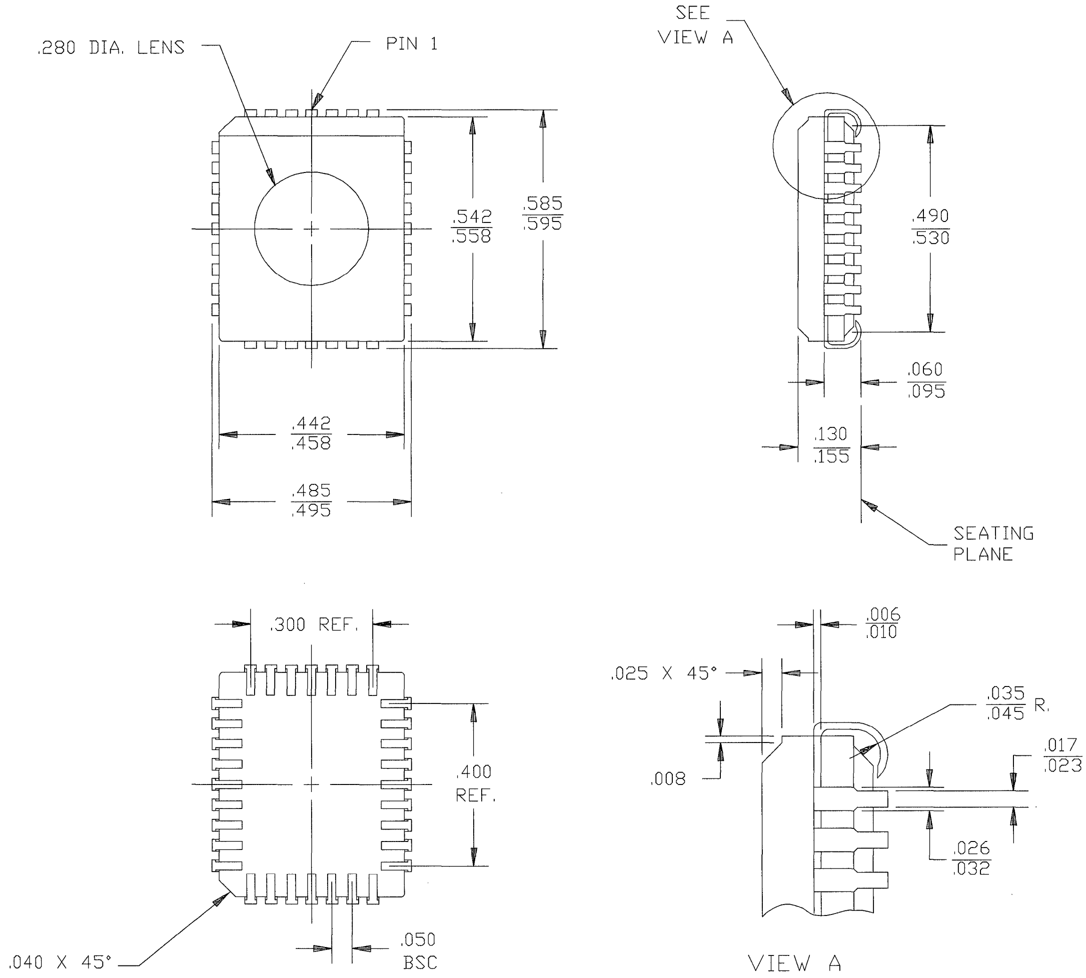
### 32-Pin Windowed Leaded Chip Carrier H32



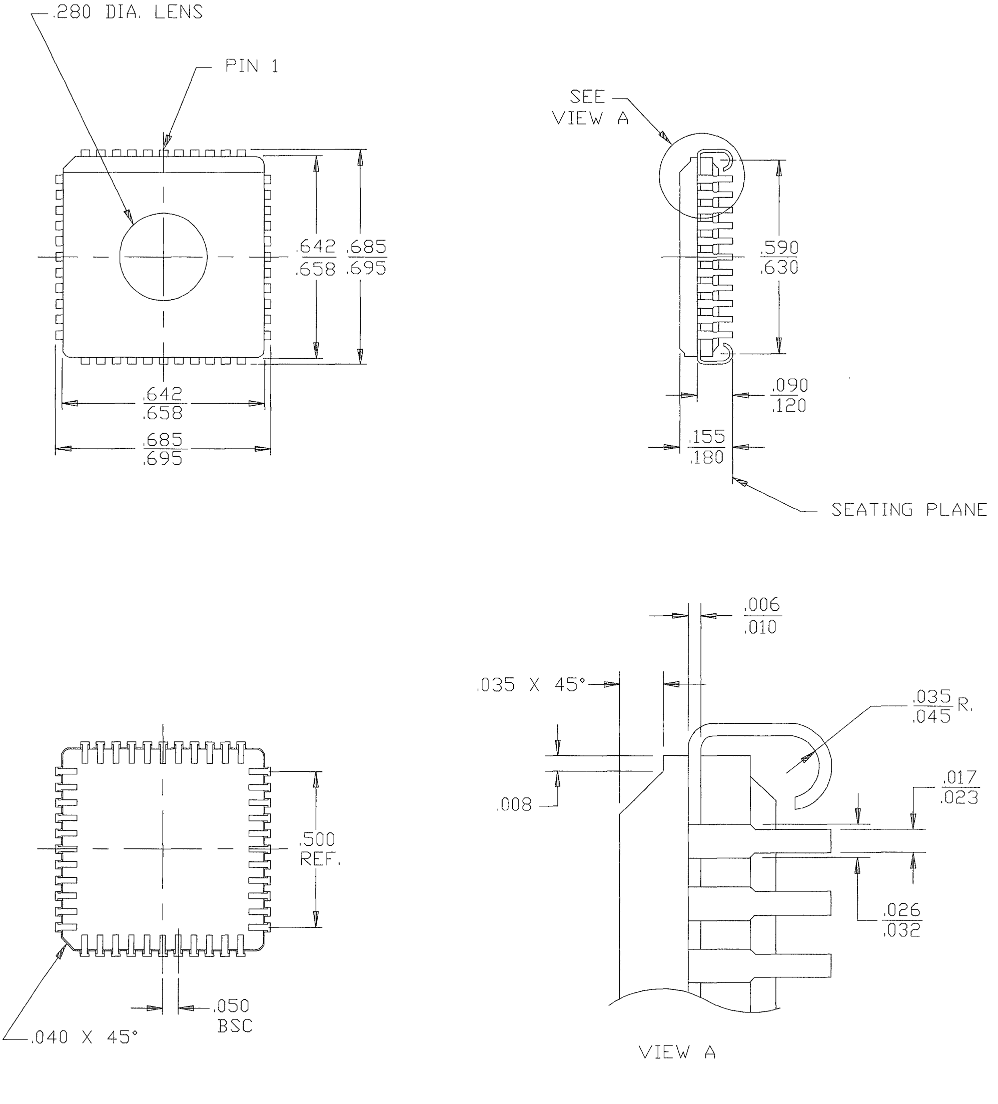
### 28-Pin Windowed Leaded Chip Carrier H64



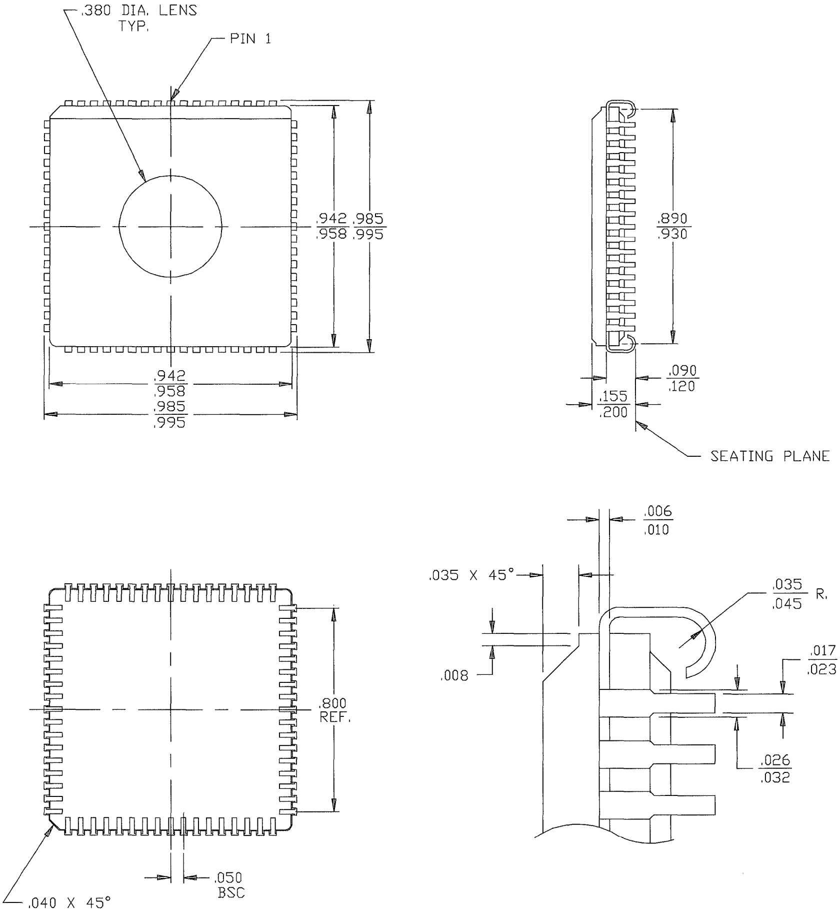
## 32-Pin Windowed Leaded Chip Carrier H65



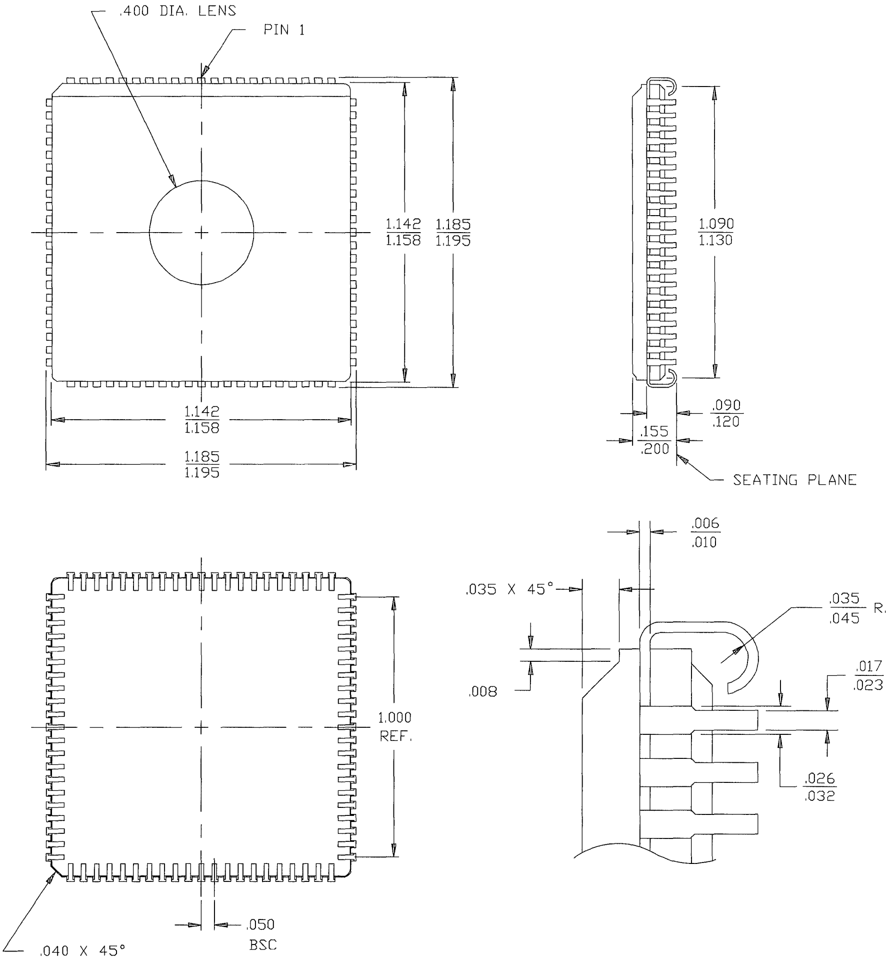
### 44-Pin Windowed Leaded Chip Carrier H67



### 68-Pin Windowed Leaded Chip Carrier H81

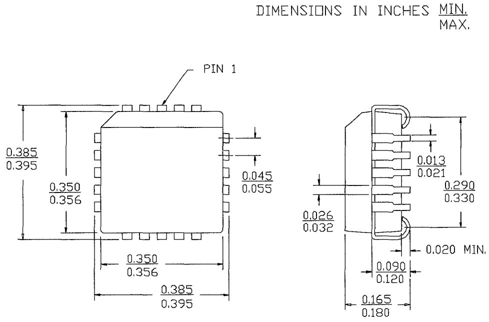


### 84-Lead Windowed Leaded Chip Carrier H84

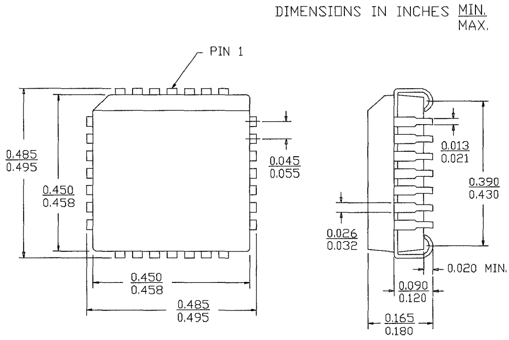


## Plastic Leaded Chip Carriers

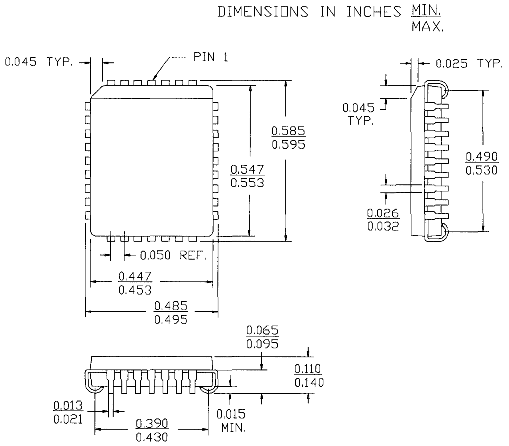
### 20-Lead Plastic Leaded Chip Carrier J61



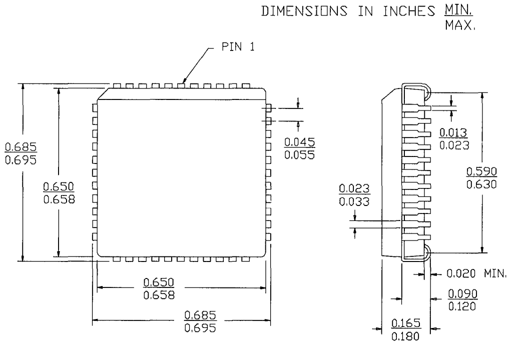
### 28-Lead Plastic Leaded Chip Carrier J64



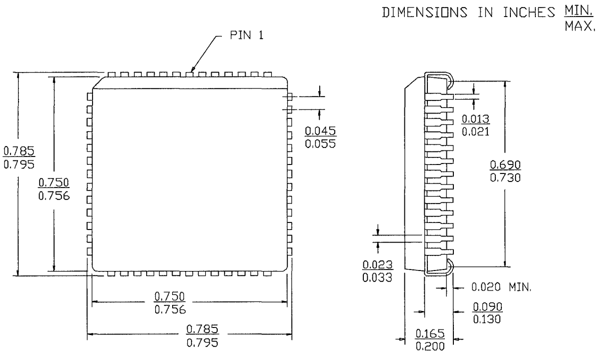
## 32-Lead Plastic Leaded Chip Carrier J65



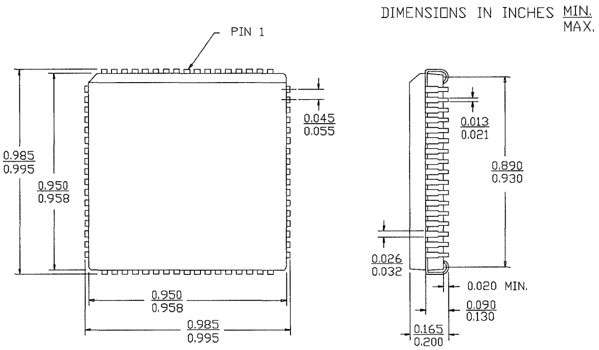
## 44-Lead Plastic Leaded Chip Carrier J67



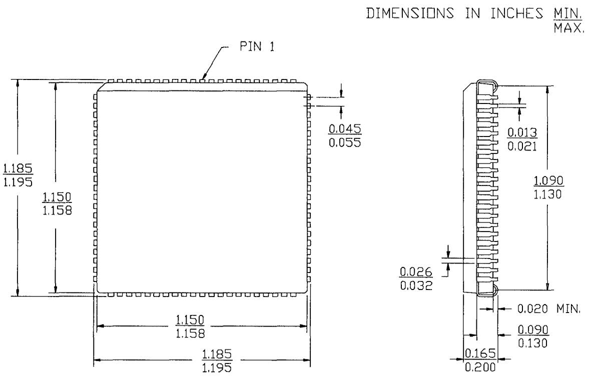
### 52-Lead Plastic Leaded Chip Carrier J69



### 68-Lead Plastic Leaded Chip Carrier J81

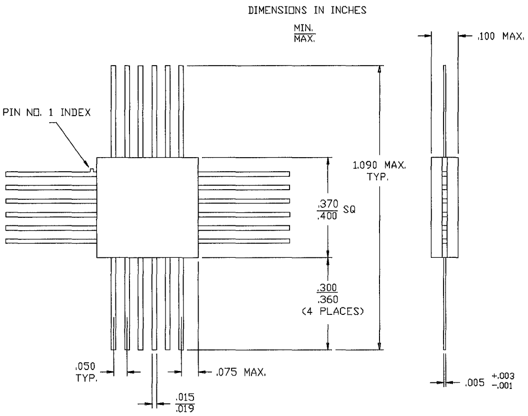


### 84-Lead Plastic Leaded Chip Carrier J83



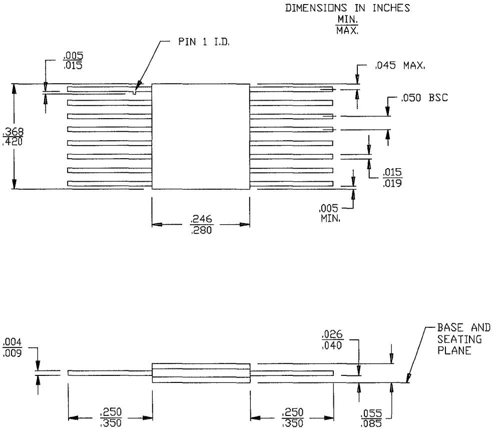
## Cerpacks

### 24-Lead Square Cerpack K63

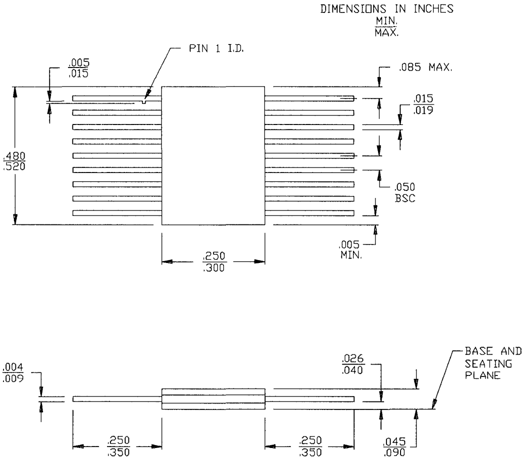


### 16-Lead Rectangular Cerpack K69

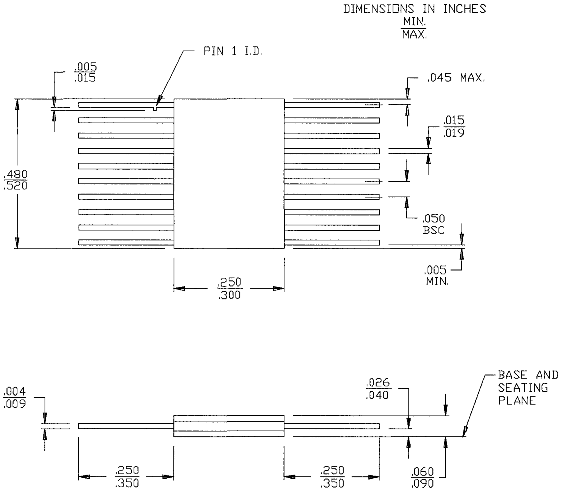
MIL-STD-1835 F-5 Config. A



### 18-Lead Rectangular Cerpack K70
MIL-STD-1835 F-10 Config. A

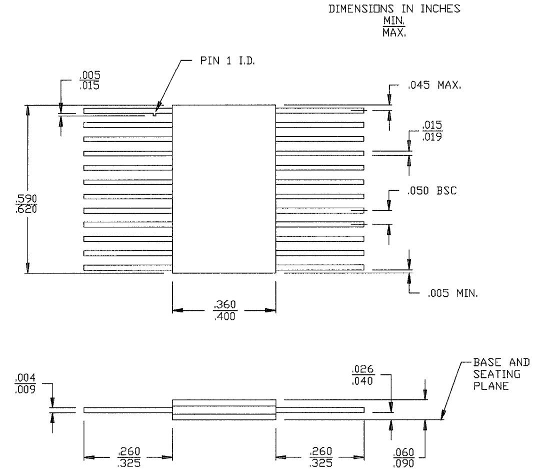


### 20-Lead Rectangular Cerpack K71
MIL-STD-1835 F-9 Config. A



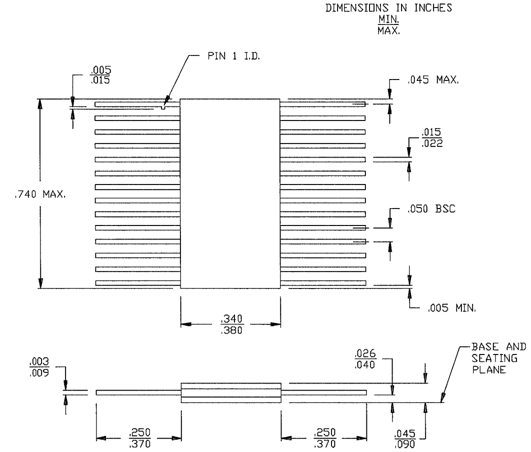
### 24-Lead Rectangular Cerpack K73

MIL-STD-1835 F-6 Config. A

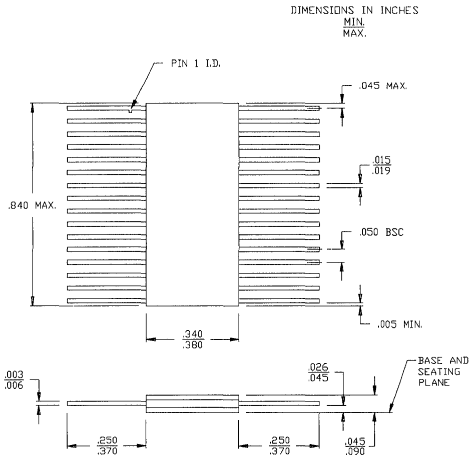


### 28-Lead Rectangular Cerpack K74

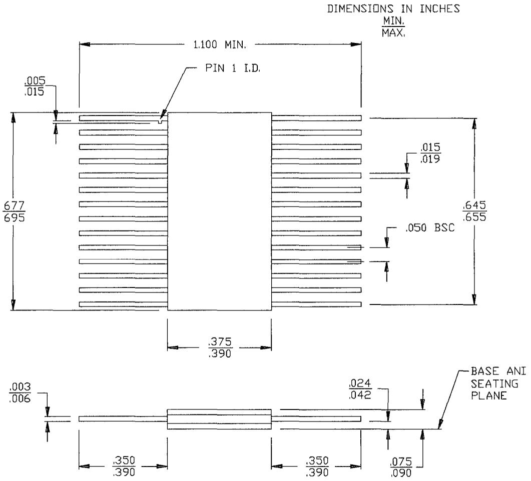
MIL-STD-1835 F-11 Config. A



## 32-Lead Rectangular Cerpack K75

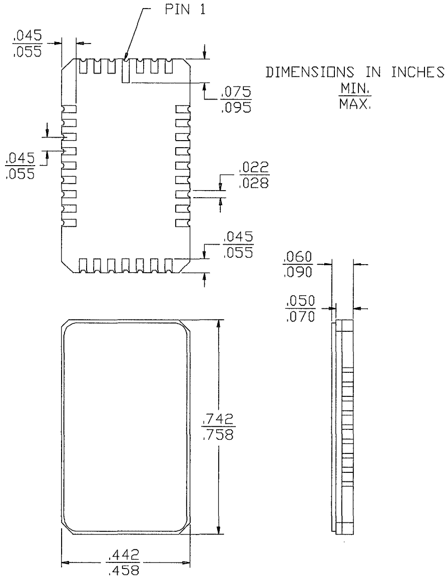


## 28-Lead Rectangular Cerpack K80



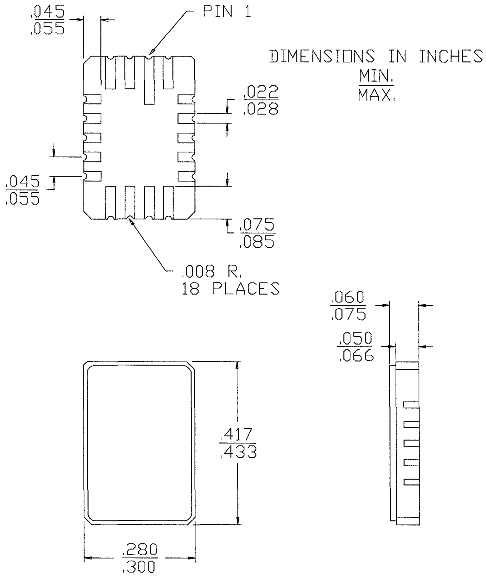
## Ceramic Leadless Chip Carriers

### 32-Lead Leadless Chip Carrier L45



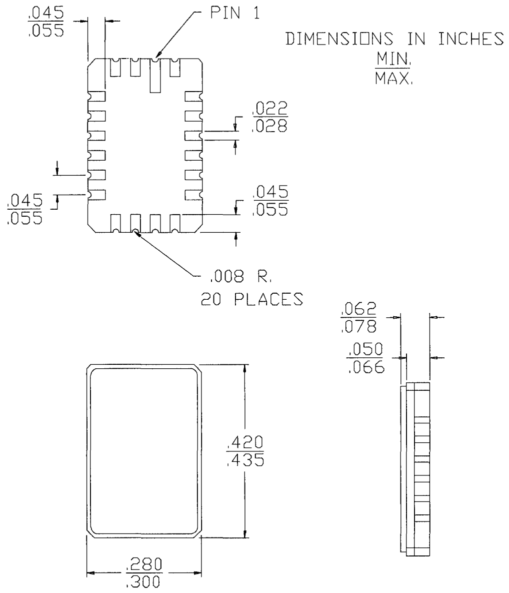
### 18-Pin Rectangular Leadless Chip Carrier L50

MIL-STD-1835 C-10 A

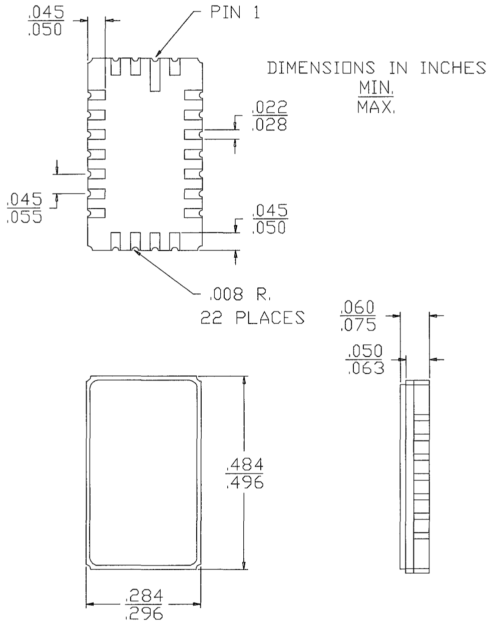


### 20-Pin Rectangular Leadless Chip Carrier L51

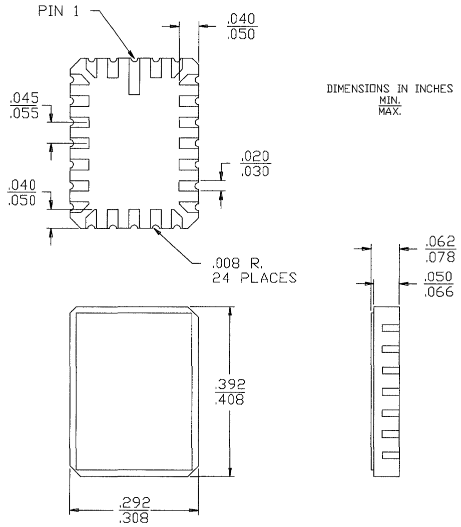
MIL-STD-1835 C-13



### 22-Pin Rectangular Leadless Chip Carrier L52

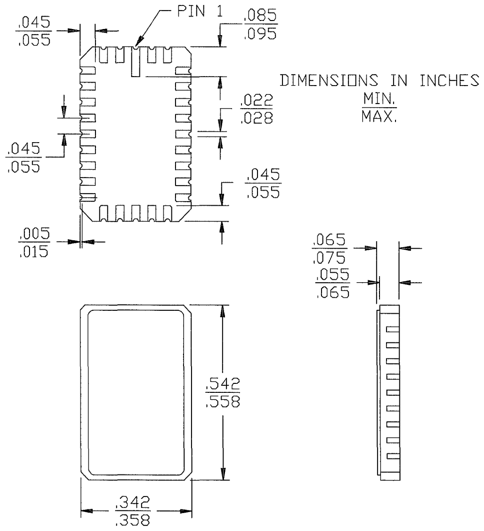


### 24-Pin Rectangular Leadless Chip Carrier L53

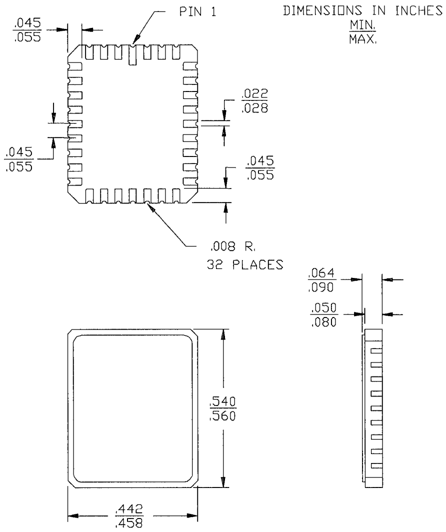


### 28-Pin Rectangular Leadless Chip Carrier L54

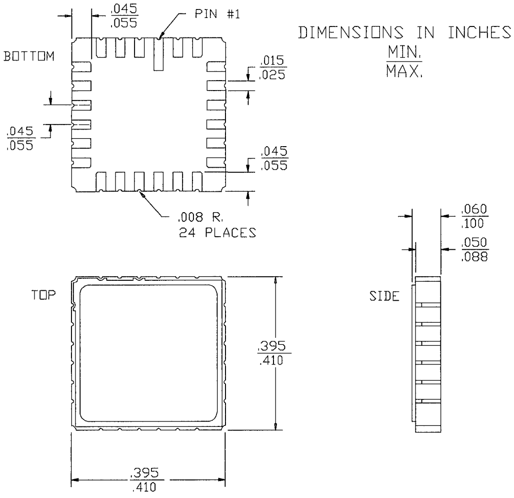
MIL-STD-1835C-11A



### 32-Pin Rectangular Leadless Chip Carrier L55
MIL-STD-1835 C-12

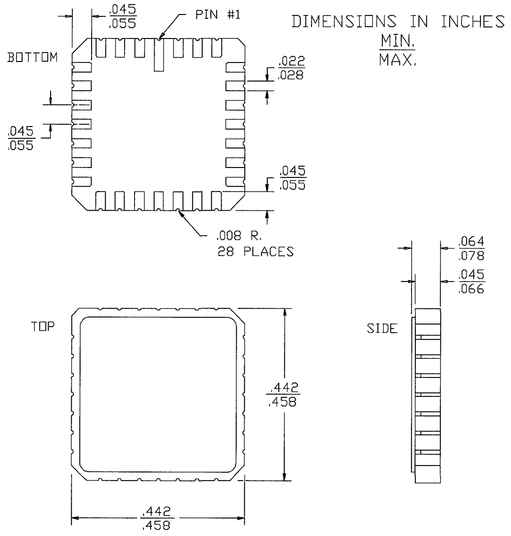


### 24-Square Leadless Chip Carrier L63



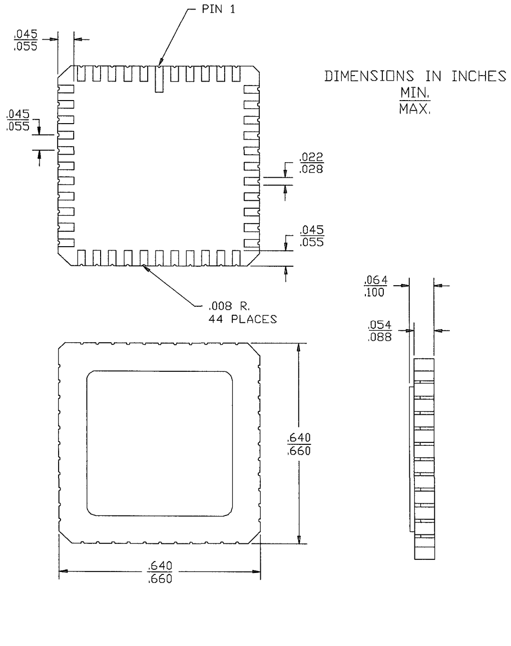
### 28-Square Leadless Chip Carrier L64

MIL-STD-1835 C-4

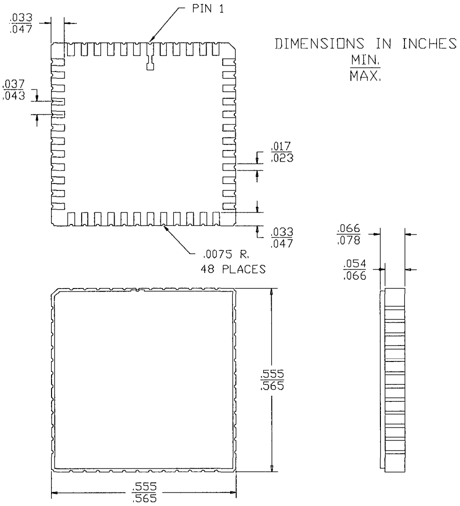


### 44-Square Leadless Chip Carrier L67

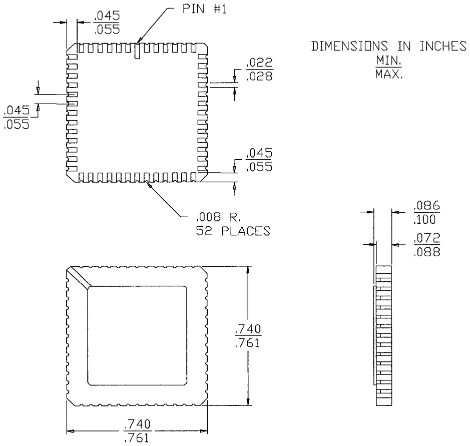
MIL-STD-1835 C-5



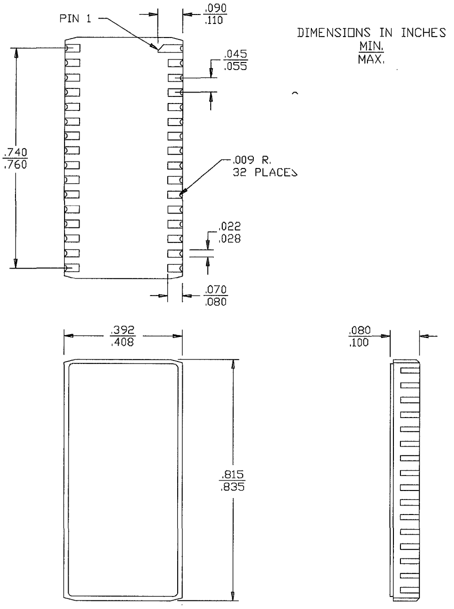
### 48—Square Leadless Chip Carrier L68



### 52-Square Leadless Chip Carrier L69

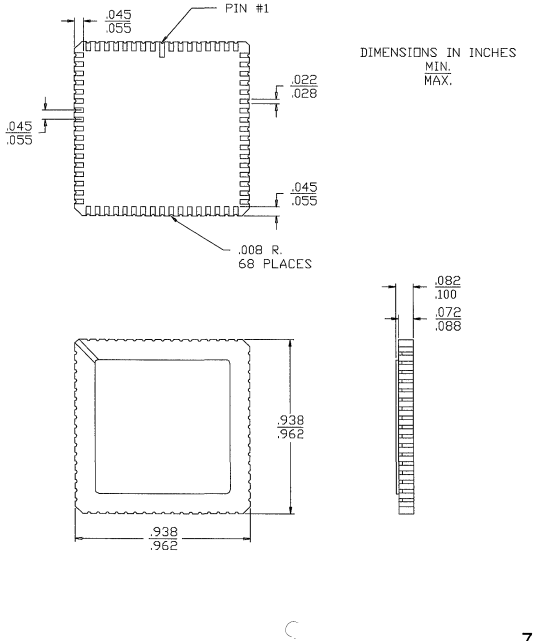


### 32-Pin Leadless Chip Carrier L75



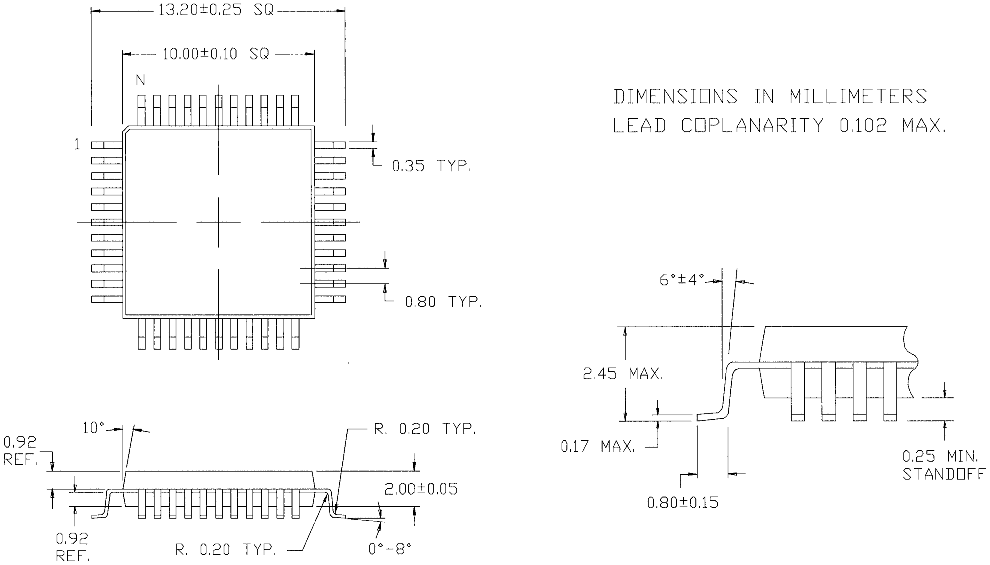
### 68-Square Leadless Chip Carrier L81

MIL-STD-1835 C-7



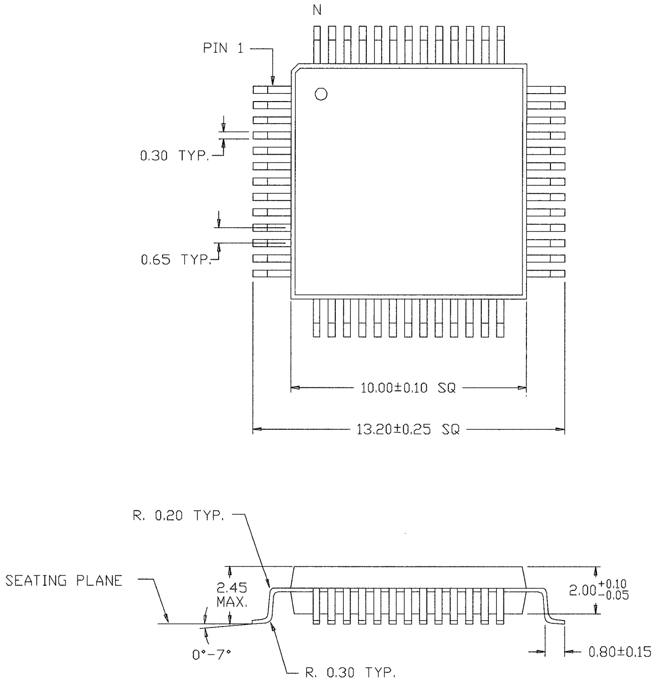
## Plastic Quad Flatpacks

### 44-Lead Plastic Quad Flatpack N44



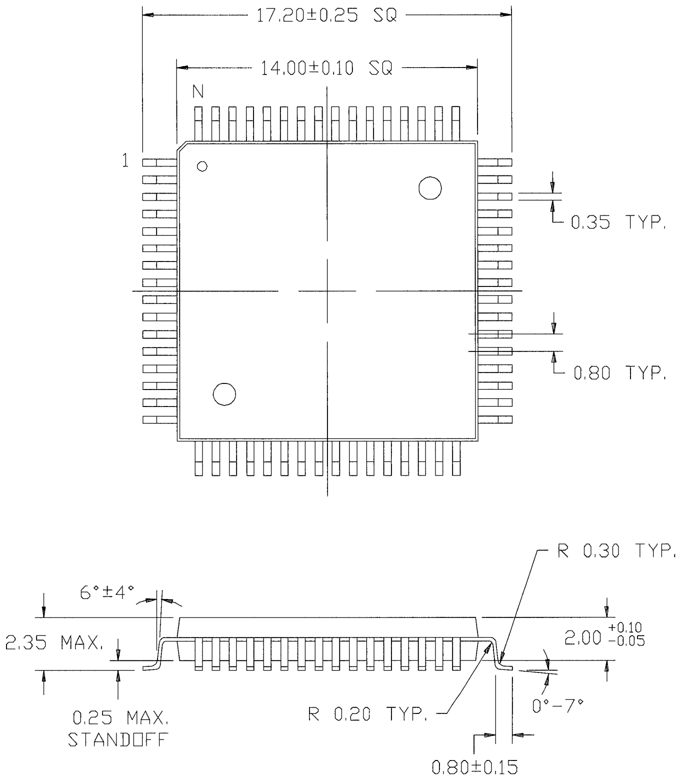
DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.102 MAX.

### 52-Lead Plastic Quad Flatpack N52



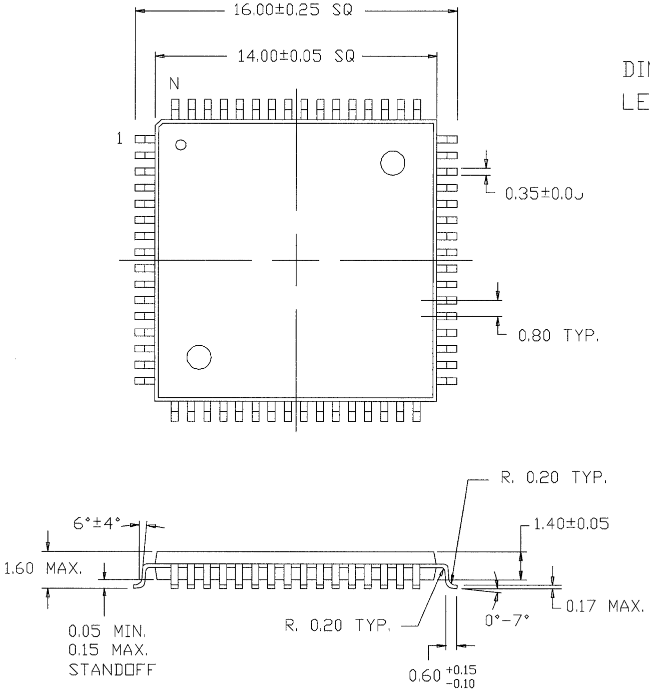
DIMENSIONS ARE IN MILLIMETERS
LEAD COPLANARITY 0.102 MAX.

**64-Lead Plastic Quad Flatpack N64**



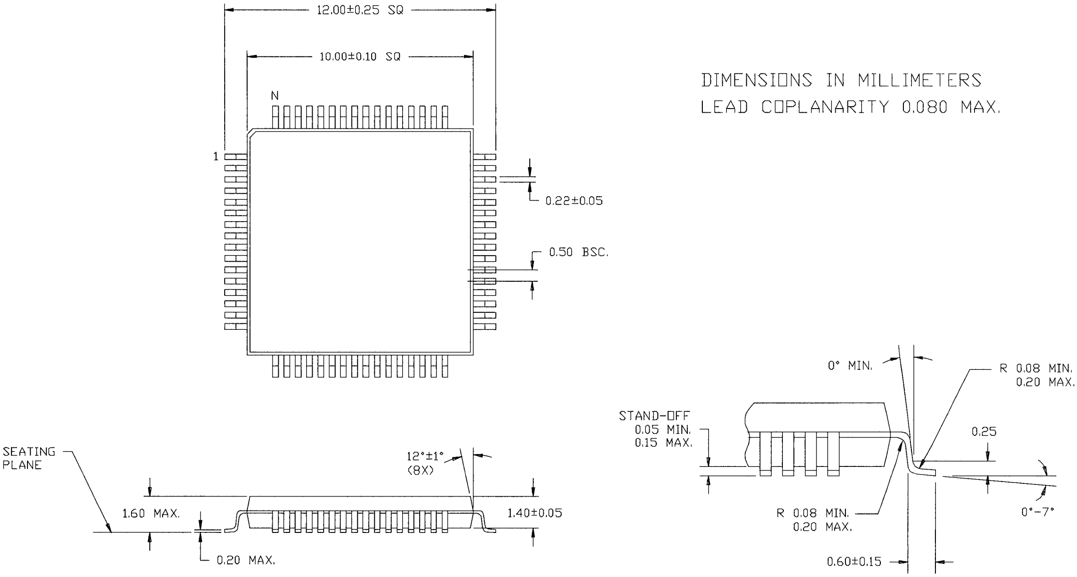
DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.102 MAX

### 64-Lead Plastic Thin Quad Flatpack N65



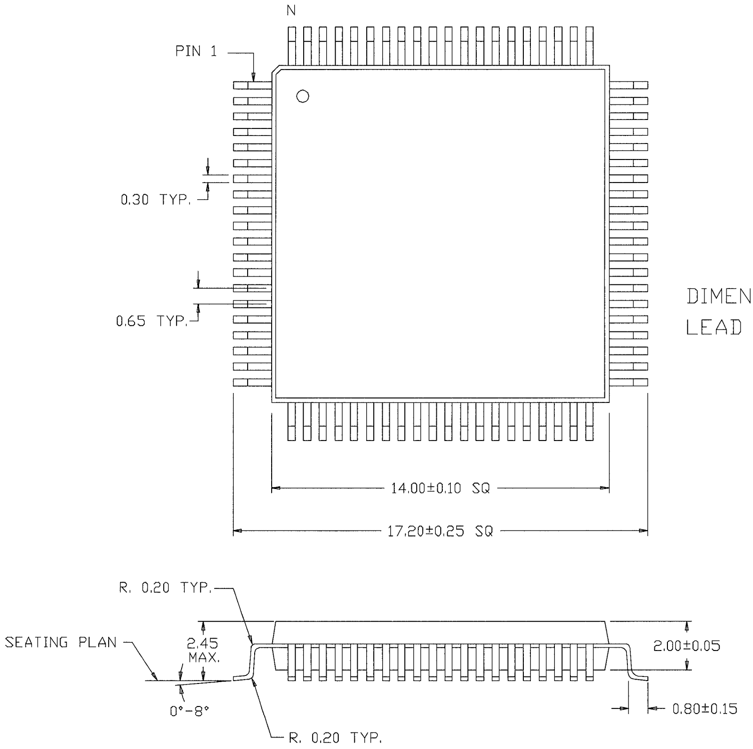
DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.100 MAX.

### 64-Lead Plastic Thin Quad Flatpack N66



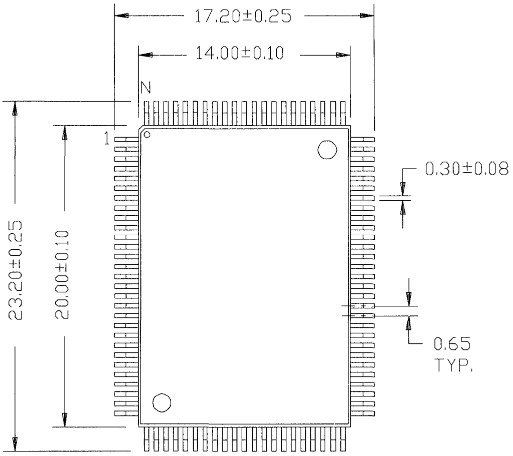
DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.080 MAX.

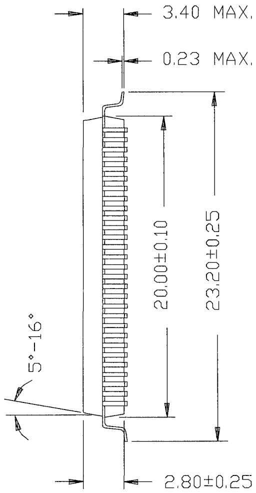
### 80-Lead Plastic Quad Flatpack N80

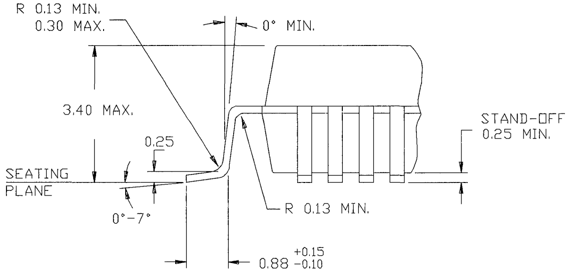


DIMENSIONS ARE IN MILLIMETERS
LEAD COPLANARITY 0.102 MAX.

## 100-Lead Plastic Quad Flatpack N100



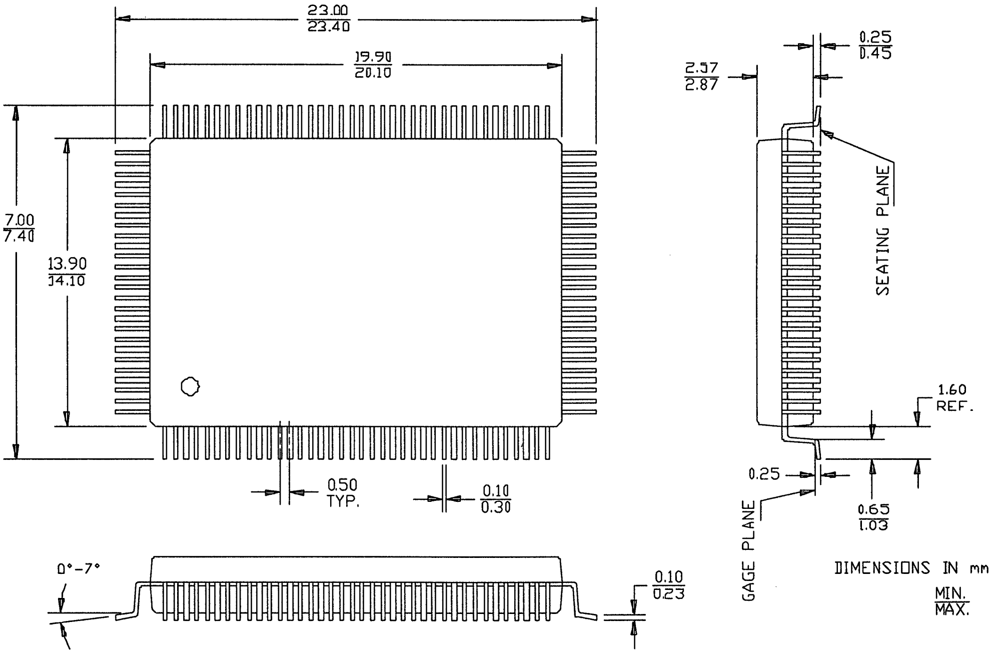




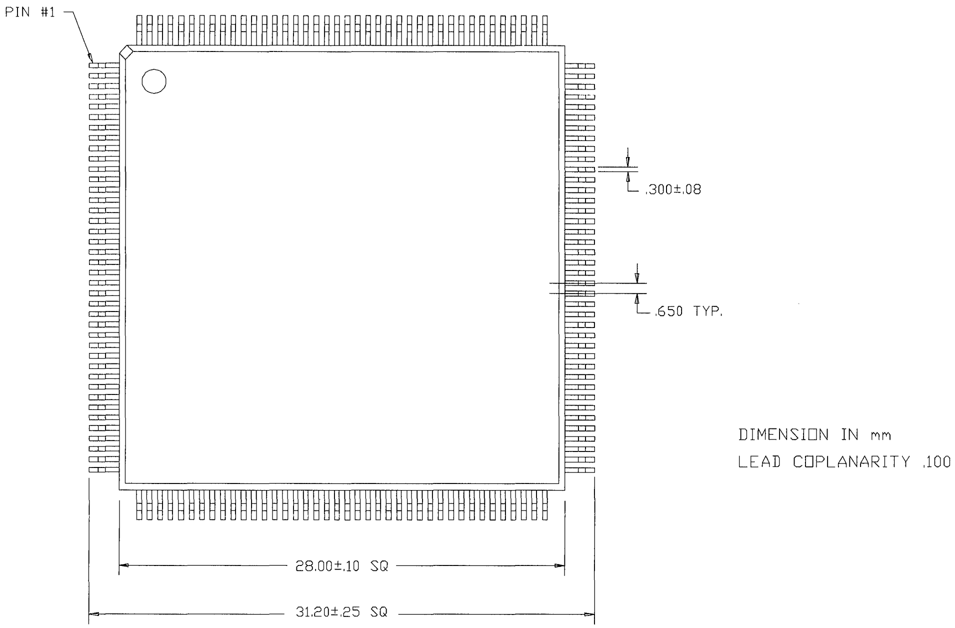
NOTES:

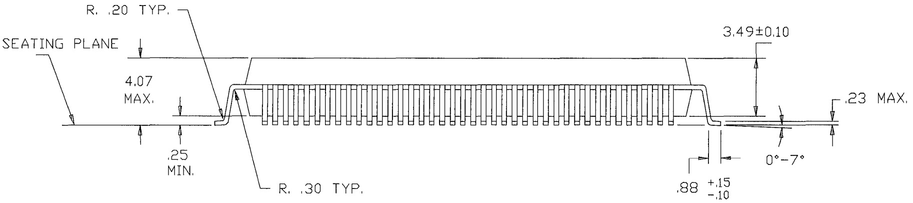
1. DIMENSIONS ARE IN MILLIMETERS.
2. LEAD COPLANARITY 0.100 MAX.
3. PACKAGE WIDTH (14.00±0.10) AND LENGTH (20.00±0.10) DOES NOT INCLUDE MOLD PROTRUSION. MAX. ALLOWABLE PROTRUSION IS 0.25 MM.
4. LEAD WIDTH DOES NOT INCLUDE DAMBAR PROTRUSION. MAX. ALLOWABLE DAMBAR PROTRUSION ABOVE LOWER RADIUS IS 0.08 MM.

## 128-Lead Plastic Quad Flatpack

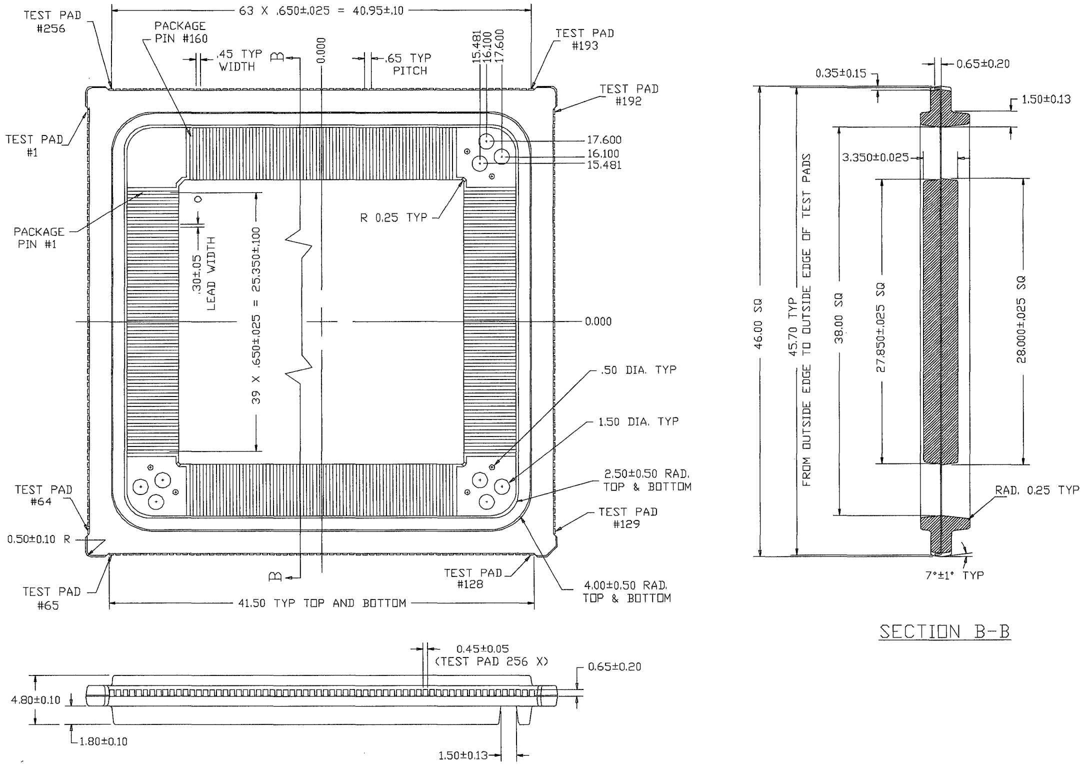


## 160-Lead Plastic Quad Flatpack N160

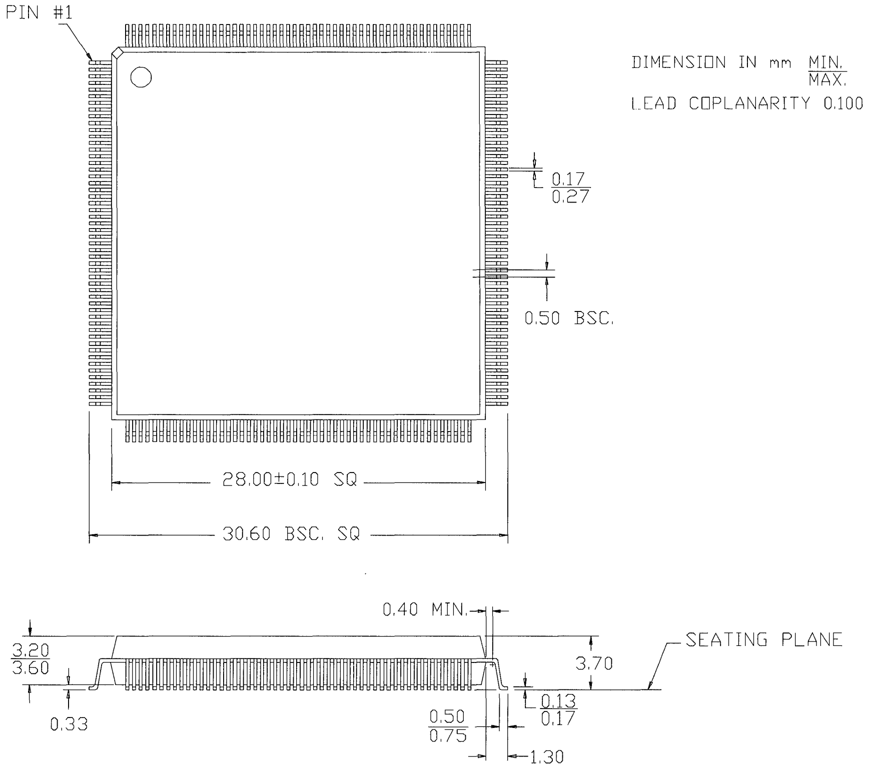




## 160-Lead Plastic Quad Flatpack with Molded Carrier Ring N161

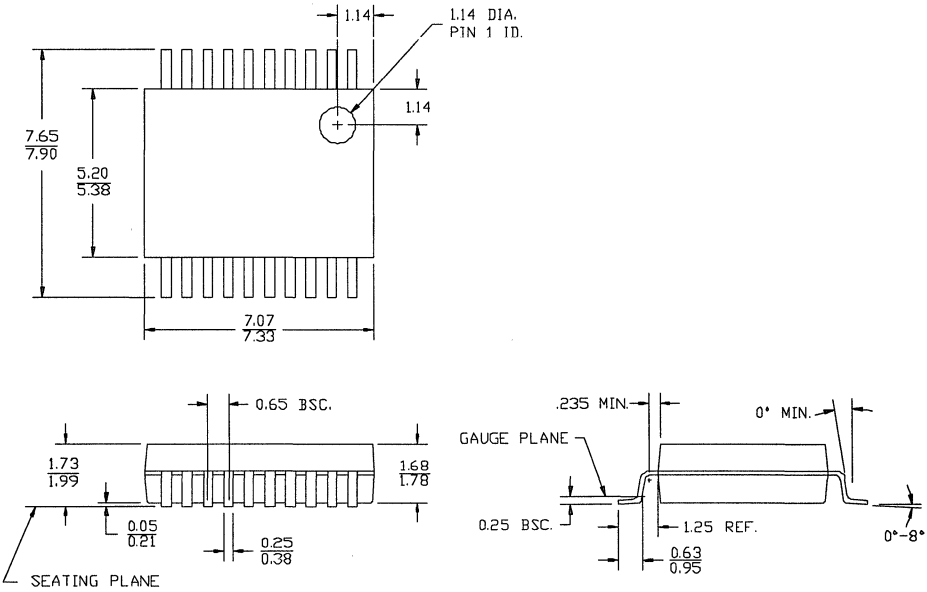


## 208-Lead Plastic Quad Flatpack N208

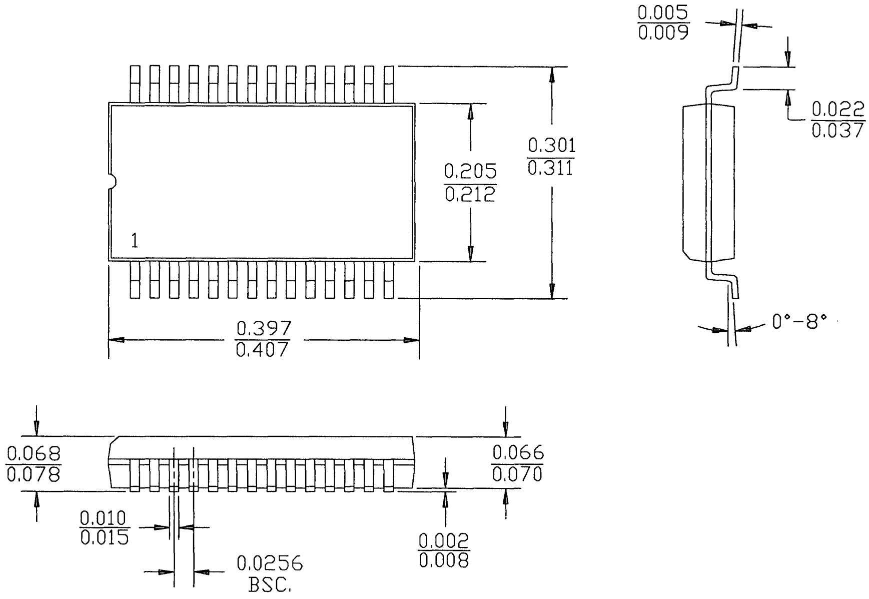


## Shrunk Small Outline Packages

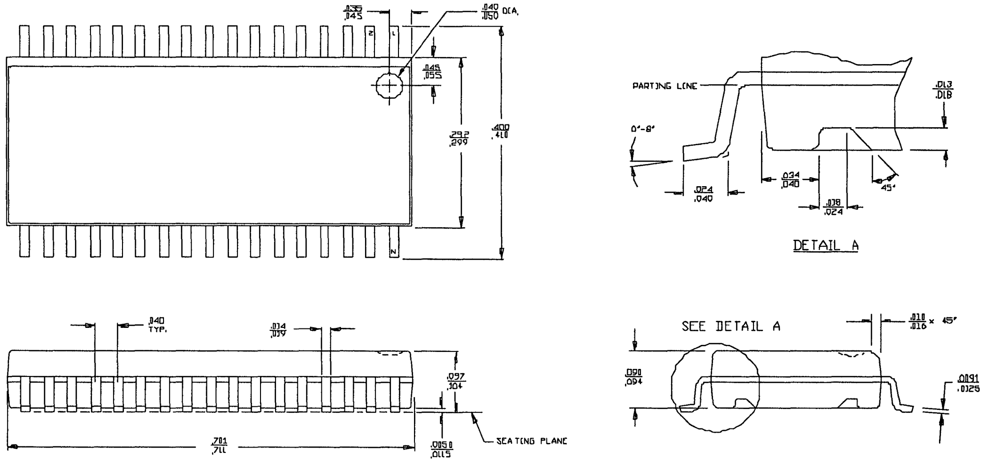
### 20-Pin Shrunk Small Outline Package O20



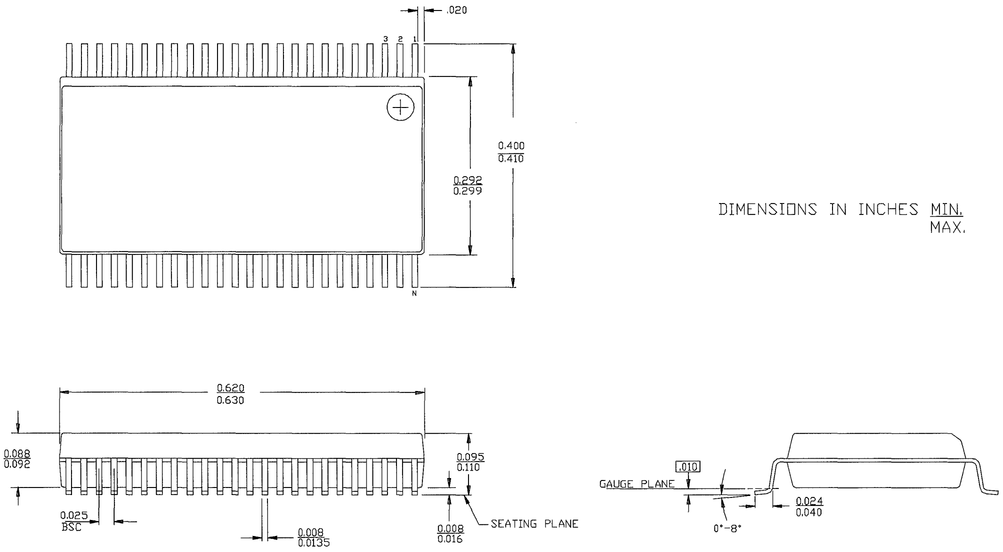
### 28-Lead Shrunk Small Outline Package O28



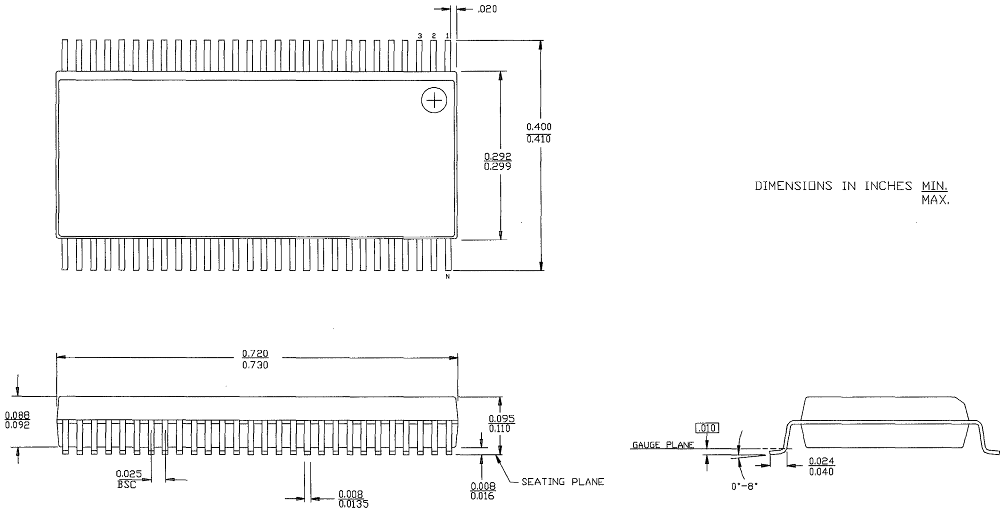
### 34-Pin Shrunk Small Outline Package O34



## 48-Lead Shrunk Small Outline Package O48

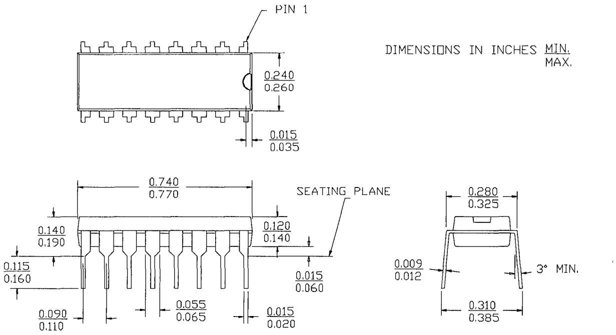


## 56-Lead Shrunk Small Outline Package O56

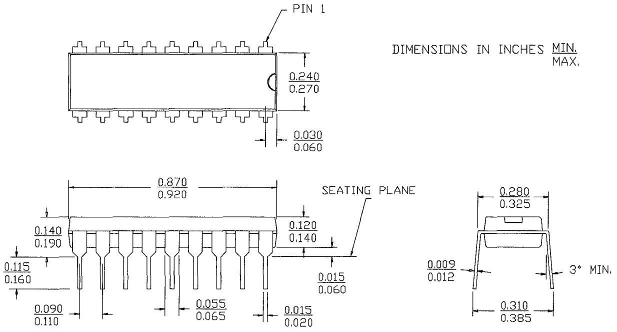


## Plastic Dual-In-Line Packages

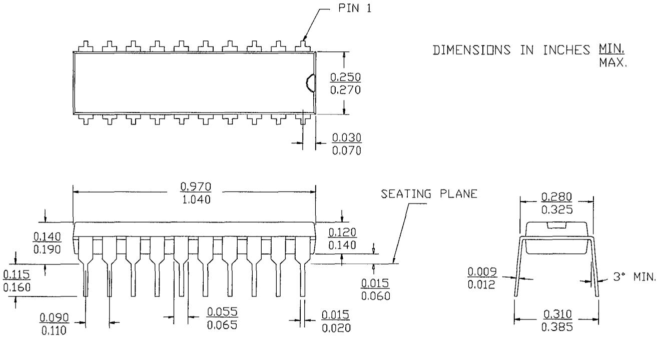
### 16-Lead (300-Mil) Molded DIP P1



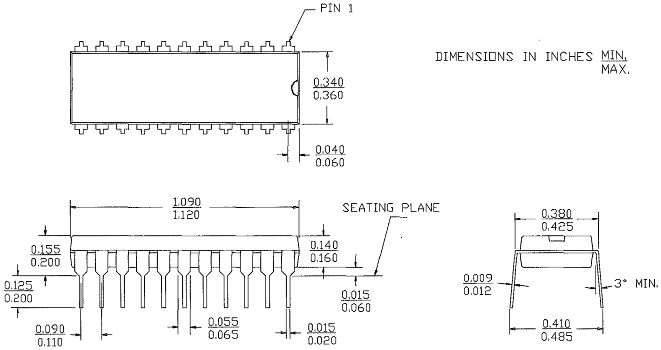
### 18-Lead (300-Mil) Molded DIP P3



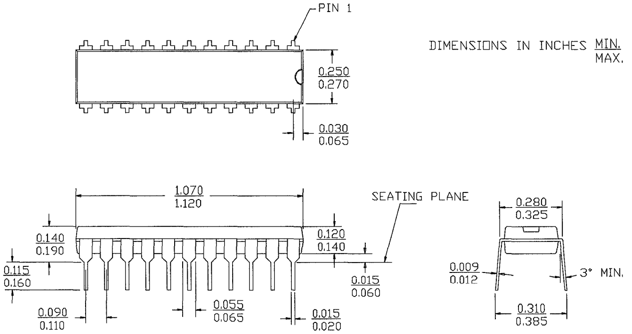
### 20-Lead (300-Mil) Molded DIP P5



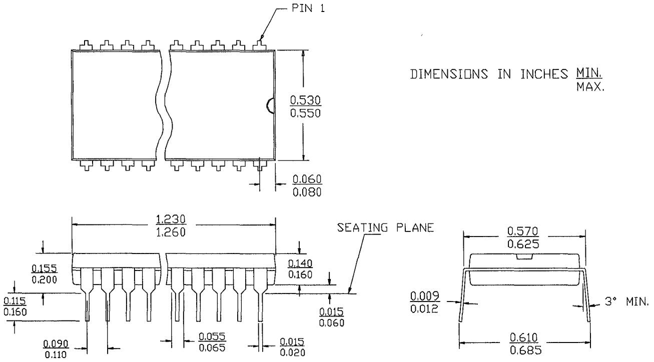
## 22-Lead (400-Mil) Molded DIP P7



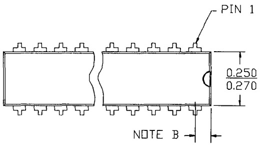
## 22-Lead (300-Mil) Molded DIP P9



## 24-Lead (600-Mil) Molded DIP P11

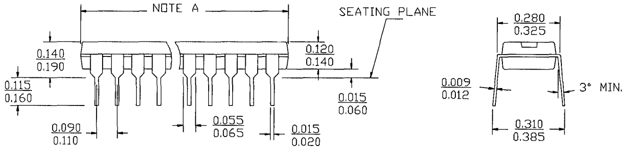


## 24-Lead (300-Mil) Molded DIP P13/P13A

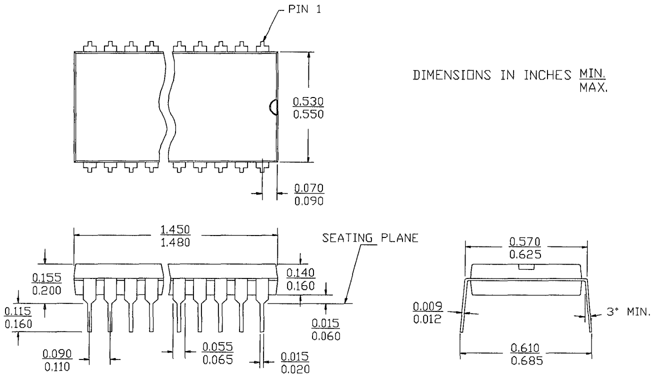


DIMENSIONS IN INCHES MIN./MAX.

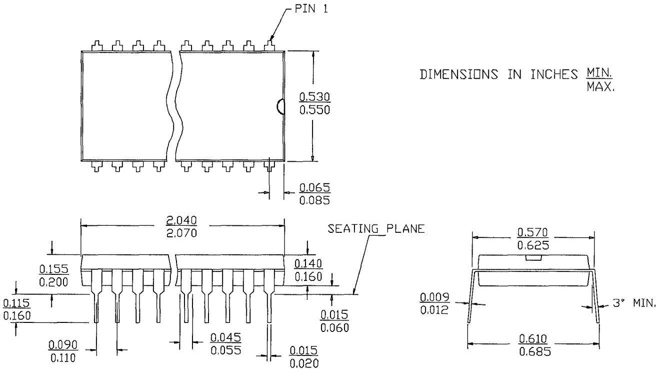
| | P 13 | P 13A |
|---|---|---|
| NOTE A | 1.170<br>1.200 | 1.230<br>1.260 |
| NOTE B | 0.030<br>0.050 | 0.060<br>0.080 |



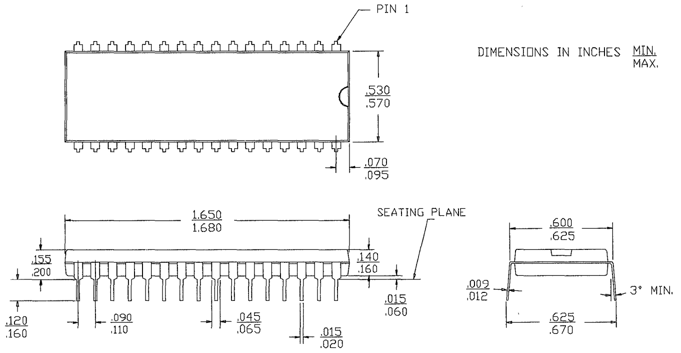
## 28-Lead (600-Mil) Molded DIP P15



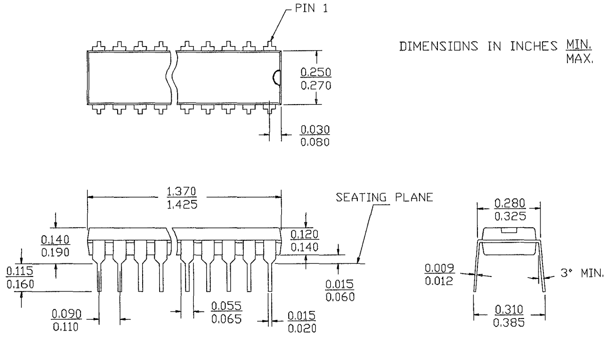
## 40-Lead (600-Mil) Molded DIP P17



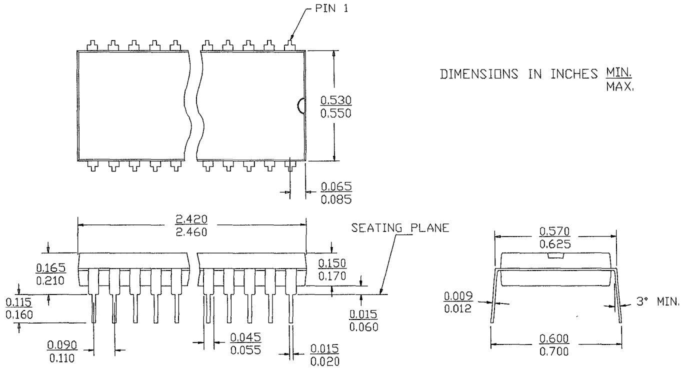
### 32-Lead (600-Mil) Molded DIP P19



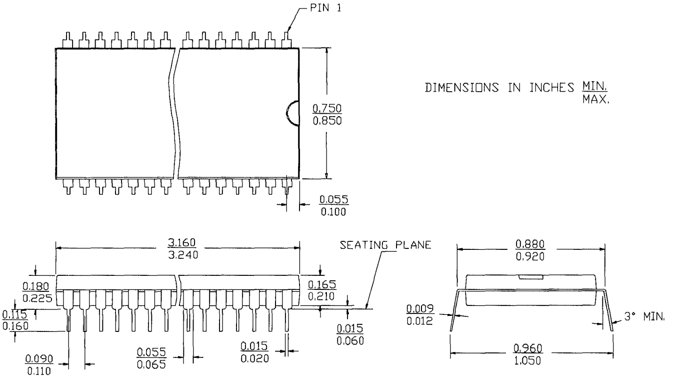
### 28-Lead (300-Mil) Molded DIP P21



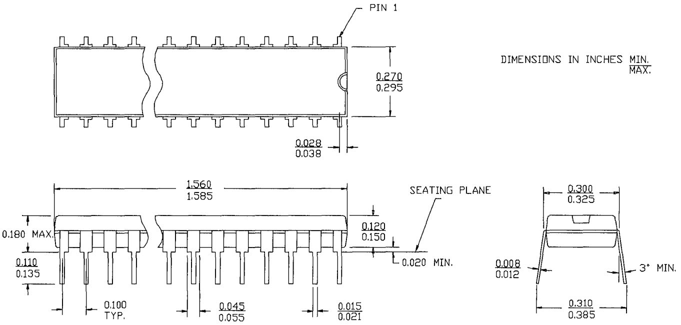
### 48-Lead (600-Mil) Molded DIP P25



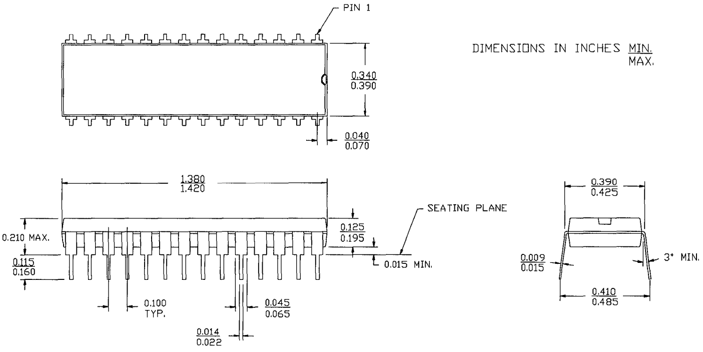
### 64-Lead (900-Mil) Molded DIP P29



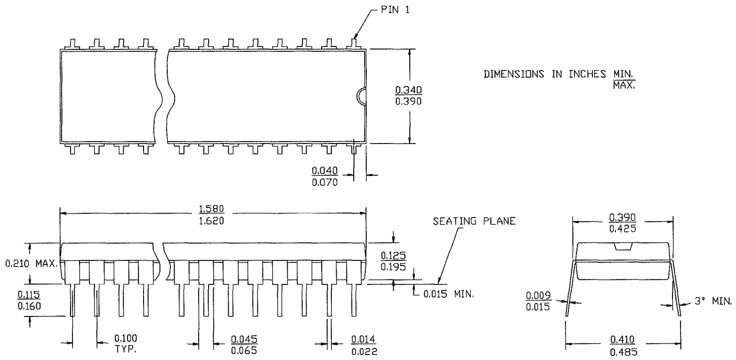
### 32-Lead (300-Mil) Molded DIP P31



### 28-Lead (400-Mil) Molded DIP P41

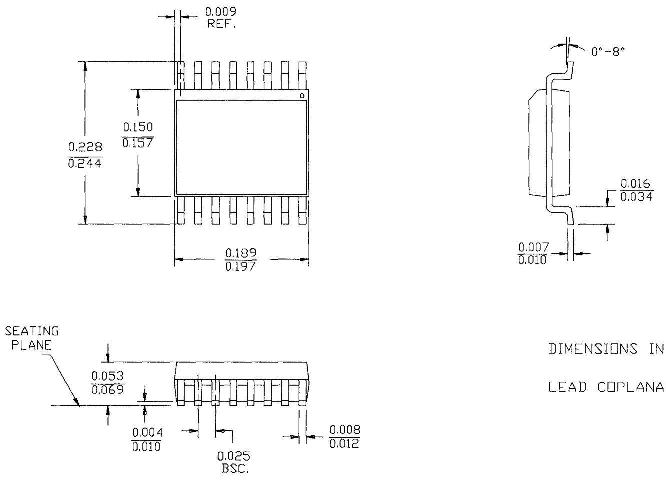


## 32-Lead (400-Mil) Molded DIP P43



## Quarter Size Outline Packages

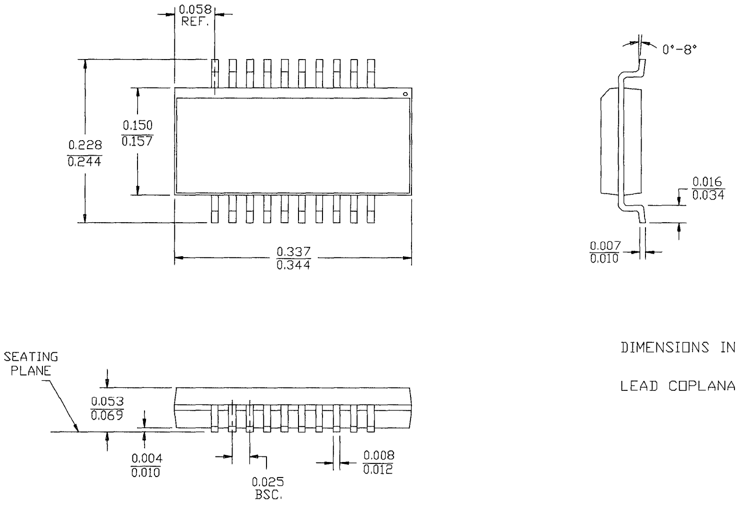
### 16-Lead Quarter Size Outline Q1



DIMENSIONS IN INCHES MIN./MAX.

LEAD COPLANARITY 0.004 MAX.

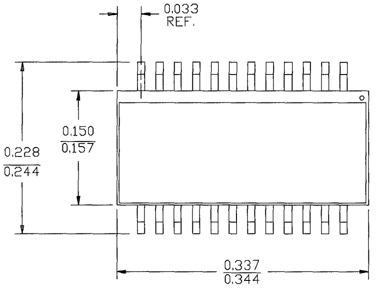
### 20-Lead Quarter Size Outline Q5

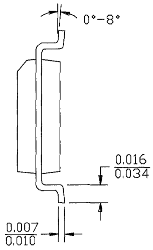


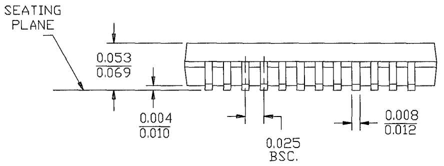
DIMENSIONS IN INCHES MIN./MAX.

LEAD COPLANARITY 0.004 MAX.

## 24-Lead Quarter Size Outline Q13







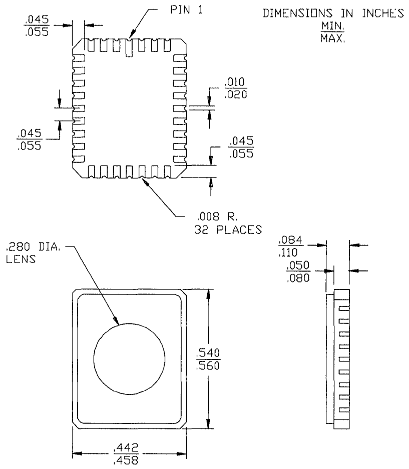
DIMENSIONS IN INCHES $\frac{\text{MIN.}}{\text{MAX.}}$

LEAD COPLANARITY 0.004 MAX.

## Ceramic Windowed Leadless Chip Carriers

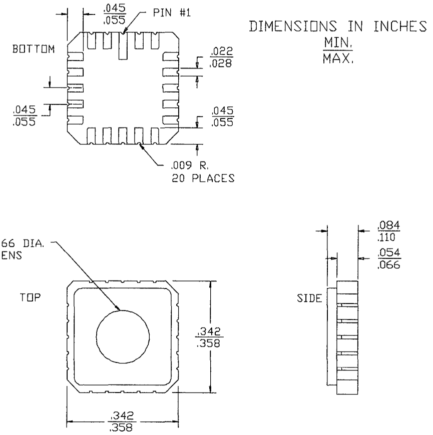
### 32-Pin Windowed Rectangular Leadless Chip Carrier Q55

MIL–STD–1835 C–12



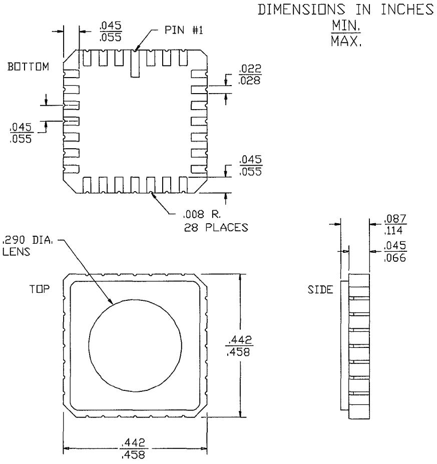
### 20-Pin Windowed Square Leadless Chip Carrier Q61

MIL–STD–1835 C–2A



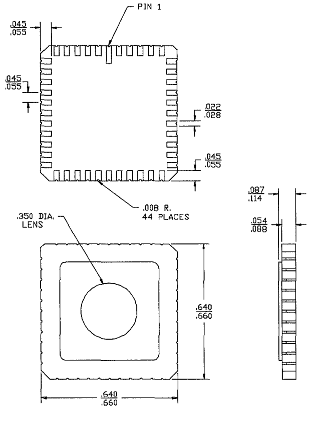
### 28-Pin Windowed Leadless Chip Carrier Q64

MIL–STD–1835 C–4



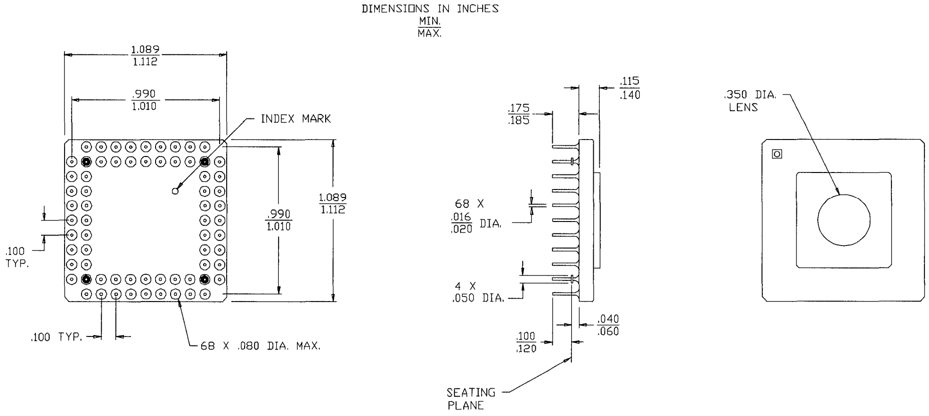
### 44-Pin Windowed Leadless Chip Carrier Q67

MIL–STD–1835 C–5

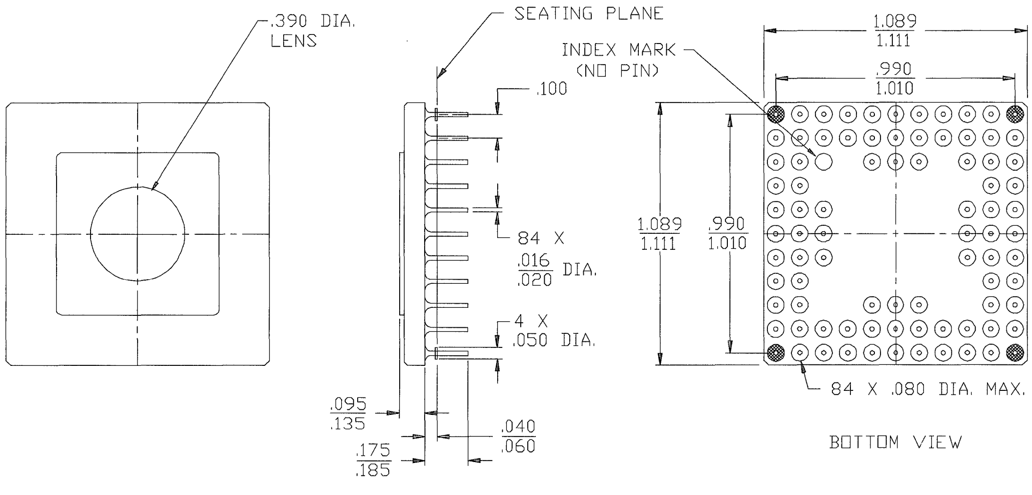


## Ceramic Windowed Pin Grid Arrays

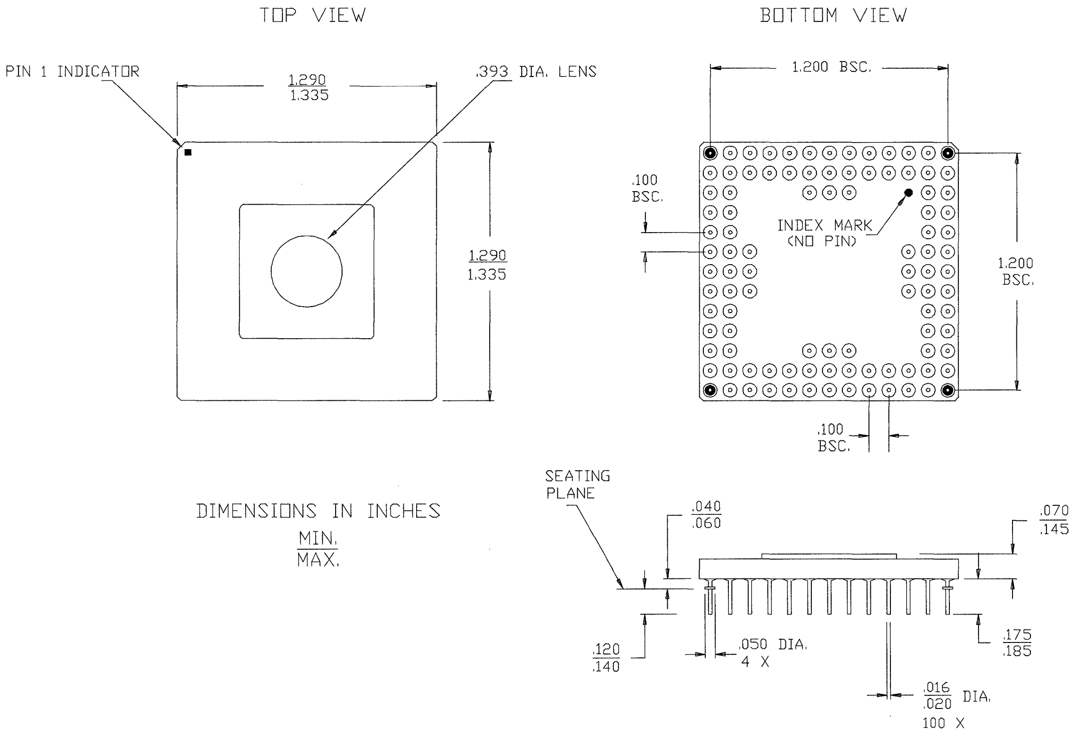
### 68-Pin Windowed PGA Ceramic R68



### 84-Lead Windowed Pin Grid Array R84

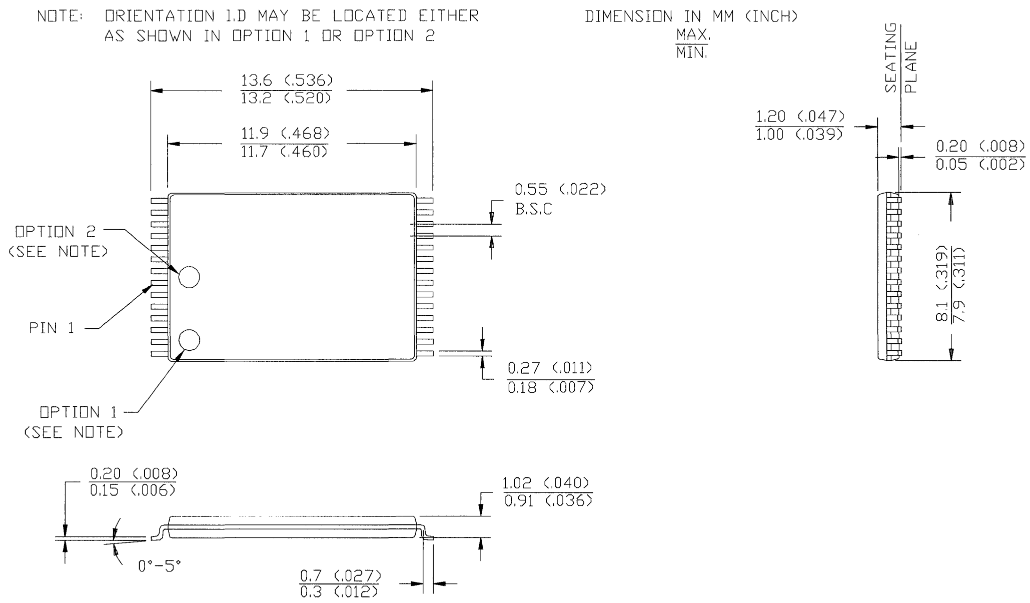


### 100-Pin Windowed Ceramic Pin Grid Array R100



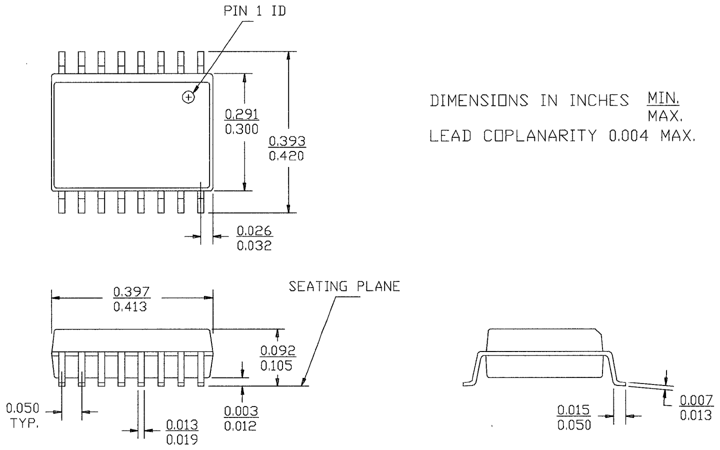
## Reverse Thin Small Outline Package

### 28-Lead Reverse Thin Small Outline Package RZ28



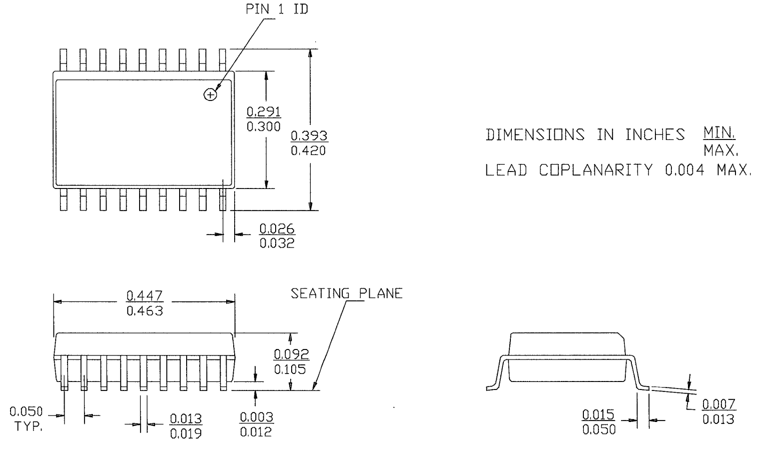
## Plastic Small Outline ICs

### 16-Lead Molded SOIC S1



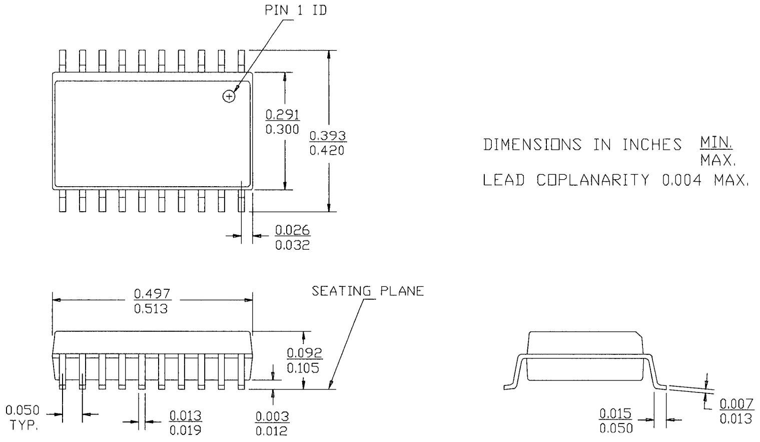
DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

### 18-Lead (300-Mil) Molded SOIC S3



DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

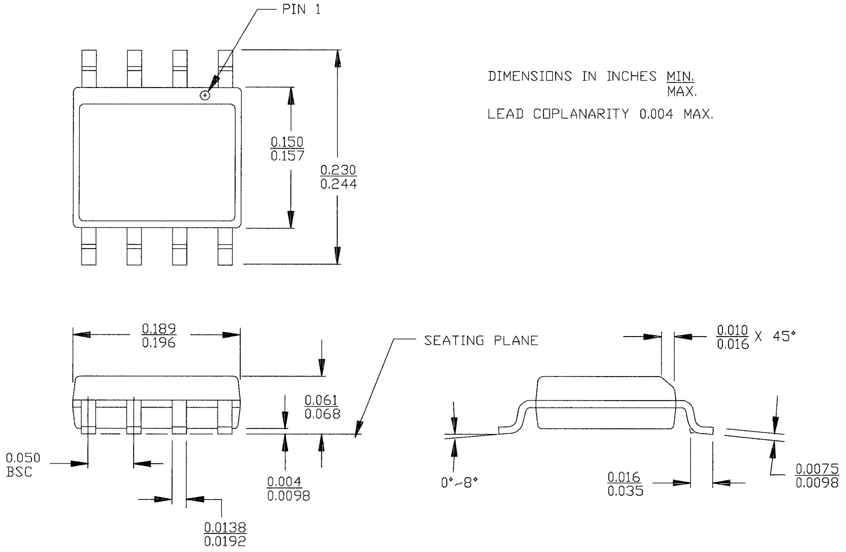
## 20-Lead (300-Mil) Molded SOIC S5



DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

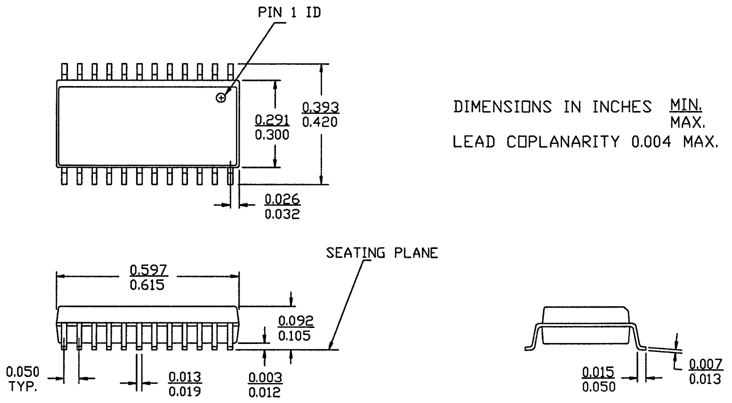
## 8-Lead (150-Mil) SOIC S8

PIN 1 ID IS OPTIONAL,
ROUND ON SINGLE LEADFRAME
RECTANGULAR ON MATRIX LEADFRAME



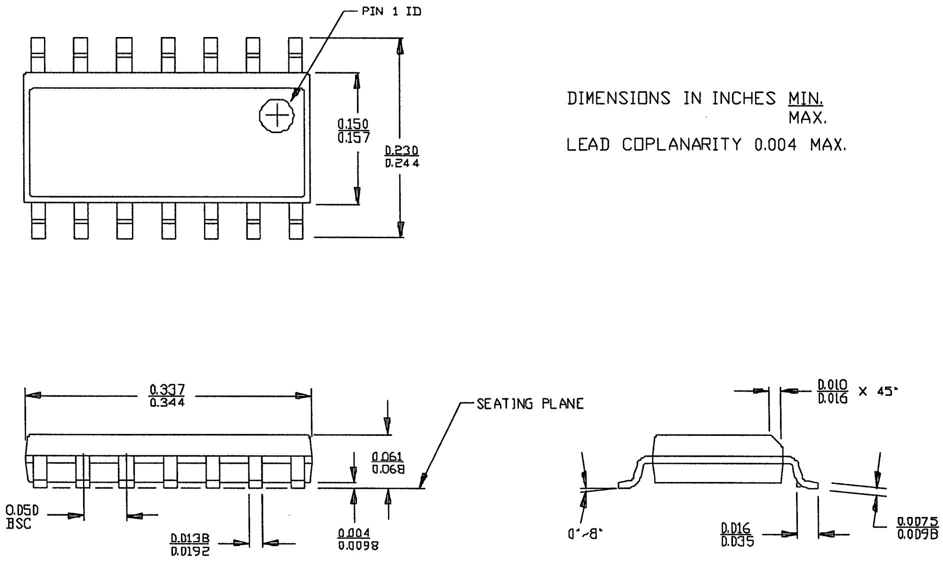
DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

## 24-Lead (300-Mil) Molded SOIC S13



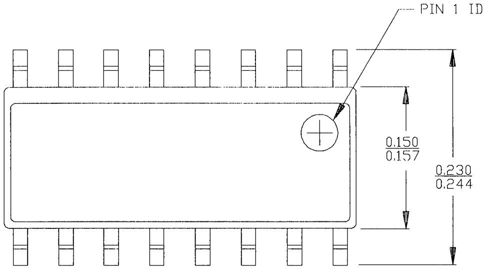
DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

## 14-Lead SOIC S14



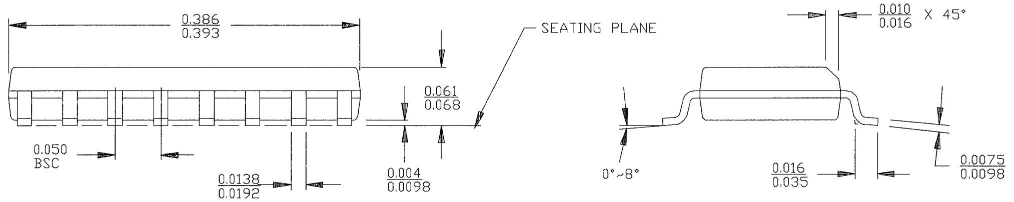
DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

## 16-Lead (150-Mil) Molded SOIC S16

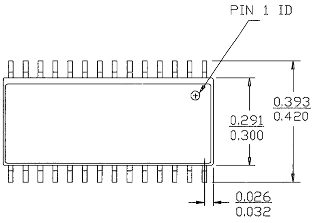


DIMENSIONS IN INCHES MIN.
MAX.

LEAD COPLANARITY 0.004 MAX.

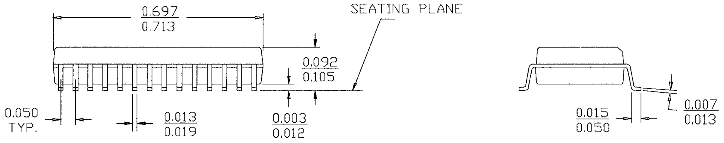


## 28-Lead (300-Mil) Molded SOIC S21

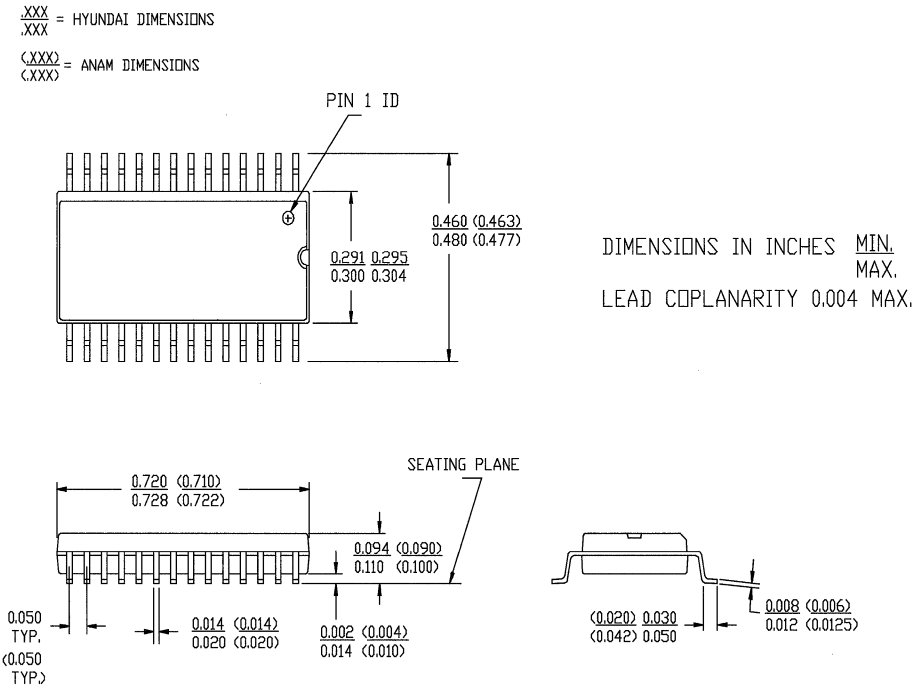


DIMENSIONS IN INCHES MIN.
MAX.

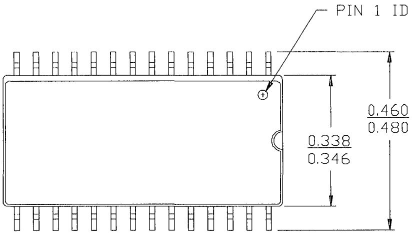
LEAD COPLANARITY 0.004 MAX.



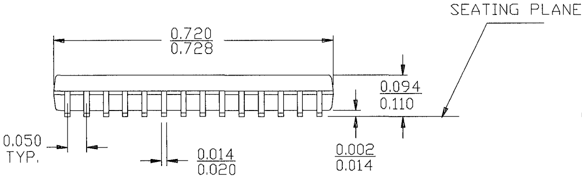
### 28-Lead 450-Mil (300-Mil Body Width) SOIC S22

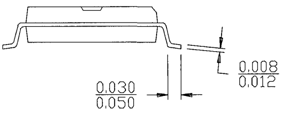


## 28-Lead (330-Mil) SOIC S23



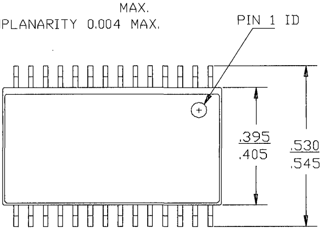
DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.

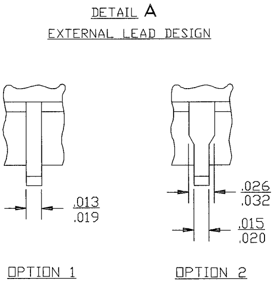


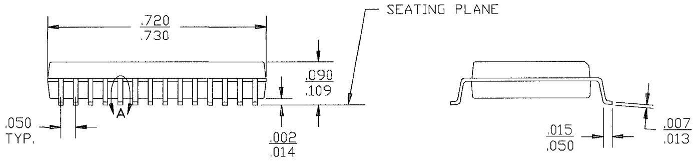


## 28-Lead (400-Mil) Molded SOIC S28

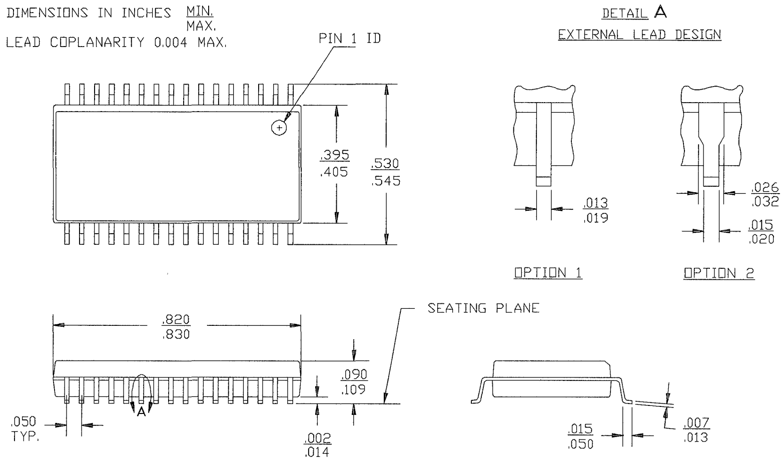
DIMENSIONS IN INCHES MIN.
MAX.
LEAD COPLANARITY 0.004 MAX.



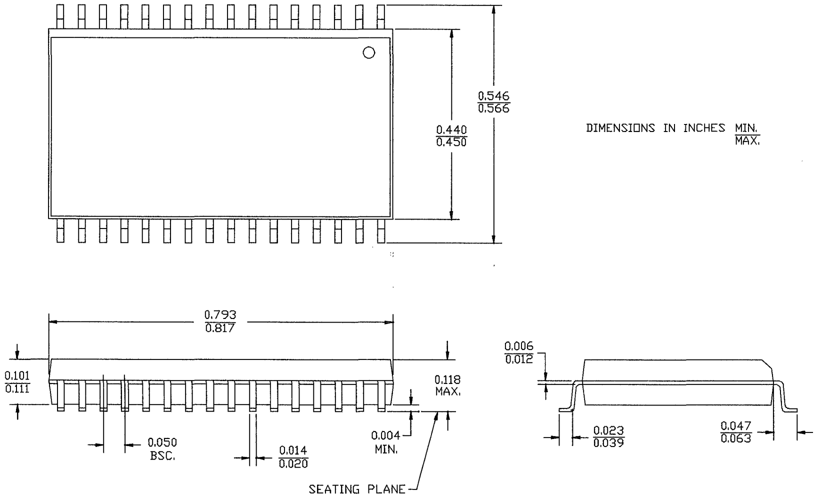




### 32-Lead (400-Mil) Molded SOIC S33

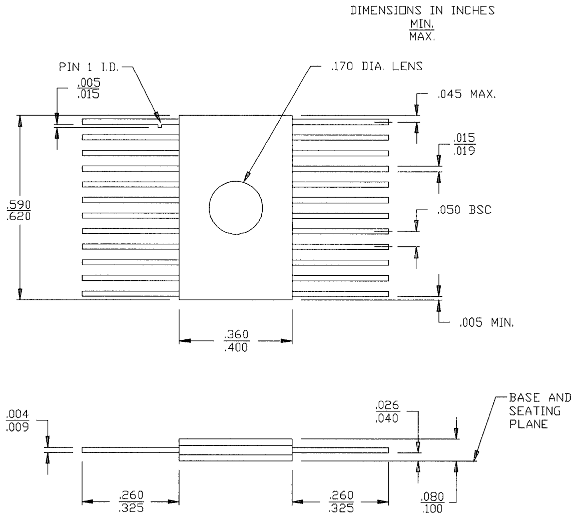


### 32-Lead (450 MIL) Molded SOIC S34

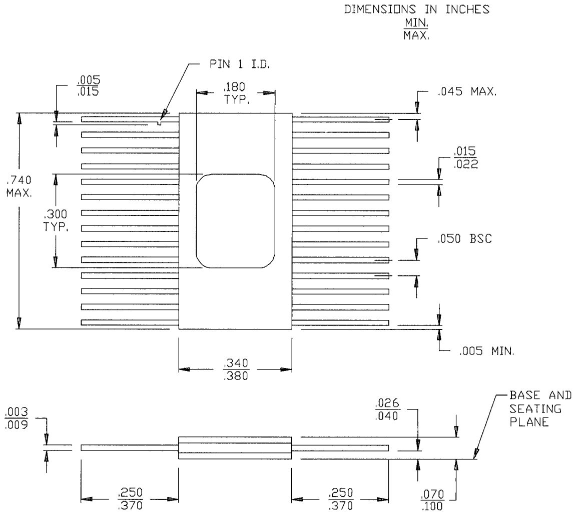


## Windowed Cerpacks

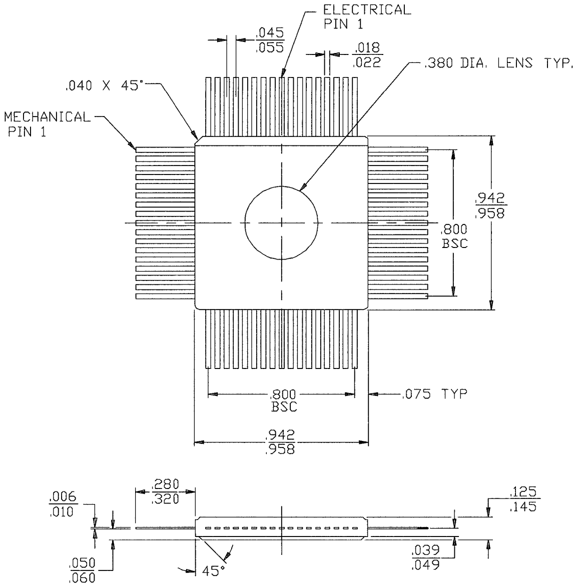
### 24-Lead Windowed Cerpack T73



### 28-Lead Windowed Cerpack T74

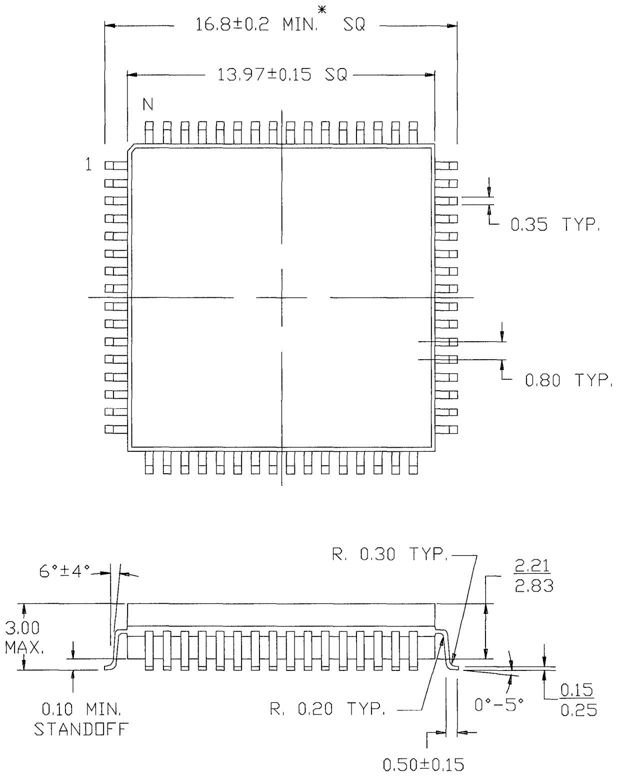


### 68-Lead Windowed Cerquad Flatpack T91



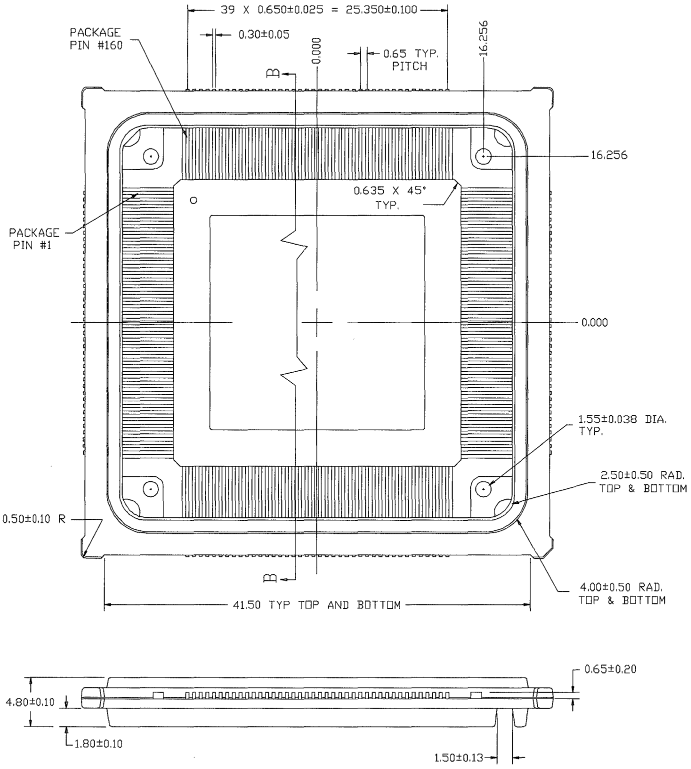
## Ceramic Quad Flatpacks

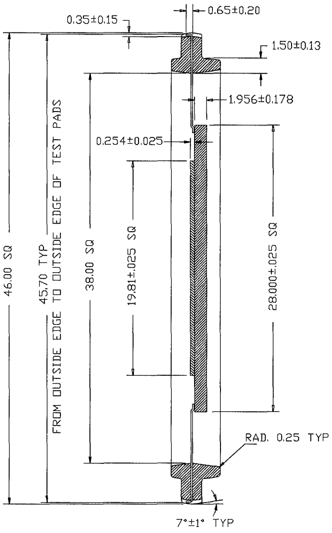
### 64-Lead Ceramic Quad Flatpack (Cavity Up) U65



DIMENSIONS IN MILLIMETERS
LEAD COPLANARITY 0.102 MAX.
DIMENSION MIN.
MAX.

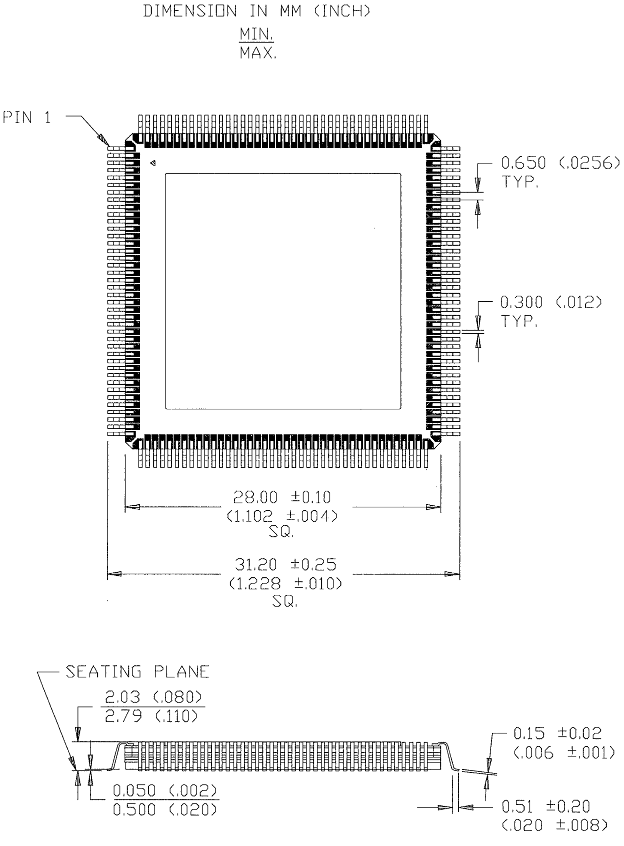
## 160-Lead Ceramic Quad Flatpack In Ring U160





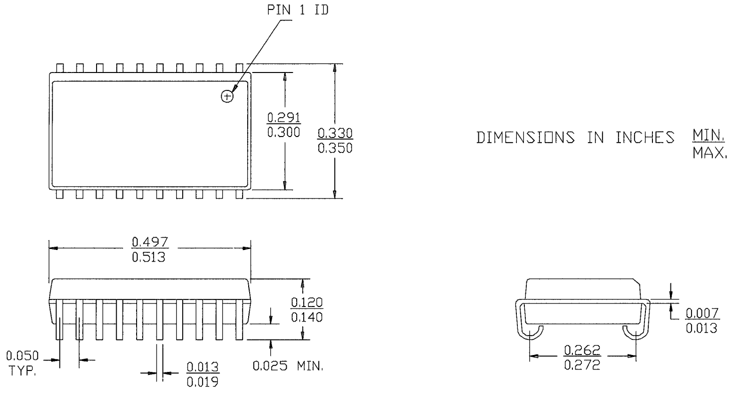
SECTION B-B

### 160-Lead Ceramic Quad Flatpack (Cavity Up) U162



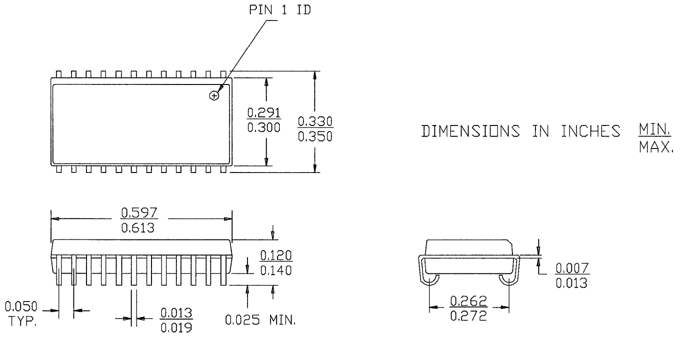
## Plastic Small Outline J - Bend

### 20-Lead (300-Mil) Molded SOJ V5



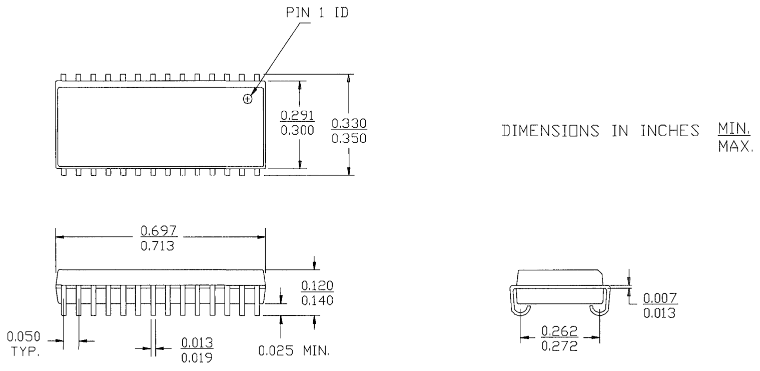
DIMENSIONS IN INCHES MIN. / MAX.

### 24-Lead (300-Mil) Molded SOJ V13

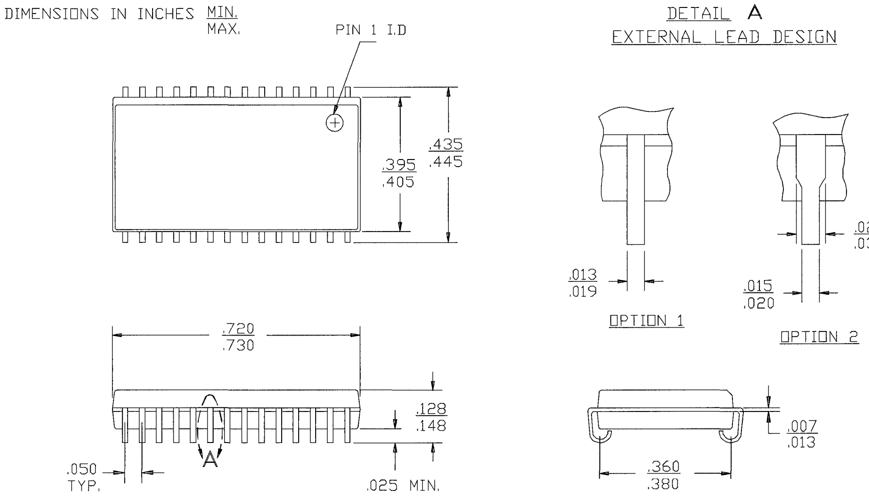


DIMENSIONS IN INCHES MIN. / MAX.

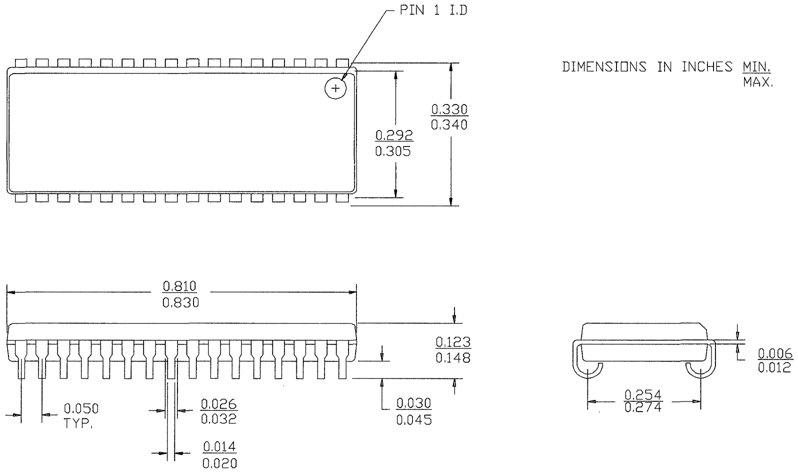
### 28-Lead (300-Mil) Molded SOJ V21



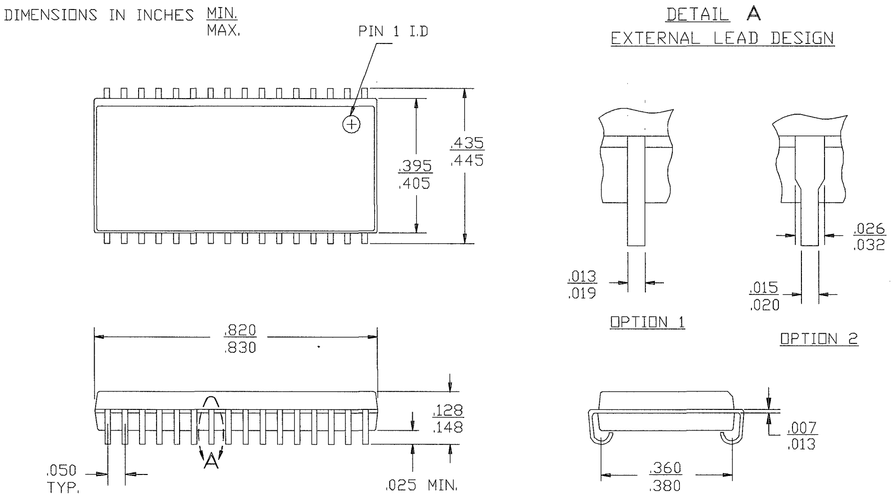
### 28-Lead (400-Mil) Molded SOJ V28



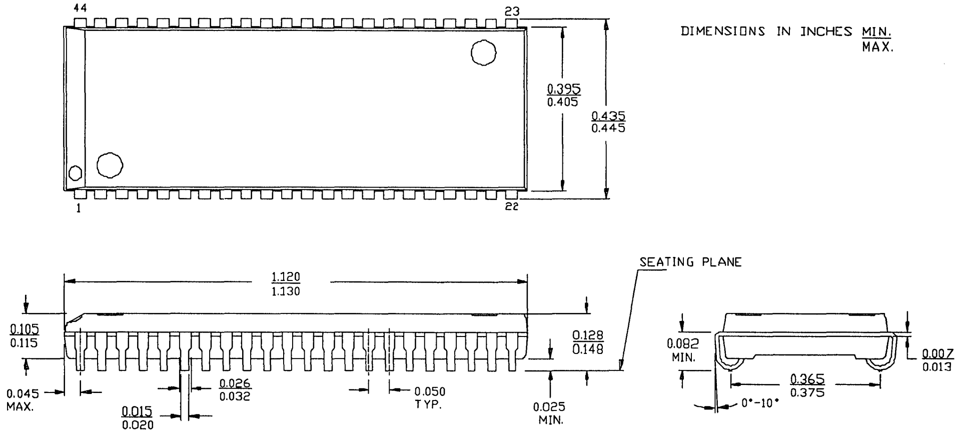
## 32-Lead (300-Mil) Molded SOJ V32



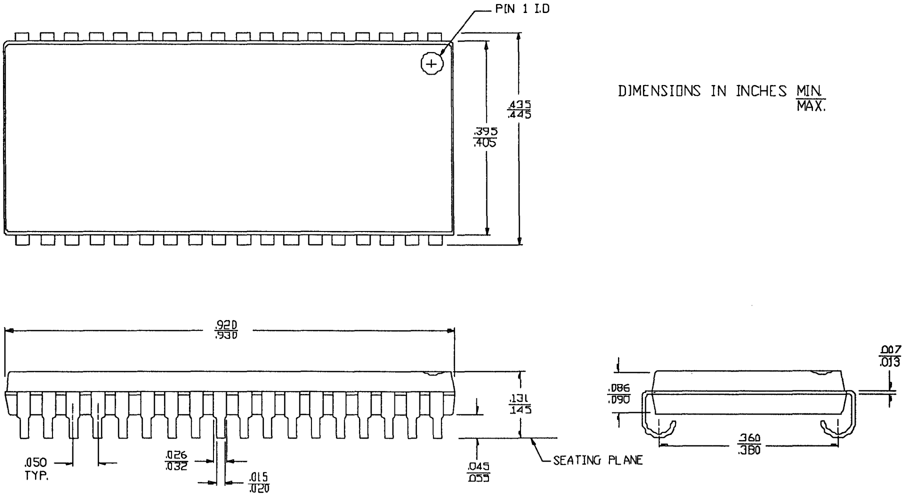
## 32-Lead (400-Mil) Molded SOJ V33



## 44-Lead (400-Mil) Molded SOJ V34



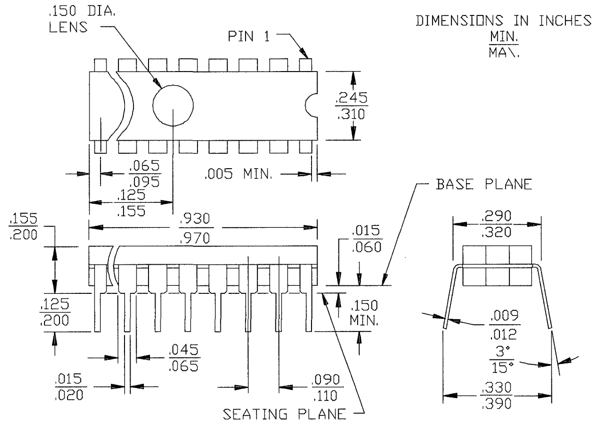
## 36-Lead (400-Mil) Molded SOJ



## Ceramic Windowed Dual-In-Line Packages

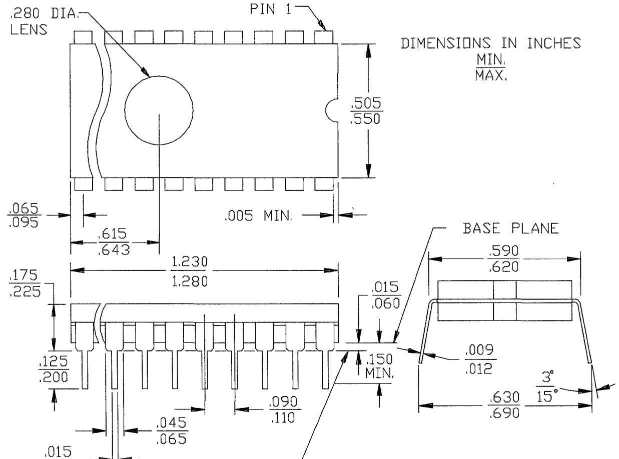
### 20-Lead (300-Mil) Windowed CerDIP W6

MIL-STD-1835 D-8 Config. A



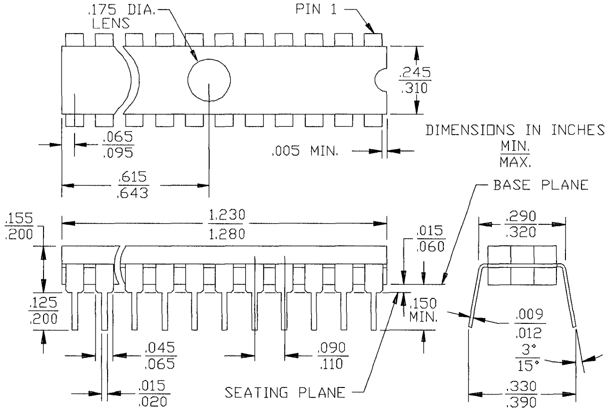
### 24-Lead (600-Mil) Windowed CerDIP W12

MIL-STD-1835 D-3 Config. A



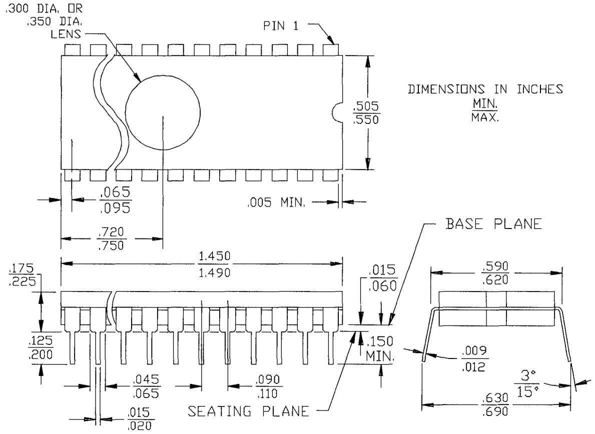
## 24-Lead (300-Mil) Windowed CerDIP W14

MIL-STD-1835 D-9 Config. A

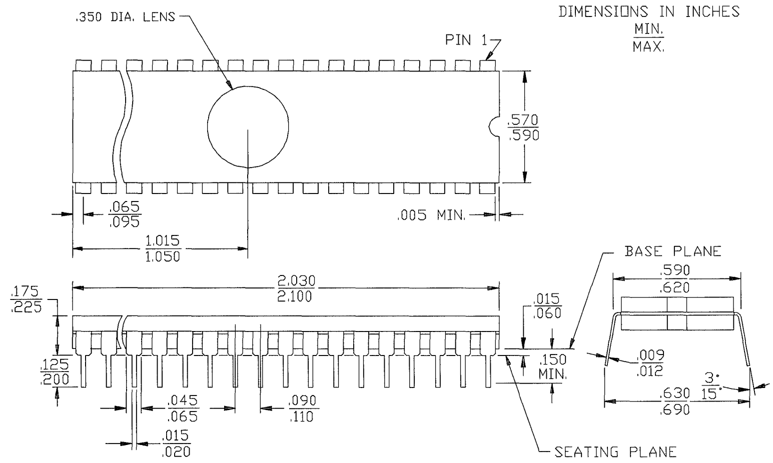


## 28-Lead (600-Mil) Windowed CerDIP W16

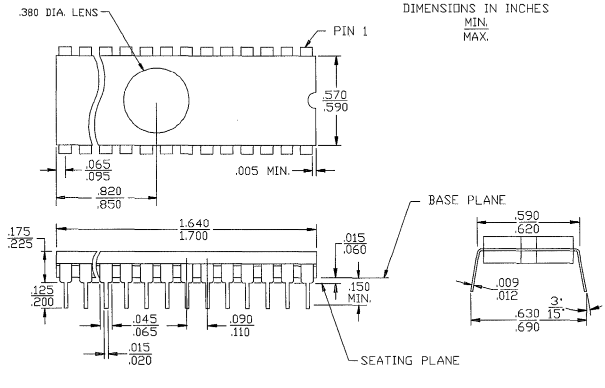
MIL-STD-1835 D-10 Config. A



### 40-Lead (600-Mil) Windowed CerDIP W18

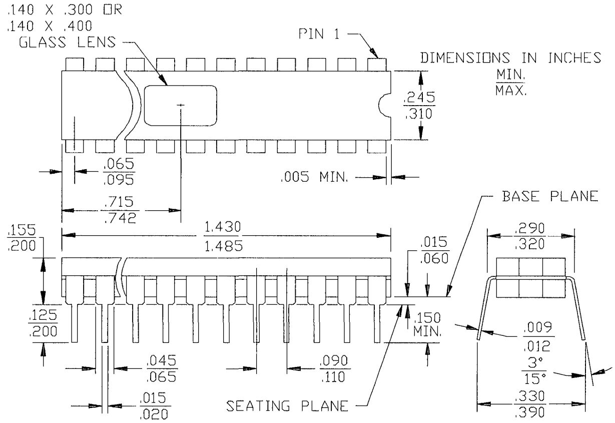


### 32-Lead (600-Mil) Windowed CerDIP W20

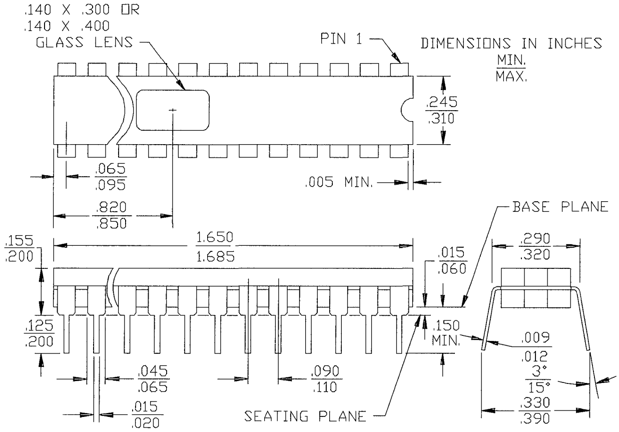


### 28-Lead (300-Mil) Windowed CerDIP W22

MIL-STD-1835 D-15 Config. A

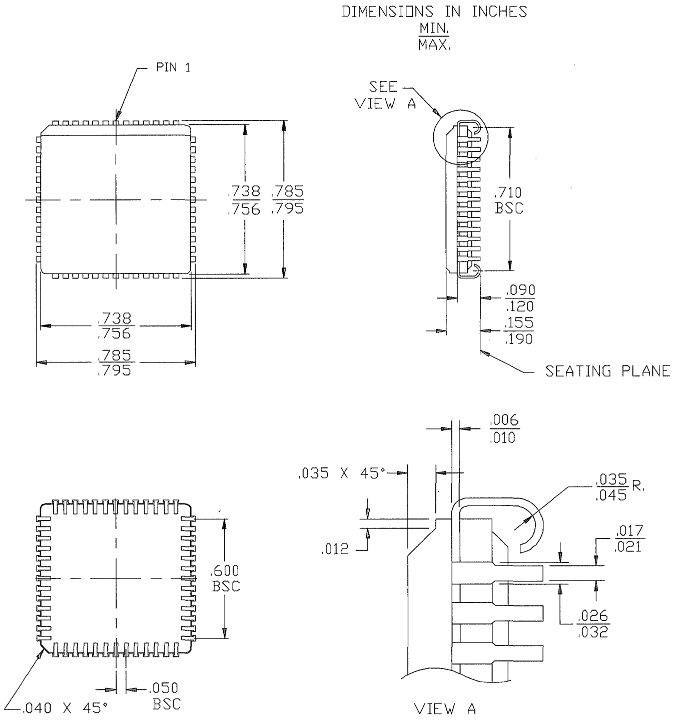


### 32-Lead (300-Mil) Windowed CerDIP W32

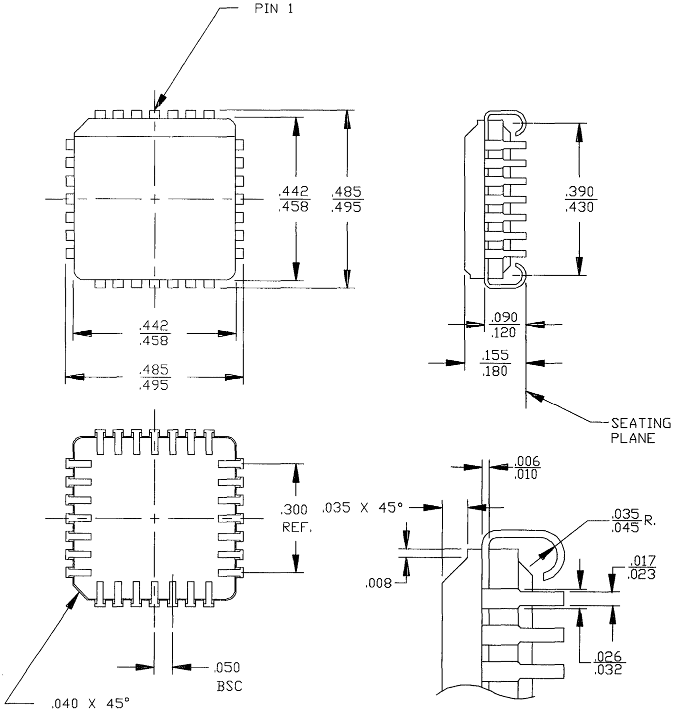


## Ceramic J-Leaded Chip Carriers

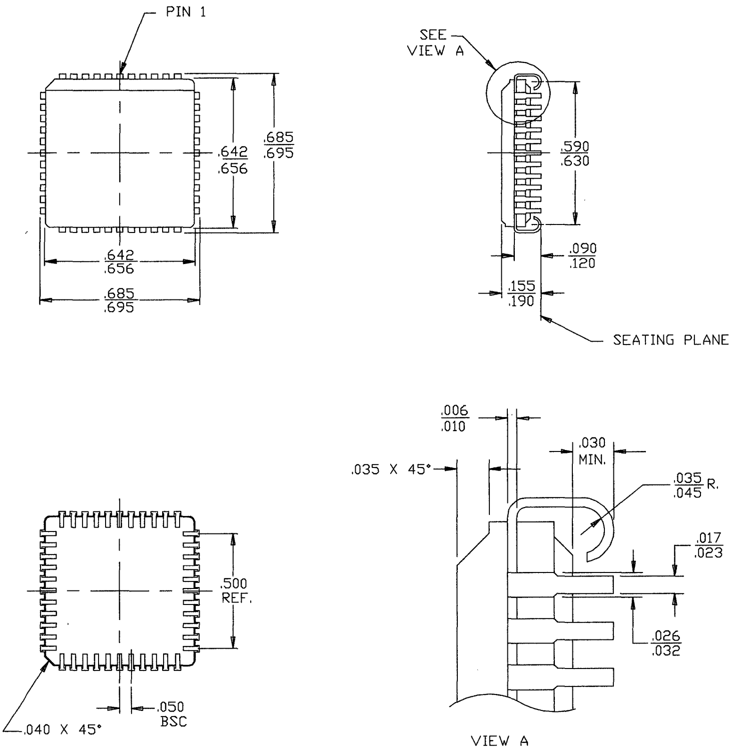
### 52-Pin Ceramic Leaded Chip Carrier Y59



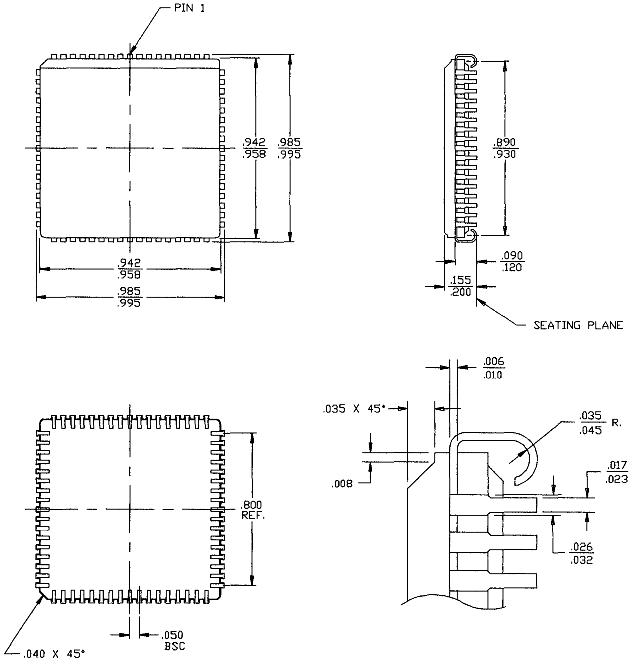
### 28-Pin Ceramic Leaded Chip Carrier Y64



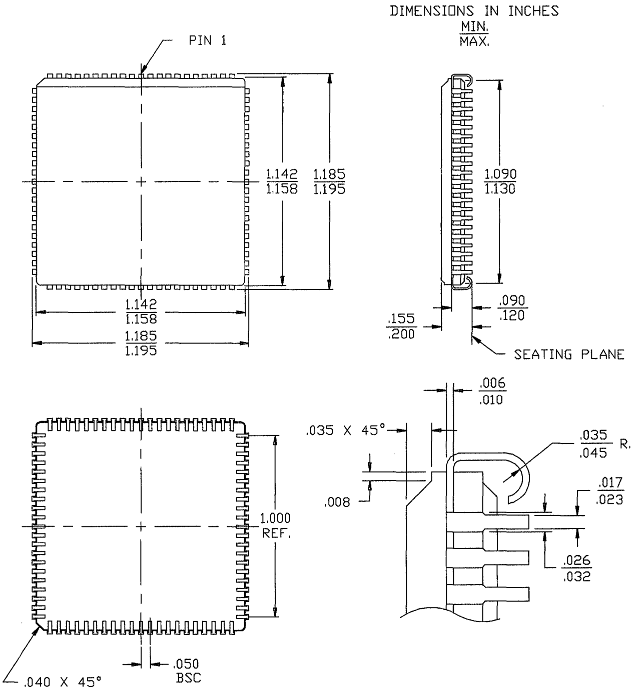
### 44-Pin Ceramic Leaded Chip Carrier Y67



## 68-Pin Ceramic Leaded Chip Carrier Y68

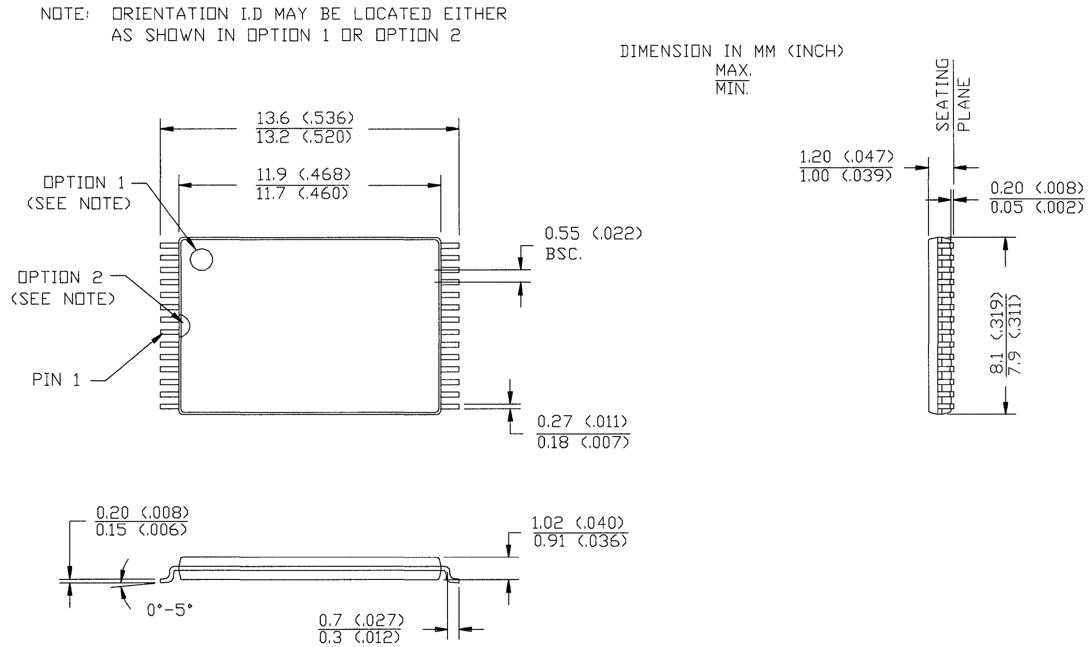


### 84-Pin Ceramic Leaded Chip Carrier Y84

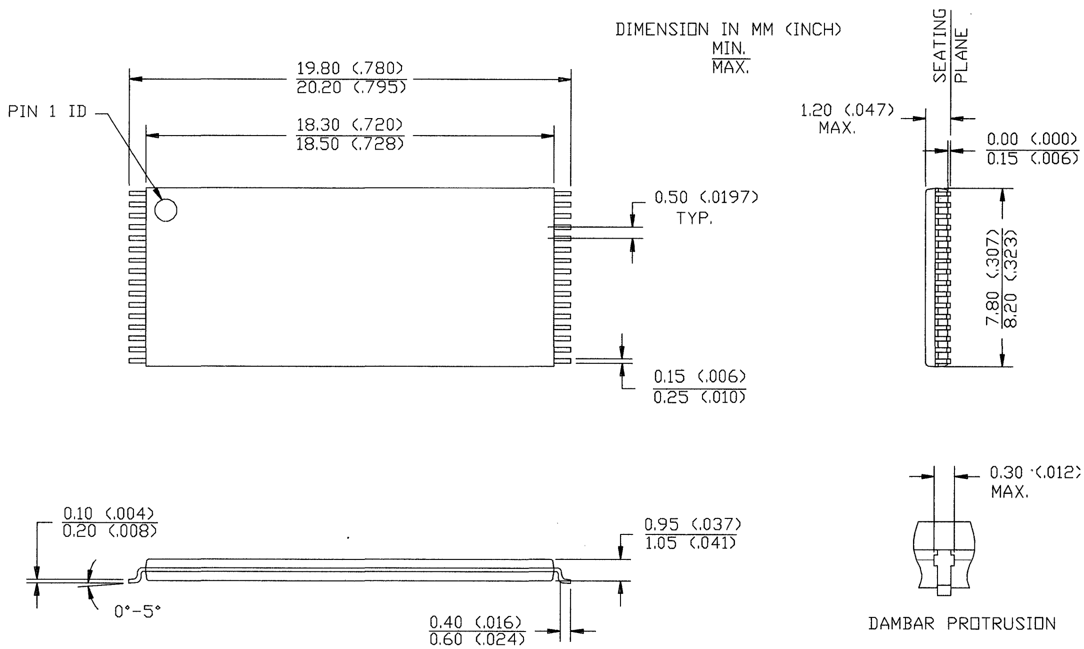


## Thin Small Outline Packages

### 28-Lead Thin Small Outline Package Z28



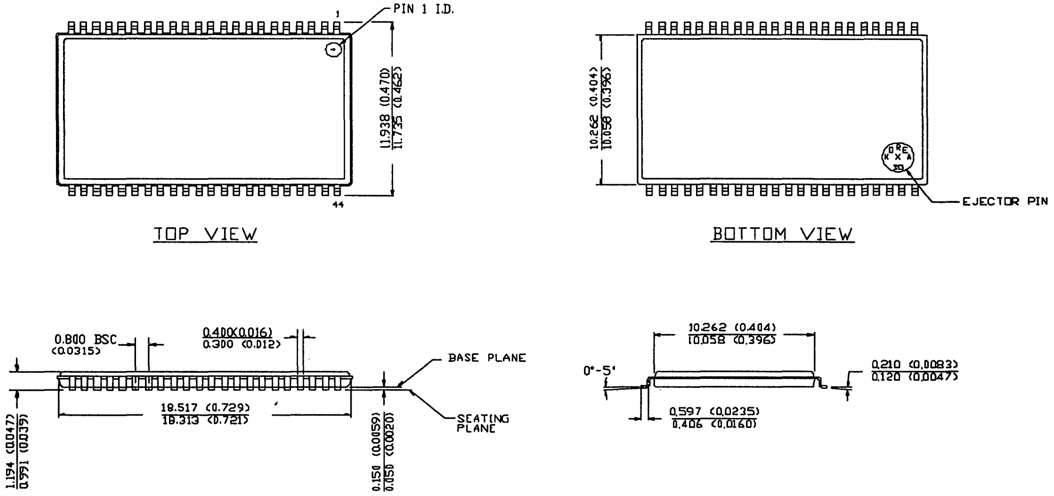
## 32-Lead Thin Small Outline Package Z32



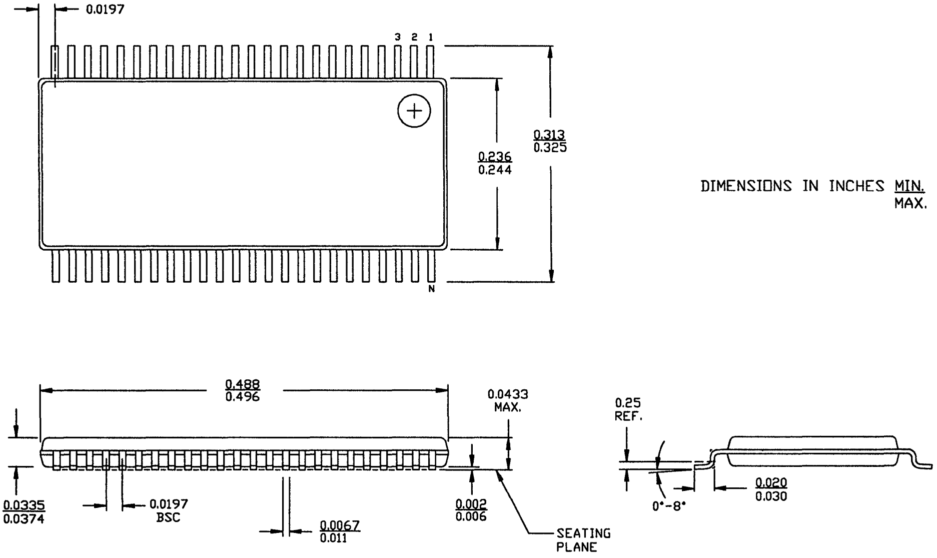
## 44-Pin TSOP II Z44

DIMENSION IN MM (INCH)
MAX
MIN

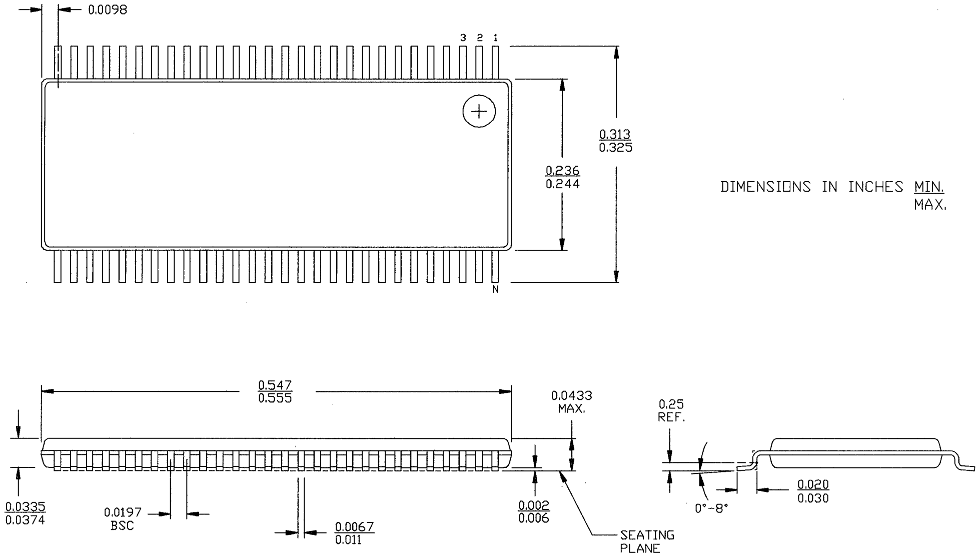
LEAD COPLANARITY 0.004 INCHES.



## 48-Lead Thin Shrunk Small Outline Package Z48



### 56-Lead Thin Shrunk Small Outline Package Z56



DIMENSIONS IN INCHES MIN.
MAX.

CYPRESS

## Cypress Semiconductor

### CORPORATE HEADQUARTERS

**Cypress Semiconductor**
3901 North First Street
San Jose, CA 95134
**Tel.: (800) 858-1810** (toll-free in U.S.)
Press "1" to reach your local sales rep.
**FAX: 408-943-6848**
Internet:
***http://www.cypress.com***

### EUROPE HEADQUARTERS

**Cypress International**
Dreve Richelle 161, Building C
1410 Waterloo
Belgium
**Tel.: (32) 02-357-0220**
**FAX: (32) 02-357-0230**

### ASIA HEADQUARTERS

**Cypress Semiconductor**
583 Orchard Road Forum #11-03
Singapore 238884
**Tel.: (65) 735-0338**
**FAX: (65) 735-0228**

### JAPAN HEADQUARTERS

**Cypress Semiconductor**
Shinjuku-Marune Bldg.
1-23-1 Shinjuku, Shinjuku-ku
Tokyo, Japan 160
**Tel.: (81) 3-5269-0781**
**FAX: (81) 3-5269-0788**

1-597PLDBK