intel®

16-Bit Embedded Controllers

Ahearn/Ericksen

Order Number 270646-002

intel® 16-Bit Embedded Controllers 1990

LITERATURE

To order Intel Literature or obtain literature pricing information in the U.S. and Canada call or write Intel Literature Sales. In Europe and other international locations, please contact your local sales office or distributor.

INTEL LITERATURE SALES
P.O. BOX 7641
Mt. Prospect, IL 60056-7641

In the U.S. and Canada
call toll free
(800) 548-4725

CURRENT HANDBOOKS

Product line handbooks contain data sheets,application notes, article reprints and other design information.

TITLE	LITERATURE ORDER NUMBER
SET OF 11 HANDBOOKS (Available in U.S. and Canada only)	**231003**
EMBEDDED APPLICATIONS	270648
8-BIT EMBEDDED CONTROLLERS	270645
16-BIT EMBEDDED CONTROLLERS	270646
16/32-BIT EMBEDDED PROCESSORS	270647
MEMORY	210830
MICROCOMMUNICATIONS (2 volume set)	231658
MICROCOMPUTER SYSTEMS	280407
MICROPROCESSORS	230843
PERIPHERALS	296467
PRODUCT GUIDE (Overview of Intel's complete product lines)	210846
PROGRAMMABLE LOGIC	296083
ADDITIONAL LITERATURE (Not included in handbook set)	
AUTOMOTIVE SUPPLEMENT	231792
COMPONENTS QUALITY/RELIABILITY HANDBOOK	210997
INTEL PACKAGING OUTLINES AND DIMENSIONS (Packaging types, number of leads, etc.)	231369
INTERNATIONAL LITERATURE GUIDE	E00029
LITERATURE PRICE LIST (U.S. and Canada) (Comprehensive list of current Intel Literature)	210620
MILITARY (2 volume set)	210461
SYSTEMS QUALITY/RELIABILITY	231762

intel

U.S. and CANADA LITERATURE ORDER FORM

NAME: ______________________________

COMPANY: ______________________________

ADDRESS: ______________________________

CITY: ____________________ STATE: __________ ZIP: __________

COUNTRY: ______________________________

PHONE NO.: (   ) ______________________________

ORDER NO.	TITLE	QTY.		PRICE		TOTAL
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	
□□□□□□			×		=	

Subtotal __________

Must Add Your Local Sales Tax __________

Postage: add 10% of subtotal → Postage __________

Total __________

Pay by check, money order, or include company purchase order with this form ($100 minimum).We also accept VISA, MasterCard or American Express. Make payment to Intel Literature Sales. Allow 2-4 weeks for delivery.

□ VISA □ MasterCard □ American Express Expiration Date ____________________

Account No. ______________________________

Signature ______________________________

Mail To: Intel Literature Sales
P.O. Box 7641
Mt. Prospect, Il 60056-7641

International Customers outside the U.S. and Canada should use the International order form or contact their local Sales Office or Distributor.

For phone orders in the U.S. and Canada
Call Toll Free: (800) 548-4725

Prices good until 12/31/90.

Source HB

intel®

INTERNATIONAL LITERATURE ORDER FORM

NAME: ____

COMPANY: ____

ADDRESS: ____

CITY: ______________________ STATE: __________ ZIP: __________

COUNTRY: ____

PHONE NO.: (   ) ____________________________________

ORDER NO.	TITLE	QTY.		PRICE		TOTAL
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____
□□□□□□□	______________________	_____	×	_____	=	_____

Subtotal __________

Must Add Your
Local Sales Tax __________

Total __________

PAYMENT

Cheques should be made payable to your local Intel Sales Office

Other forms of payment may be available in your country. Please contact the Literature Coordinator at your local Intel Sales Office for details.

The completed form should be marked to the attention of the LITERATURE COORDINATOR and returned to your local Intel Sales Office.

Intel the Microcomputer Company:
When Intel invented the microprocessor in 1971, it created the era of microcomputers. Whether used in embedded applications such as automobiles or microwave ovens, or as the CPU in personal computers or supercomputers, Intel's microcomputers have always offered leading-edge technology. Intel continues to strive for the highest standards in memory, microcomputer components, modules and systems to give its customers the best possible competitive advantages.

16-BIT EMBEDDED CONTROLLER HANDBOOK

1990

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local sales office to obtain the latest specifications before placing your order.

The following are trademarks of Intel Corporation and may only be used to identify Intel Products:

> 376, 386, 387, 486, 4-SITE, Above, ACE51, ACE96, ACE186, ACE196, ACE960, BITBUS, COMMputer, CREDIT, Data Pipeline, ETOX, Genius, î, i486, i860, ICE, iCEL, ICEVIEW, iCS, iDBP, iDIS, I^{2}ICE, iLBX, iMDDX, iMMX, Inboard, Insite, Intel, int$_\mathrm{e}$l, Intel386, int$_\mathrm{e}$lBOS, Intel Certified, Intelevision, int$_\mathrm{e}$ligent Identifier, int$_\mathrm{e}$ligent Programming, Intellec, Intellink, iOSP, iPDS, iPSC, iRMK, iRMX, iSBC, iSBX, iSDM, iSXM, Library Manager, MAPNET, MCS, Megachassis, MICROMAINFRAME, MULTIBUS, MULTICHANNEL, MULTIMODULE, MultiSERVER, ONCE, OpenNET, OTP, PROMPT, Promware, QUEST, QueX, Quick-Erase, Quick-Pulse Programming, Ripplemode, RMX/80, RUPI, Seamless, SLD, SugarCube, UPI, and VLSiCEL, and the combination of ICE, iCS, iRMX, iSBC, iSBX, iSXM, MCS, or UPI and a numerical suffix.

MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation.

*MULTIBUS is a patented Intel bus.

CHMOS and HMOS are patented processes of Intel Corp.

Intel Corporation and Intel's FASTPATH are not affiliated with Kinetics, a division of Excelan, Inc. or its FASTPATH trademark or products.

Additional copies of this manual or other Intel literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 7641
Mt. Prospect, IL 60056-7641

CUSTOMER SUPPORT

INTEL'S COMPLETE SUPPORT SOLUTION WORLDWIDE

Customer Support is Intel's complete support service that provides Intel customers with hardware support, software support, customer training, consulting services and network management services. For detailed information contact your local sales offices.

After a customer purchases any system hardware or software product, service and support become major factors in determining whether that product will continue to meet a customer's expectations. Such support requires an international support organization and a breadth of programs to meet a variety of customer needs. As you might expect, Intel's customer support is quite extensive. It can start with assistance during your development effort to network management. 100 Intel sales and service offices are located worldwide—in the U.S., Canada, Europe and the Far East. So wherever you're using Intel technology, our professional staff is within close reach.

HARDWARE SUPPORT SERVICES

Intel's hardware maintenance service, starting with complete on-site installation will boost your productivity from the start and keep you running at maximum efficiency. Support for system or board level products can be tailored to match your needs, from complete on-site repair and maintenance support to economical carry-in or mail-in factory service.

Intel can provide support service for not only Intel systems and emulators, but also support for equipment in your development lab or provide service on your product to your end-user/customer.

SOFTWARE SUPPORT SERVICES

Software products are supported by our Technical Information Service (TIPS) that has a special toll free number to provide you with direct, ready information on known, documented problems and deficiencies, as well as work-arounds, patches and other solutions.

Intel's software support consists of two levels of contracts. Standard support includes TIPS (Technical Information Phone Service), updates and subscription service (product-specific troubleshooting guides and; *COMMENTS Magazine*). Basic support consists of updates and the subscription service. Contracts are sold in environments which represent product groupings (e.g., iRMX® environment).

CONSULTING SERVICES

Intel provides field system engineering consulting services for any phase of your development or application effort. You can use our system engineers in a variety of ways ranging from assistance in using a new product, developing an application, personalizing training and customizing an Intel product to providing technical and management consulting. Systems Engineers are well versed in technical areas such as microcommunications, real-time applications, embedded microcontrollers, and network services. You know your application needs; we know our products. Working together we can help you get a successful product to market in the least possible time.

CUSTOMER TRAINING

Intel offers a wide range of instructional programs covering various aspects of system design and implementation. In just three to ten days a limited number of individuals learn more in a single workshop than in weeks of self-study. For optimum convenience, workshops are scheduled regularly at Training Centers worldwide or we can take our workshops to you for on-site instruction. Covering a wide variety of topics, Intel's major course categories include: architecture and assembly language, programming and operating systems, BITBUS™ and LAN applications.

NETWORK MANAGEMENT SERVICES

Today's networking products are powerful and extremely flexible. The return they can provide on your investment via increased productivity and reduced costs can be very substantial.

Intel offers complete network support, from definition of your network's physical and functional design, to implementation, installation and maintenance. Whether installing your first network or adding to an existing one, Intel's Networking Specialists can optimize network performance for you.

Table of Contents

Alphanumeric Index

MCS®-96 8096BH Architectural Overview

1

This overview is written about the 8X9XBH devices, generically referred to as an 8096BH. An 8-bit bus version of this device, the 8098/8398, is described in the 8-bit Embedded Controller Handbook.

The 8096BH can be separated into several sections for the purpose of describing its operation. There is a 16-bit CPU, a programmable High Speed I/O Unit, an analog to digital converter, a serial port, and a Pulse Width Modulated (PWM) output for digital to analog conversion. In addition to these functional units, there are some sections which support overall operation of the chip such as the clock generator. The CPU and the programmable I/O make the 8096BH very different from any other microcontroller. Let us first examine the CPU.

1.0 CPU OPERATION

The major components of the CPU on the 8096BH are the Register File and the RALU. Communication with the outside world is done through either the Special Function Registers (SFRs) or the Memory Controller. The RALU (Register/Arithmetic Logic Unit) does not use an accumulator, it operates directly on the 256-byte register space made up of the Register File and the SFRs. Efficient I/O operations are possible by directly controlling the I/O through the SFRs. The main benefits of this structure are the ability to quickly change context, the absence of accumulator bottleneck, and fast throughput and I/O times.

1.1 CPU Buses

A "Control Unit" and two busses connect the Register File and RALU. Figure 1 shows the CPU with its

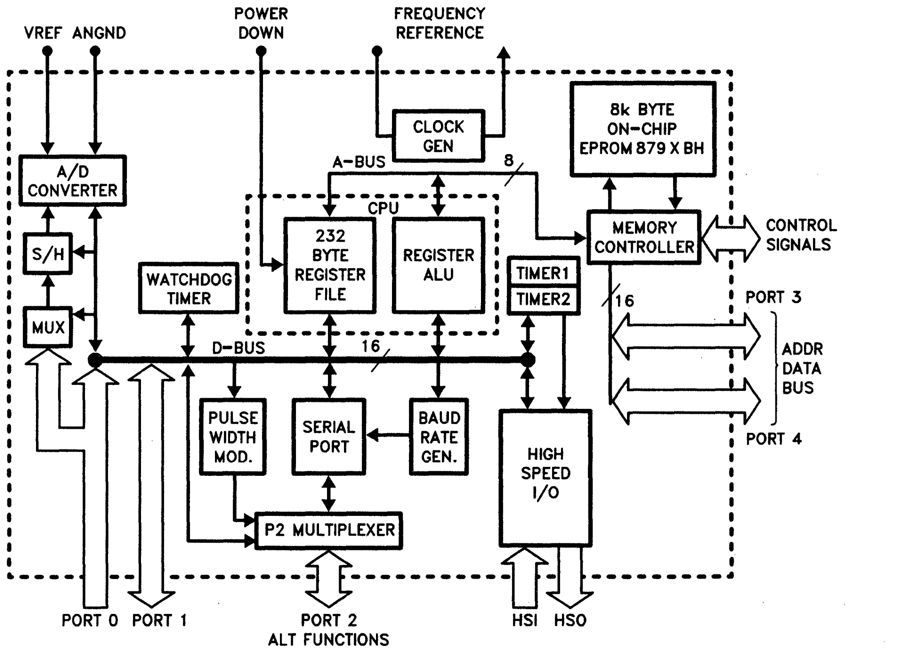


Figure 1. Block Diagram

major bus connections. The two buses are the "A-Bus" which is 8-bits wide, and the "D-Bus" which is 16-bits wide. The D-Bus transfers data only between the RALU and the Register File or Special Function Registers (SFRs). The A-Bus is used as the address bus for the above transfers or as a multiplexed address/data bus connecting to the "Memory Controller". Any accesses of either the internal ROM or external memory are done through the Memory Controller.

Within the memory controller is a slave program counter (Slave PC) which keeps track of the PC in the CPU. By having most program fetches from memory referenced to the slave PC, the processor saves time as addresses seldom have to be sent to the memory controller. If the address jumps sequence then the slave PC is loaded with a new value and processing continues. Data fetches from memory are also done through the memory controller, but the slave PC is bypassed for this operation.

1.2 CPU Register File

The Register File contains 232 bytes of RAM which can be accessed as bytes, words, or double-words. Since each of these locations can be used by the RALU, there are essentially 232 "accumulators". The first word in the Register File is reserved for use as the stack pointer so it can not be used for data when stack manipulations are taking place. Addresses for accessing the Register File and SFRs are temporarily stored in two 8-bit address registers by the CPU hardware.

1.3 RALU Control

Instructions to the RALU are taken from the A-Bus and stored temporarily in the instruction register. The Control Unit decodes the instructions and generates the correct sequence of signals to have the RALU perform the desired function. Figure 1 shows the instruction register and the control unit.

1.4 RALU

Most calculations performed by the 8096BH take place in the RALU. The RALU, shown in Figure 2, contains a 17-bit ALU, the Program Status Word (PSW), the Program Counter (PC), a loop counter, and three temporary registers. All of the registers are 16-bits or 17-bits (16+ sign extension) wide. Some of the registers have the ability to perform simple operations to offload the ALU.

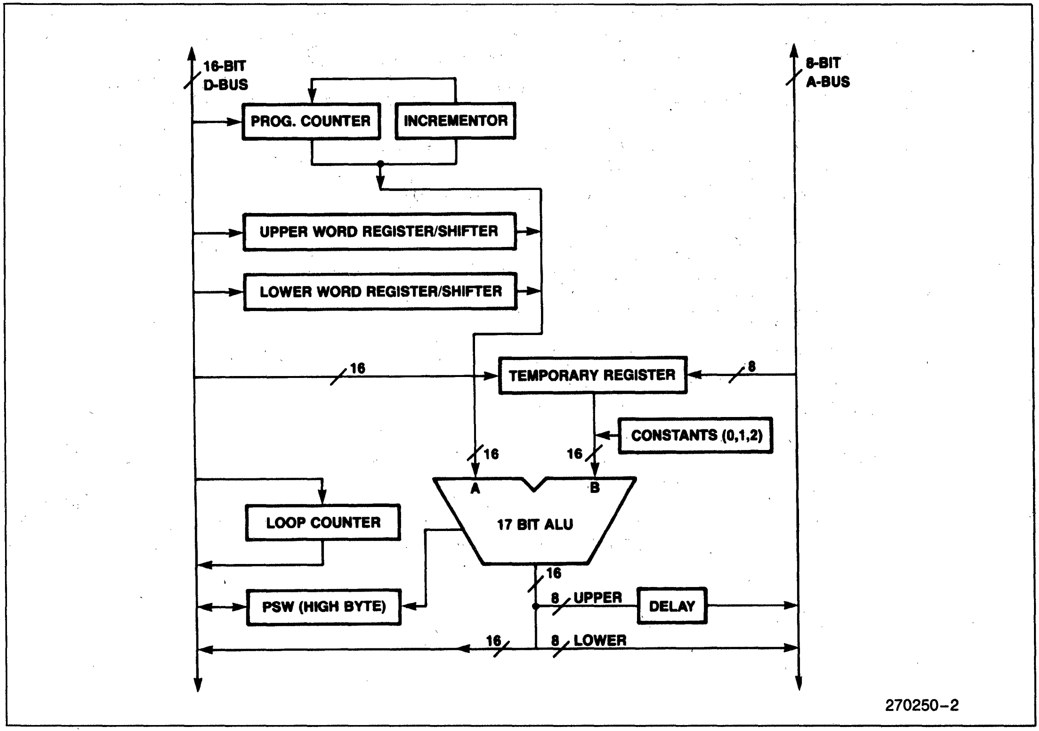


Figure 2. RALU Block Diagram

A separate incrementor is used for the PC; however, jumps must be handled through the ALU. Two of the temporary registers have their own shift logic. These registers are used for the operations which require logical shifts, including Normalize, Multiply, and Divide. The "Lower Word" register is used only when double-word quantities are being shifted, the "Upper Word" register is used whenever a shift is performed or as a temporary register for many instructions. Repetitive shifts are counted by the 5-bit "Loop Counter".

A temporary register is used to store the second operand of two operand instructions. This includes the multiplier during multiplications and the divisor during divisions. To perform subtractions, the output of this register can be complemented before being placed into the "B" input of the ALU.

The DELAY shown in Figure 2 is used to convert the 16-bit bus into an 8-bit bus. This is required as all addresses and instructions are carried on the 8-bit A-Bus. Several constants, such as 0, 1 and 2 are stored in the RALU for use in speeding up certain calculations. These come in handy when the RALU needs to make a 2's complement number or perform an increment or decrement instruction.

1.5 Internal Timing

The 8096BH requires an input clock frequency of between 6.0 MHz and 12 MHz to function. This frequency can be applied directly to XTAL1. Alternatively, since XTAL1 and XTAL2 are inputs and outputs of an inverter, it is also possible to use a crystal to generate the clock. A block diagram of the oscillator section is shown in Figure 3. Details of the circuit and suggestions for its use can be found in Section 1 of the Hardware Design chapter.

The crystal or external oscillator frequency is divided by 3 to generate the three internal timing phases as shown in Figure 4. Each of the internal phases repeat every 3 oscillator periods: 3 oscillator periods are referred to as one "state time", the basic time measurement for 8096BH operations. Most internal operations are synchronized to either Phase A, B or C, each of which have a 33% duty cycle. Phase A is represented externally by CLKOUT, a signal available on the 68-pin device. Phases B and C are not available externally. The relationships of XTAL1, CLKOUT, and Phases A, B, and C are shown in Figure 4. It should be noted that propagation delays have not been taken into account in this diagram. Details on these and other timing relationships can be found in the Hardware Design chapter.

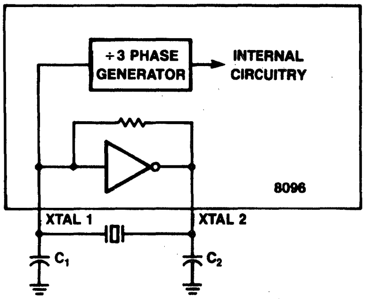


Figure 3. Block Diagram of Oscillator

The $\overline{\text{RESET}}$ line can be used to start the 8096BH at an exact time to provide for synchronization of test equipment and multiple chip systems. Use of this feature is fully explained under RESET, Section 13.

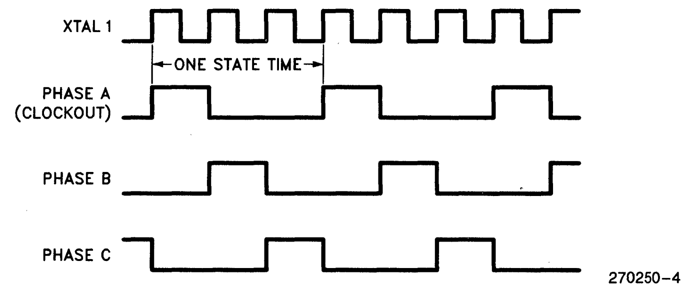


Figure 4. Internal Timings Relative to XTAL 1

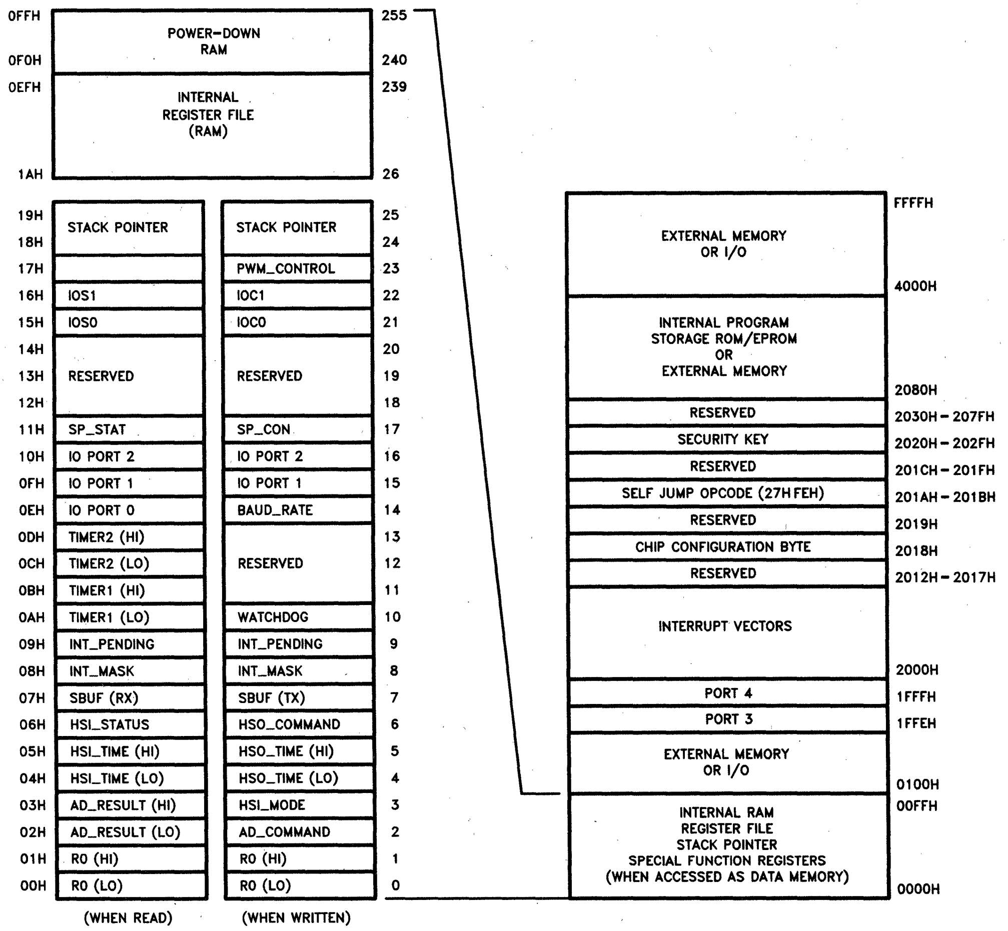
2.0 MEMORY SPACE

The addressable memory space on the 8096BH consists of 64K bytes, most of which is available to the user for program or data memory. Locations which have special purposes are 0000H through 00FFH and 1FFEH through 2080H. All other locations can be used for either program or data storage or for memory mapped peripherals. A memory map is shown in Figure 5.

2.1 Register File

Locations 00H through 0FFH contain the Register File and Special Function Registers, (SFRs). No code can be executed from this internal RAM section. If an attempt to execute instructions from locations 000H through 0FFH is made, the instructions will be fetched from *external* memory. This section of external memory is reserved for use by Intel development tools. Execution of a nonmaskable interrupt (NMI) will force a call to external location 0000H, therefore, the NMI and TRAP interrupt are also reserved for Intel development tools.

The RALU can operate on any of the 256 internal register locations. Locations 00H through 17H are used to access the SFRs. Locations 18H and 19H contain the stack pointer. These are not SFRs, and may be used as standard RAM if stack operations are not being performed. The stack pointer must be initialized by the



	(WHEN READ)	(WHEN WRITTEN)	
0FFH	POWER-DOWN RAM		255
0F0H			240
0EFH	INTERNAL REGISTER FILE (RAM)		239
1AH			26
19H	STACK POINTER	STACK POINTER	25
18H			24
17H		PWM_CONTROL	23
16H	IOS1	IOC1	22
15H	IOS0	IOC0	21
14H	RESERVED	RESERVED	20
13H			19
12H			18
11H	SP_STAT	SP_CON	17
10H	IO PORT 2	IO PORT 2	16
0FH	IO PORT 1	IO PORT 1	15
0EH	IO PORT 0	BAUD_RATE	14
0DH	TIMER2 (HI)	RESERVED	13
0CH	TIMER2 (LO)		12
0BH	TIMER1 (HI)		11
0AH	TIMER1 (LO)	WATCHDOG	10
09H	INT_PENDING	INT_PENDING	9
08H	INT_MASK	INT_MASK	8
07H	SBUF (RX)	SBUF (TX)	7
06H	HSI_STATUS	HSO_COMMAND	6
05H	HSI_TIME (HI)	HSO_TIME (HI)	5
04H	HSI_TIME (LO)	HSO_TIME (LO)	4
03H	AD_RESULT (HI)	HSI_MODE	3
02H	AD_RESULT (LO)	AD_COMMAND	2
01H	R0 (HI)	R0 (HI)	1
00H	R0 (LO)	R0 (LO)	0

EXTERNAL MEMORY OR I/O	FFFFH
INTERNAL PROGRAM STORAGE ROM/EPROM OR EXTERNAL MEMORY	4000H
RESERVED	2080H 2030H–207FH
SECURITY KEY	2020H–202FH
RESERVED	201CH–201FH
SELF JUMP OPCODE (27H FEH)	201AH–201BH
RESERVED	2019H
CHIP CONFIGURATION BYTE	2018H
RESERVED	2012H–2017H
INTERRUPT VECTORS	2000H
PORT 4	1FFFH
PORT 3	1FFEH
EXTERNAL MEMORY OR I/O	0100H
INTERNAL RAM REGISTER FILE STACK POINTER SPECIAL FUNCTION REGISTERS (WHEN ACCESSED AS DATA MEMORY)	00FFH 0000H

270250–5

Figure 5. Memory Map

user program and can point anywhere in the 64K memory space. The stack builds down. There are no restrictions on the use of the remaining 230 locations except that code cannot be executed from them.

2.2 Special Function Registers

All of the I/O on the 8096BH is controlled through the SFRs. Many of these registers serve two functions; one if they are read from, the other if they are written to. Figure 5 shows the locations and names of these registers. A summary of the capabilities of each of these registers is shown in Figure 6, with complete descriptions reserved for later sections.

There are several restrictions on using special function registers.

Neither the source or destination addresses of the Multiply and Divide instructions can be a writable special function register.

These registers may not be used as base or index registers for indirect or indexed instructions.

These registers can only be accessed as bytes unless otherwise specified in Figure 6. Note that some of these registers can only be accessed as words, and not as bytes.

Within the SFR space are several registers labeled "RESERVED". These registers are reserved for future expansion and test purposes. Operations should not be performed with these registers as reads from them and writes to them may produce unexpected results. For example, in some versions of the 8096BH writing to location OCH will set both timers to OFFFXH. This may not be the case in future products, so it should not be used as a feature.

2.3 Power Down

The upper 16 RAM locations (OFOH through OFFH) receive their power from the V_{PD} pin. If it is desired to keep the memory in these locations alive during a power down situation, one need only keep voltage on the V_{PD} pin. The current required to keep the RAM alive is approximately 1 milliamp (refer to the data sheet for the exact specification). Both V_{CC} and V_{PD} must have power applied for normal operation. If V_{PD} is not applied the power down RAM will not function properly, even if V_{CC} is applied.

To place the 8096BH into a power down mode, the $\overline{\text{RESET}}$ pin is pulled low. Two state times later the device will be in reset. This is necessary to prevent the device from writing into RAM as the power goes down. The power may now be removed from the V_{CC} pin, the V_{PD} pin must remain within specifications. The 8096BH can remain in this state for any amount of time and the 16 RAM bytes will retain their values.

To bring the 8096BH out of power down, $\overline{\text{RESET}}$ is held low while V_{CC} is applied. Two state times after the oscillator has stabilized, the $\overline{\text{RESET}}$ pin can be pulled high. The 8096BH will begin to execute code at location 02080H 10 state times after $\overline{\text{RESET}}$ is pulled high. Figure 7 shows a timing diagram of the power down sequence. To ensure that the 2 state time minimum reset time (synchronous with CLKOUT) is met, it is recommended that 10 XTAL1 cycles be used. Suggestions for actual hardware connections are given in the Hardware Design Chapter. Reset is discussed in Section 13.

To determine if a reset is a return from power down or a complete cold start a "key" can be written into power-down RAM while the device is running. This key can be checked on reset to determine which type of reset has occurred. In this way the validity of the power-down RAM can be verified. The length of this key determines the probability that this procedure will work, however, there is always a statistical chance that the RAM will power up with a replica of the key.

Under most circumstances, the power-fail indicator which is used to initiate a power-down condition must come from the unfiltered, unregulated section of the power supply. The power supply must have sufficient storage capacity to operate the 8096BH until it has completed its reset operation.

Register	Description	Section
R0	Zero Register — Always reads as a zero, useful for a base when indexing and as a constant for calculations and compares.	3
AD_RESULT	A/D Result Hi/Low — Low and high order Results of the A/D converter (byte read only)	8
AD_COMMAND	A/D Command Register — Controls the A/D	8
HSI_MODE	HSI Mode Register — Sets the mode of the High Speed Input unit.	6
HSI_TIME	HSI Time Hi/Lo — Contains the time at which the High Speed Input unit was triggered. (word read only)	6
HSO_TIME	HSO Time Hi/Lo — Sets the time or count for the High Speed Output to execute the command in the Command Register. (word write only)	7
HSO_COMMAND	HSO Command Register — Determines what will happen at the time loaded into the HSO Time registers.	7
HSI_STATUS	HSI Status Registers — Indicates which HSI pins were detected at the time in the HSI Time registers and the current state of the pins.	6
SBUF (TX)	Transmit buffer for the serial port, holds contents to be outputted.	9
SBUF (RX)	Receive buffer for the serial port, holds the byte just received by the serial port.	9
INT_MASK	Interrupt Mask Register — Enables or disables the individual interrupts.	4
INT_PENDING	Interrupt Pending Register — Indicates that an interrupt signal has occurred on one of the sources and has not been serviced.	4
WATCHDOG	Watchdog Timer Register — Written to periodically to hold off automatic reset every 64K state times.	12
TIMER1	Timer 1 Hi/Lo — Timer 1 high and low bytes. (word read only)	5
TIMER2	Timer 2 Hi/Lo — Timer 2 high and low bytes. (word read only)	5
IOPORT0	Port 0 Register — Levels on pins of port 0.	10
BAUD_RATE	Register which determines the baud rate, this register is loaded sequentially.	9
IOPORT1	Port 1 Register — Used to read or write to Port 1.	10
IOPORT2	Port 2 Register — Used to read or write to Port 2.	10
SP_STAT	Serial Port Status — Indicates the status of the serial port.	9
SP_CON	Serial Port Control — Used to set the mode of the serial port.	9
IOS0	I/O Status Register 0 — Contains information on the HSO status	11
IOS1	I/O Status Register 1 — Contains information on the status of the timers and of the HSI.	11
IOC0	I/O Control Register 0 — Controls alternate functions of HSI pins, Timer 2 reset sources and Timer 2 clock sources.	11
IOC1	I/O Control Register 1 — Controls alternate functions of Port 2 pins, timer interrupts and HSI interrupts.	11
PWM_CONTROL	Pulse Width Modulation Control Register — Sets the duration of the PWM pulse.	8

Figure 6. SFR Summary

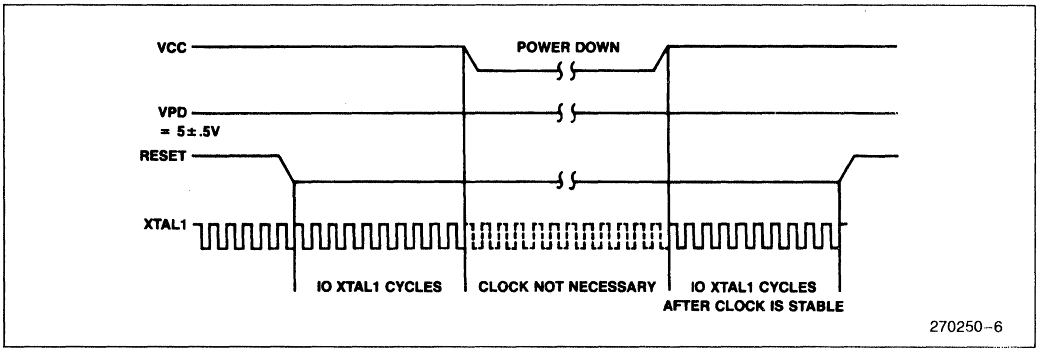


Figure 7. Power Down Timing

2.4 Reserved Memory Spaces

A listing of locations with special significance is shown in Figure 8. The locations marked "Reserved" are reserved by Intel for use in testing or future products. They must be filled with the Hex value FFH to insure compatibility with future devices.

Locations 1FFEH and 1FFFH are reserved for Ports 3 and 4 respectively. This is to allow easy reconstruction of these ports if external memory is used in the system. An example of reconstructing the I/O ports is given in section 7 of the Hardware Design chapter. If ports 3 and 4 are not going to be reconstructed, these locations can be treated as any other external memory location.

The 9 interrupt vectors are stored in locations 2000H through 2011H. The 9th vector is used by Intel development systems, as explained in Section 4.

Locations 2012H through 2017H are reserved for future use. Location 2018H is the Chip Configuration byte which will be discussed in the next section. The Jump-To-Self opcodes at locations 201AH and 201BH are provided for EPROM programming as detailed in the Hardware Design chapter. Locations 2020H through 202FH are the security key used with the ROM Lock feature which will be discussed in the next section. All unspecified addresses in locations 2000H through 207FH, including those marked Reserved, should be considered reserved for use by Intel.

Resetting the 8096BH causes instructions to be fetched starting from location 2080H. This location was chosen to allow a system to have up to 8K of RAM continuous with the register file. Further information on reset can be found in Section 13.

0000H-	0017H	Register Mapped I/O (SFRs)
0018H-	0019H	Stack Pointer
1FFEH-	1FFFH	Ports 3 and 4
2000H-	2011H	Interrupt Vectors
2012H-	2017H	Reserved
2018H		Chip Configuration Byte
2019H		Reserved
201AH-	201BH	"Jump to Self" Opcode (27H FEH)
201CH-	201FH	Reserved
2020H-	202FH	Security Key
2030H-	207FH	Reserved
2080H		Reset Location

Figure 8. Registers with Special Significance

2.5 Internal ROM and EPROM

When a ROM device is ordered, or an EPROM device is programmed, the internal memory locations 2080H through 3FFFH are user specified, as are the interrupt vectors, Chip Configuration Register and Security Key in locations 2000H through 202FH.

Instruction and data fetches from the internal ROM or EPROM occur only if the device has a ROM or EPROM, $\overline{EA}$ is tied high, and the address is between 2000H and 3FFFH. At all other times data is accessed from either the internal RAM space or external memory and instructions are fetched from external memory. The $\overline{EA}$ pin is latched on $\overline{RESET}$ rising. Information on programming EPROMs can be found in Section 10 of the Hardware Design chapter.

Do not execute code out of the last three locations of internal ROM/EPROM.

2.6 Memory Controller

The RALU talks to the memory (except for the locations in the register file and SFR space) through the memory controller which is connected to the RALU by the A-Bus and several control lines. Since the A-Bus is eight bits wide, the memory controller uses a Slave Program Counter to avoid having to always get the instruction location from the RALU. This slave PC is incremented after each fetch. When a jump or call occurs, the slave PC must be loaded from the A-Bus before instruction fetches can continue.

In addition to holding a slave PC, the memory controller contains a 4 byte queue to help speed execution. This queue is transparent to the RALU and to the user unless wait states are forced during external bus cycles. The instruction execution times shown in Section 14.8 show the normal execution times with no wait states added and the 16-bit bus selected. Reloading the slave PC and fetching the first byte of the new instruction stream takes 4 state times. This is reflected in the jump taken/not-taken times shown in the table.

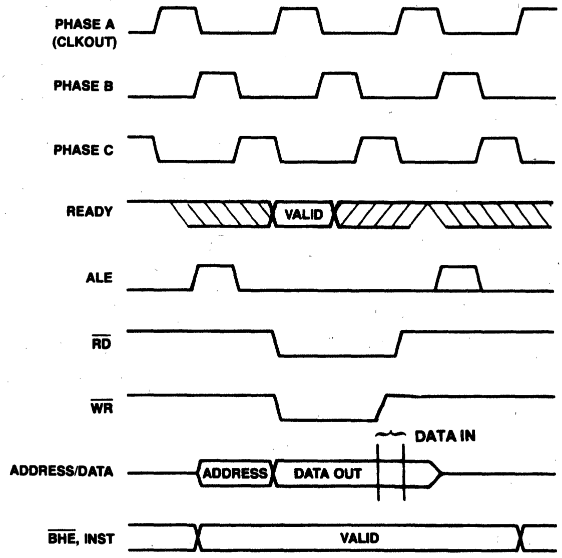
2.7 System Bus

There are several operating modes on the 8096BH. The standard bus mode uses a 16-bit multiplexed address/data bus. Other bus modes include an 8-bit mode and a mode in which the bus size can dynamically be switched between 8-bits and 16-bits. In addition, there are several options available on the type of control signals used by the bus.

In the standard mode, external memory is addressed through lines AD0 through AD15 which form a 16-bit multiplexed (address/data) data bus. These lines share pins with I/O Ports 3 and 4. The falling edge of the Address Latch Enable (ALE) line is used to provide a clock to a transparent latch (74LS373) in order to demultiplex the bus. A typical circuit and the required timings are shown in Section 7 of the Hardware Design chapter. Since the 8096BH's external memory can be addressed as either bytes or words, the decoding is controlled with two lines, Bus High Enable ($\overline{BHE}$) and Address/Data Line 0 (AD0).

To avoid confusion during the explanation of the memory system it is reasonable to give names to the demultiplexed address/data signals. The address signals will be called MA0 through MA15 (Memory Address), and the data signals will be called MD0 through MD15 (Memory Data).

When $\overline{BHE}$ is active (low), the memory connected to the high byte of the data bus should be selected. When MA0 is low the memory connected to the low byte of



270250–7

Figure 9. External Memory Timings

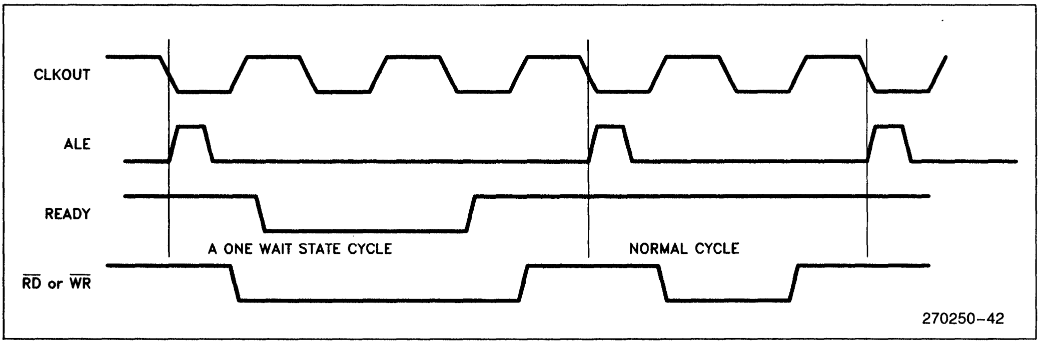


Figure 9A

the data bus should be selected. In this way accesses to a 16-bit wide memory can be to the low (even) byte only (MA0 = 0, $\overline{BHE}$ = 1), to the high (odd) byte only (MA0 = 1, $\overline{BHE}$ = 0), or to both bytes (MA0 = 0, $\overline{BHE}$ = 0). When a memory block is being used only for reads, $\overline{BHE}$ and MA0 need not be decoded.

TIMINGS

Figure 9 shows the idealized waveforms related to the following description of external memory manipulations. For exact timing specifications please refer to the latest data sheet. When an external memory fetch begins, the address latch enable (ALE) line rises, the address is put on AD0–AD15 and $\overline{BHE}$ is set to the required state. ALE then falls, the address is taken off the pins, and the $\overline{RD}$ (Read) signal goes low. When $\overline{RD}$ falls, external memory should present its data to the 8096BH.

READ

The data from the external memory must be on the bus and stable for a minimum of the specified set-up time before the rising edge of $\overline{RD}$. The rising edge of $\overline{RD}$ latches the information into the 8096BH. If the read is for data, the INST pin will be low when the address is valid, if it is for an instruction the INST pin will be high during this time. The 48-lead device does not have the INST pin. The INST pin will be low for the Chip Configuration Byte and Interrupt Vector fetches.

WRITE

Writing to external memory requires timings that are similar to those required when reading from it. The main difference is that the write ($\overline{WR}$) signal is used instead of the $\overline{RD}$ signal. The timings are the same until the falling edge of the $\overline{WR}$ line. At this point the 8096BH removes the address and places the data on the bus. When the $\overline{WR}$ line goes high the data should be latched to the external memory. In systems which can write to byte locations, the AD0 and $\overline{BHE}$ lines must be used to decode $\overline{WR}$ into WRite to Low byte ($\overline{WRL}$) and WRite to High byte ($\overline{WRH}$) signals. INST is always low during a write, as instructions cannot be written. The exact timing specifications for memory accesses can be found in the data sheet.

READY

A ready line is available on the 8096BH to extend the width of the RD and WR pulses in order to allow access of slow memories or for DMA purposes. If the READY line is low by the specified time after ALE falls, the 8096BH will hold the bus lines to their values at the falling edge of CLKOUT. When the READY line rises the bus cycle will continue with the next falling edge of CLKOUT. (See Figure 9A.)

Since the bus is synchronized to CLKOUT, it can be held only for an integral number of state times. If more than TYLYH nanoseconds are added the processor will act unpredictably.

There are several set-up and hold times associated with the READY signal. If these timings are not met, the device may not respond with the proper number of wait states.

For falling edges of READY, sampling is done internally on the falling edge of Phase A. Since Phase A generates CLKOUT, (after some propagation delay) the sample will be taken prior to CLKOUT falling. The timing specification for this is given as TLLYV, the time between when ALE falls and READY must be valid. If READY changes between TLLYV max and the falling edge of CLKOUT (TLLYH MIN on 48-lead devices) it would be possible to have the READY signal transitioning as it is being sampled.

This situation could cause a metastable condition which could make the device operate unpredictably.

For the rising edge of READY, sampling is done internally on the rising edge of Phase A. The rising edge logic is fully synchronized, so it is not possible to cause a metastable condition once the device is in a valid not-ready condition. To cause one wait state to occur the rising edge of READY must occur before TLLYH MAX after ALE falls. If the signal is brought up after this time two wait states may occur. If two wait states are desired, READY should be brought high within the TLLYH specification + 3 Tosc. Additional wait states can be caused by adding additional state times to the READY low time. The maximum amount of time that a device may be held not-ready is specified as TYLYH.

The 8096BH has the ability to internally limit the number of wait states to 1, 2, or 3 as determined by the value in the Chip Configuration Register, (CCR). Using the CCR for ready timing is discussed at the end of this section. If a ready limit is set, the TLLYH MAX specification is not used.

OPERATING MODES

The 8096BH supports a variety of options to simplify memory systems, interfacing requirements and ready control. Bus flexibility is provided by allowing selection of bus control signal definitions and runtime selection of the external bus width. In addition, several ready control modes are available to simplify the external hardware requirements for accessing slow devices. The Chip Configuration Register (CCR) is used to store the operating mode information.

CHIP CONFIGURATION REGISTER (CCR)

Configuration information is stored in the Chip Configuration Register (CCR). Four of the bits in the register specify the bus control mode and ready control mode. Two bits also govern the level of ROM/EPROM protection and one bit is NANDed with the BUSWIDTH pin every bus cycle to determine the bus size. The CCR bit map is shown in Figure 10. The functions associated with each bit are described in this section.

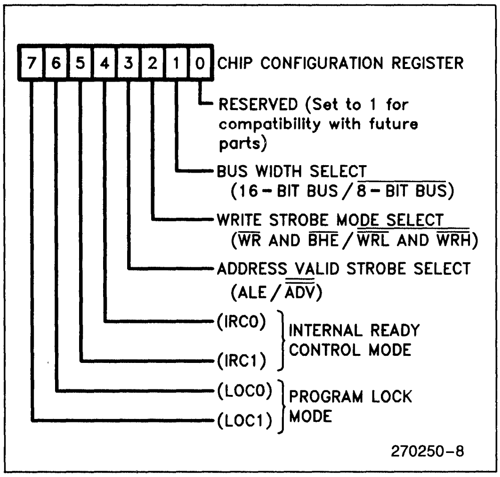

7 6 5 4 3 2 1 0
CHIP CONFIGURATION REGISTER
RESERVED (Set to 1 for compatibility with future parts)
BUS WIDTH SELECT (16-BIT BUS/$\overline{\text{8-BIT BUS}}$)
WRITE STROBE MODE SELECT ($\overline{\text{WR}}$ AND $\overline{\text{BHE}}$/$\overline{\text{WRL}}$ AND $\overline{\text{WRH}}$)
ADDRESS VALID STROBE SELECT (ALE/$\overline{\text{ADV}}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE
270250-8

Figure 10. Chip Configuration Register

The CCR is loaded on reset with the Chip Configuration Byte, located at address 2018H. The CCR register is a non-memory mapped location that can only be written to during the reset sequence; once it is loaded it cannot be changed until the next reset occurs. The 8096BH will correctly read this location in every bus mode.

If the $\overline{\text{EA}}$ pin is set to a logical 0, the access to 2018H comes from external memory. If $\overline{\text{EA}}$ is a logical 1, the access comes from internal ROM/EPROM. If $\overline{\text{EA}}$ is + 12.5V, the CCR is loaded with a byte from a separate non-memory-mapped location called PCCB (Programming CCB). The Programming mode is described in Section 10 of the Hardware Design chapter.

BUS WIDTH

The 8096BH external bus width can be run-time configured to operate as a standard 16-bit multiplexed address/data bus, or as an 8051 style 16-bit address/8-bit data bus.

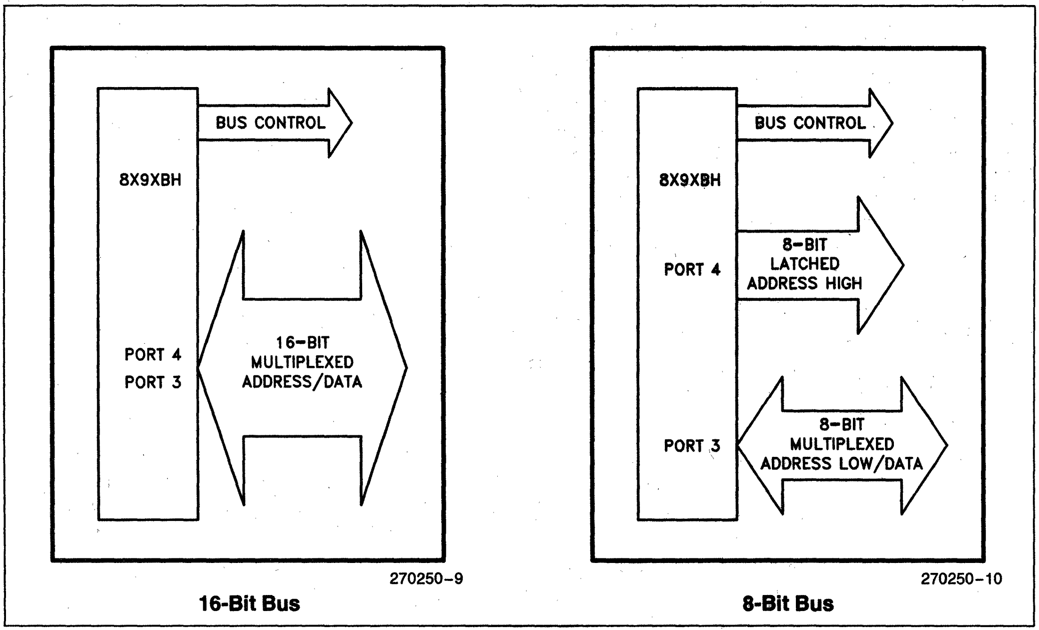
During 16-bit bus cycles, Ports 3 and 4 contain the address multiplexed with data using ALE to latch the address. In 8-bit bus cycles, Port 3 is multiplexed address/data while Port 4 is address bits 8 through 15. The address bits on Port 4 are valid throughout an 8-bit bus cycle. Figure 11 shows the two options.

The bus width can be changed each bus cycle and is controlled using bit 1 of the CCR with the BUSWIDTH pin. If either CCR.1 or BUSWIDTH is a 0, external accesses will be over a 16-bit address/8-bit data bus. If both CCR.1 and BUSWIDTH are 1s, external accesses will be over a 16-bit address/16-bit data bus. Internal accesses are always 16-bits wide.

The bus width can be changed every external bus cycle if a 1 was loaded into CCR bit 1 at reset. If this is the case, changing the value of the BUSWIDTH pin at runtime will dynamically select the bus width. For example, the user could feed the INST line into the BUSWIDTH pin, thus causing instruction accesses to be word wide from EPROMs while data accesses are byte wide to and from RAMs. A second example would be to place an inverted version of Address bit 15 on the BUSWIDTH pin. This would make half of external memory word wide, while half is byte wide.

Since BUSWIDTH is sampled after address decoding has had time to occur, even more complex memory maps could be constructed. See the timing specifications for an exact description of BUSWIDTH timings. The bus width will be determined by bit 1 of the CCR alone on 48-pin devices since they do not have a BUSWIDTH pin.

When using an 8-bit bus, some performance degradation is to be expected. On the 8096BH, instruction execution times with an 8-bit bus will slow down if any of three conditions occur. First, word writes to external memory will cause the executing instruction to take two extra state times to complete. Second, word reads from external memory will cause a one state time extension of instruction execution time. Finally, if the prefetch queue is empty when an instruction fetch is requested, instruction execution is lengthened by one state time for each byte that must be externally acquired (worst case is the number of bytes in the instruction minus one.)



16-Bit Bus

8-Bit Bus

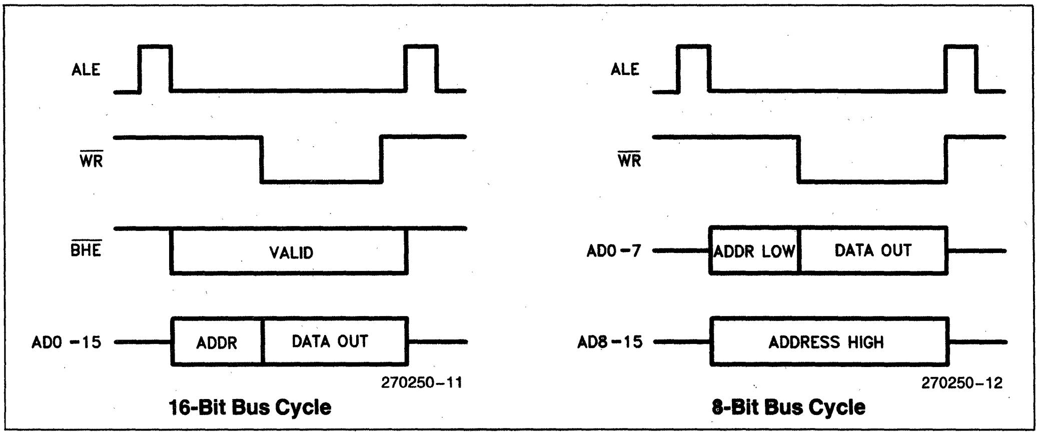
Figure 11. Bus Width Options

BUS CONTROL

Using the CCR, the 8096BH can be made to provide bus control signals of several types. Three control lines have dual functions designed to reduce external hardware. Bits 2 and 3 of the CCR specify the functions performed by these control lines. Figures 12–15 show the signals which can be modified by changing bits in the CCR, all other lines will operate as shown in Figure 9.

Standard Bus Control

If CCR bits 2 and 3 are 1s, then the standard 8096BH control signals $\overline{\text{WR}}$, $\overline{\text{BHE}}$ and ALE are provided (Figure 12). $\overline{\text{WR}}$ will come out for every write. $\overline{\text{BHE}}$ will be valid throughout the bus cycle and can be combined with $\overline{\text{WR}}$ and address line 0 to form $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$. ALE will rise as the address starts to come out, and will fall to provide the signal to externally latch the address.



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 12. Standard Bus Control

Write Strobe Mode

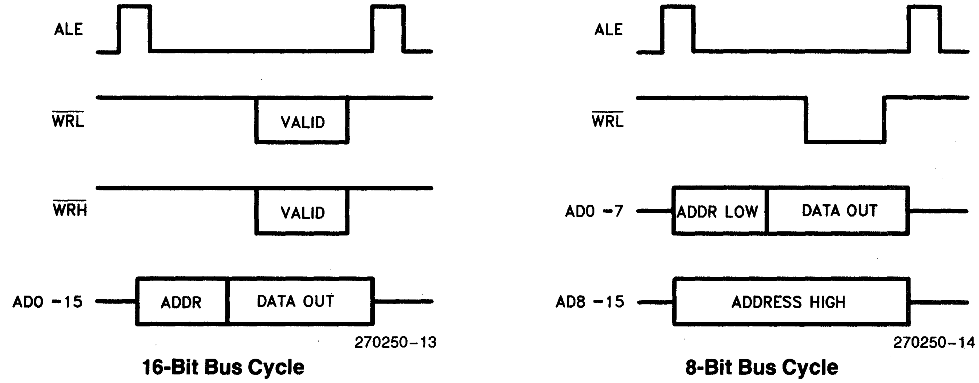
The Write Strobe Mode eliminates the necessity to externally decode for odd or even byte writes. If CCR bit 2 is a 0, and the bus is in a 16-bit cycle, $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$ signals are provided in place of $\overline{\text{WR}}$ and $\overline{\text{BHE}}$ (Figure 13). $\overline{\text{WRL}}$ will go low for all byte writes to an even address and all word writes. $\overline{\text{WRH}}$ will go low for all byte writes to an odd address and all word writes.

Write Strobe Mode is particularly well suited to memory systems latching data on the falling edge of WRITE.

$\overline{\text{WRL}}$ is provided for all 8-bit bus write cycles.

Address Valid Strobe Mode

If CCR bit 3 is a 0, then an Address Valid strobe is provided in the place of ALE (Figure 14). When the address valid mode is selected, $\overline{\text{ADV}}$ will go low after an external address is set up. It will stay low until the end of the bus cycle, where it will go inactive high. This can be used by ROM devices to provide a chip select for a single external RAM device in a minimum chip count system.



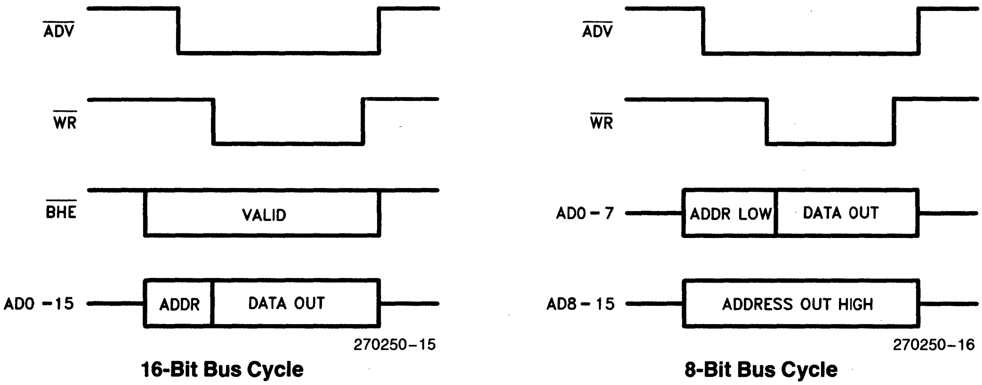
270250-13

16-Bit Bus Cycle

270250-14

8-Bit Bus Cycle

Figure 13. Write Strobe Mode



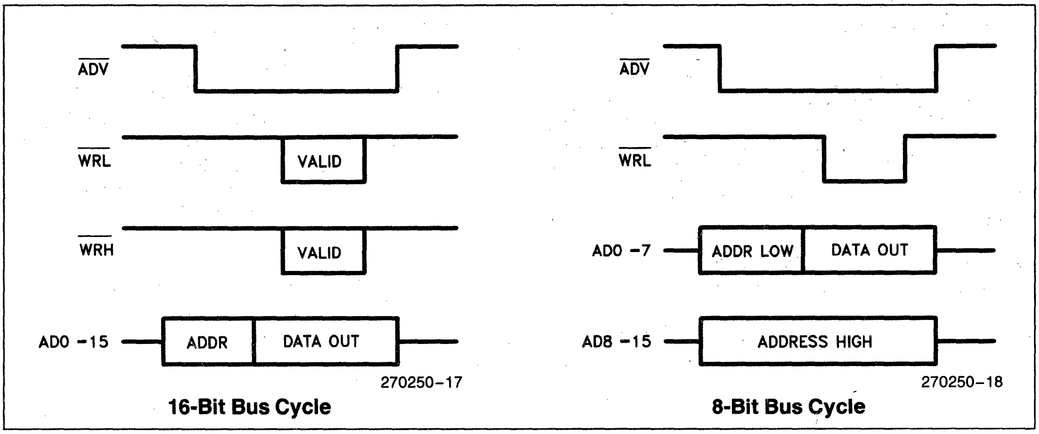
270250-15

16-Bit Bus Cycle

270250-16

8-Bit Bus Cycle

Figure 14. Address Valid Strobe Mode



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 15. Write Strobe with Address Valid Strobe

Address Valid with Write Strobe

If both CCR bits 2 and 3 are 0s, both the Address Valid strobe and the Write Strobes will be provided for bus control. Figure 15 shows these signals.

READY CONTROL

To simplify ready control, four modes of internal ready control logic have been provided. The modes are chosen by properly configuring bits 4 and 5 of the CCR.

The internal ready control logic can be used to limit the number of wait states that slow devices can insert into the bus cycle. When the READY pin is pulled low, wait states will be inserted into the bus cycle until the READY pin goes high, or the number of wait states equals the number specified by CCR bits 4 and 5, whichever comes first. Table 1 shows the number of wait states that can be selected. Internal Ready control can be disabled by loading 11 into bits 4 and 5 of the CCR.

Table 1. Internal Ready Control

IRC1	IRC0	Description
0	0	Limit to 1 Wait State
0	1	Limit to 2 Wait States
1	0	Limit to 3 Wait States
1	1	Disable Internal Ready Control

This feature provides for simple ready control. For example, every slow memory chip select line could be ORed together and be connected to the READY pin with CCR bits 4 and 5 programmed to give the desired number of wait states to the slow devices.

ROM/EPROM LOCK

Four modes of program memory lock are available on the 839XBH and 879XBH devices. CCR bits 6 and 7 (LOC0, LOC1) select whether internal program memory can be read (or written in EPROM devices) by a program executing from external memory. The modes are shown in Table 2. Internal ROM/EPROM addresses 2020H through 3FFFH are protected from reads while 2000H through 3FFFH are protected from writes, as set by the CCR.

Table 2. Program Lock Modes

LOC1	LOC0	Protection
0	0	Read and Write Protected
0	1	Read Protected
1	0	Write Protected
1	1	No Protection

Only code executing from internal memory can read protected internal memory, while a write protected memory can not be written to, even from internal execution. As a result of 8096BH prefetching of instructions, however, accesses to protected memory are not allowed for instructions located above 3FFAH. This is because the lock protection mechanism is gated off of the Memory Controller's slave program counter and not the CPU program counter. If the bus controller receives a request to perform a read of protected memory, the read sequence occurs with indeterminate data being returned to the CPU. Note that the interrupt vectors and the CCR are not protected.

To provide verification and testing when the program lock feature is enabled, the 839XBH and 879XBH verify the security key before programming or test modes are allowed to read from protected memory. Before protected memory can be read, the chip reads external memory locations 4020H through 402FH and compares the values found to the internal security key located from 2020H through 202FH. Only when the values exactly match will accesses to protected memory be allowed. The details of ROM/EPROM accessing are discussed in Section 10 of the Hardware Design chapter.

3.0 SOFTWARE OVERVIEW

This section provides information on writing programs to execute in the 8096BH. Additional information can be found in the following documents:

MCS®-96 MACRO ASSEMBLER USER'S GUIDE
Order Number 186 ASM 96 (Intel Systems)
Order Number D86 ASM 96NL (DOS Systems)

C-96 USER'S GUIDE
Order Number D86 C96NL (DOS Systems)

PL/M-96 USER'S GUIDE
Order Number 186 PLM 96 (Intel Systems)
Order Number D86 PLM 96NL (DOS Systems)

Throughout this section, short sections of code are used to illustrate the operation of the device. For these sections it has been assumed that a set of temporary registers have been predeclared. The names of these registers have been chosen as follows:

AX, BX, CX, and DX are 16-bit registers.

AL is the low byte of AX, AH is the high byte.

BL is the low byte of BX

CL is the low byte of CX

DL is the low byte of DX

These are the same as the names for the general data registers used in the 8086 (80186). It is important to note, however, that in the 8096BH, these are not dedicated registers but merely the symbolic names assigned by the programmer to an eight byte region within the onboard register file.

3.1 Operand Types

The MCS®-96 architecture provides support for a variety of data types which are likely to be useful in a control application. In the discussion of these operand types that follows, the names adopted by the PLM-96 programming language will be used where appropriate. To avoid confusion, the name of an operand type will be capitalized. A "BYTE" is an unsigned eight bit variable; a "byte" is an eight bit unit of data of any type.

BYTES

BYTES are unsigned 8-bit variables which can take on the values between 0 and 255. Arithmetic and relational operators can be applied to BYTE operands but the result must be interpreted in modulo 256 arithmetic. Logical operations on BYTES are applied bitwise. Bits within BYTES are labeled from 0 to 7, with 0 being the least significant bit. There are no alignment restrictions for BYTES, so they may be placed anywhere in the MCS-96 address space.

WORDS

WORDS are unsigned 16-bit variables which can take on the values between 0 and 65535. Arithmetic and relational operators can be applied to WORD operands but the result must be interpreted modulo 65536. Logical operations on WORDS are applied bitwise. Bits within words are labeled from 0 to 15 with 0 being the least significant bit. WORDS must be aligned at even byte boundaries in the MCS-96 address space. The least significant byte of the WORD is in the even byte address and the most significant byte is in the next higher (odd) address. The address of a word is the address of its least significant byte. Word operations to odd addresses are not guaranteed to operate in a consistent manner.

SHORT-INTEGERS

SHORT-INTEGERS are 8-bit signed variables which can take on the values between −128 and +127. Arithmetic operations which generate results outside of the range of a SHORT-INTEGER will set the overflow indicators in the program status word. The actual numeric result returned will be the same as the equivalent operation on BYTE variables. There are no alignment restrictions on SHORT-INTEGERS so they may be placed anywhere in the MCS-96 address space.

INTEGERS

INTEGERS are 16-bit signed variables which can take on the values between −32,768 and 32,767. Arithmetic operations which generate results outside of the range of an INTEGER will set the overflow indicators in the program status word. The actual numeric result returned will be the same as the equivalent operation on WORD variables. INTEGERS conform to the same alignment and addressing rules as do WORDS.

BITS

BITS are single-bit operands which can take on the Boolean values of true and false. In addition to the normal support for bits as components of BYTE and WORD operands, the 8096 provides for the direct testing of any bit in the internal register file. The MCS-96 architecture requires that bits be addressed as components of BYTES or WORDS, it does not support the direct addressing of bits that can occur in the MCS-51 architecture.

DOUBLE-WORDS

DOUBLE-WORDS are unsigned 32-bit variables which can take on the values between 0 and 4,294,967,295. The MCS-96 architecture provides direct support for this operand type only for shifts and as the dividend in a 32 by 16 divide and the product of a 16 by 16 multiply. For these operations a DOUBLE-WORD variable must reside in the on-board register file of the 8096 and be aligned at an address which is evenly divisible by 4. A DOUBLE-WORD operand is addressed by the address of its least significant byte. DOUBLE-WORD operations which are not directly supported can be easily implemented with two WORD operations. For consistency with Intel provided software the user should adopt the conventions for addressing DOUBLE-WORD operands which are discussed in Section 3.5.

LONG-INTEGERS

LONG-INTEGERS are 32-bit signed variables which can take on the values between −2,147,483,648 and 2,147,483,647. The MCS-96 architecture provides direct support for this data type only for shifts and as the dividend in a 32 by 16 divide and the product of a 16 by 16 multiply.

LONG-INTEGERS can also be normalized. For these operations a LONG-INTEGER variable must reside in the onboard register file of the 8096 and be aligned at an address which is evenly divisible by 4. A LONG-INTEGER is addressed by the address of its least significant byte.

LONG-INTEGER operations which are not directly supported can be easily implemented with two INTEGER operations. For consistency with Intel provided software, the user should adopt the conventions for addressing LONG operands which are discussed in Section 3.5.

3.2 Operand Addressing

Operands are accessed within the address space of the 8096 with one of six basic addressing modes. Some of the details of how these addressing modes work are hidden by the assembly language. If the programmer is to take full advantage of the architecture, it is important that these details be understood. This section will describe the addressing modes as they are handled by the hardware. At the end of this section the addressing modes will be described as they are seen through the assembly language. The six basic address modes which will be described are termed register-direct, indirect, indirect with auto-increment, immediate, short-indexed, and long-indexed. Several other useful addressing operations can be achieved by combining these basic addressing modes with specific registers such as the ZERO register or the stack pointer.

REGISTER-DIRECT REFERENCES

The register-direct mode is used to directly access a register from the 256 byte on-board register file. The register is selected by an 8-bit field within the instruction and register address and must conform to the alignment rules for the operand type. Depending on the instruction, up to three registers can take part in the calculation.

Examples

```
ADD     AX,BX,CX      ; AX:=BX+CX
MUL     AX,BX         ; AX:=AX*BX
INCB    CL            ; CL:=CL+1
```

INDIRECT REFERENCES

The indirect mode is used to access an operand by placing its address in a WORD variable in the register file. The calculated address must conform to the alignment rules for the operand type. Note that the indirect address can refer to an operand anywhere within the address space of the 8096, including the register file. The register which contains the indirect address is selected by an eight bit field within the instruction. An instruction can contain only one indirect reference and the remaining operands of the instruction (if any) must be register-direct references.

Examples

```
LD      AX,[AX]      ; AX:=MEM_WORD(AX)
ADDB    AL,BL,[CX]   ; AL:=BL+MEM_BYTE(CX)
POP     [AX]         ; MEM_WORD(AX):=MEM_WORD(SP) ; SP:=SP+2
```

INDIRECT WITH AUTO-INCREMENT REFERENCES

This addressing mode is the same as the indirect mode except that the WORD variable which contains the indirect address is incremented *after* it is used to address the operand. If the instruction operates on BYTES or SHORT-INTEGERS the indirect address variable will be incremented by one, if the instruction operates on WORDS or INTEGERS the indirect address variable will be incremented by two.

Examples

```
LD     AX,[BX]+     ; AX:=MEM_WORD(BX) ; BX:=BX+2
ADDB   AL,BL,[CX]+  ; AL:=BL+MEM_BYTE(CX) ; CX:=CX+1
PUSH   [AX]+        ; SP:=SP-2;
                    ;   MEM_WORD(SP):=MEM_WORD(AX)
                    ;   AX:=AX+2
```

IMMEDIATE REFERENCES

This addressing mode allows an operand to be taken directly from a field in the instruction. For operations on BYTE or SHORT-INTEGER operands this field is eight bits wide, for operations on WORD or INTEGER operands the field is 16 bits wide. An instruction can contain only one immediate reference and the remaining operand(s) must be register-direct references.

Examples

```
ADD  AX,#340  ; AX:=AX+340
PUSH #1234H   ; SP:=SP-2; MEM_WORD(SP):=1234H
DIVB AX,#10   ; AL:=AX/10; AH:=AX MOD 10
```

SHORT-INDEXED REFERENCES

In this addressing mode an eight bit field in the instruction selects a WORD variable in the register file which is assumed to contain an address. A second eight bit field in the instruction stream is sign-extended and summed with the WORD variable to form the address of the operand which will take part in the calculation. Since the eight bit field is sign-extended, the effective address can be up to 128 bytes before the address in the WORD variable and up to 127 bytes after it. An instruction can contain only one short-indexed reference and the remaining operand(s) must be register-direct references.

Examples

```
LD     AX,12[BX]     ; AX:=MEM_WORD(BX+12)
MULB   AX,BL,3[CX]   ; AX:=BL*MEM_BYTE(CX+3)
```

LONG-INDEXED REFERENCES

This addressing mode is like the short-indexed mode except that a *16-bit* field is taken from the instruction and added to the WORD variable to form the address of the operand. No sign extension is necessary. An instruction can contain only one long-indexed reference and the remaining operand(s) must be register-direct references.

Examples

```
AND  AX,BX,TABLE[CX]     ; AX:=BX AND MEM_WORD(TABLE+CX)
ST   AX,TABLE[BX]        ; MEM_WORD(TABLE+BX):=AX
ADDB AL,BL,LOOKUP[CX]    ; AL:=BL+MEM_BYTE(LOOKUP+CX)
```

ZERO REGISTER ADDRESSING

The first two bytes in the register file are fixed at zero by the 8096 hardware. In addition to providing a fixed source of the constant zero for calculations and comparisons, this register can be used as the WORD variable in a long-indexed reference. This combination of register selection and address mode allows any location in memory to be addressed directly.

Examples

```
ADD  AX,1234[0]   ; AX:=AX+MEM_WORD(1234)
POP  5678[0]      ; MEM_WORD(5678):=MEM_WORD(SP)
                  ;  SP:=SP+2
```

STACK POINTER REGISTER ADDRESSING

The system stack pointer in the 8096BH can be accessed as register 18H of the internal register file. In addition to providing for convenient manipulation of the stack pointer, this also facilitates the accessing of operands in the stack. The top of the stack, for example, can be accessed by using the stack pointer as the WORD variable in an indirect reference. In a similar fashion, the stack pointer can be used in the short-indexed mode to access data within the stack.

Examples

```
PUSH  [SP]       ; DUPLICATE TOP_OF_STACK
LD    AX,2[SP]   ; AX:=NEXT_TO_TOP
```

ASSEMBLY LANGUAGE ADDRESSING MODES

The 8096BH assembly language simplifies the choice of addressing modes to be used in several respects:

Direct Addressing. The assembly language will choose between register-direct addressing and long-indexed with the ZERO register depending on where the operand is in memory. The user can simply refer to an operand by its symbolic name; if the operand is in the register file, a register-direct reference will be used, if the operand is elsewhere in memory, a long-indexed reference will be generated.

Indexed Addressing. The assembly language will choose between short and long indexing depending on the value of the index expression. If the value can be expressed in eight bits then short indexing will be used, if it cannot be expressed in eight bits then long indexing will be used.

The use of these features of the assembly language simplifies the programming task and should be used wherever possible.

3.3 Program Status Word

The program status word (PSW) is a collection of Boolean flags which retain information concerning the state of the user's program. The format of the PSW is shown in Figure 16. The information in the PSW can be broken down into two basic categories; interrupt control and condition flags. The PSW can be saved in the system stack with a single operation (PUSHF) and restored in a like manner (POPF).

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Z	N	V	VT	C	—	I	ST	<Interrupt Mask Reg>							

Figure 16. PSW Register

INTERRUPT FLAGS

The lower eight bits of the PSW are used to individually mask the various sources of interrupt to the 8096. A logical '1' in these bit positions enables the servicing of the corresponding interrupt. These mask bits can be accessed as an eight bit byte (INT_MASK—address 8) in the on-board register file. Bit 9 in the PSW is the global interrupt disable. If this bit is cleared then all interrupts will be locked out except for the Non Maskable Interrupt (NMI). Note that the various interrupts are collected in the INT_PENDING register even if they are locked out. Execution of the corresponding service routines will procede according to their priority when they become enabled. Further information on the interrupt structure of the 8096 can be found in Section 4.

CONDITION FLAGS

The remaining bits in the PSW are set as side effects of instruction execution and can be tested by the conditional jump instructions.

Z. The Z (Zero) flag is set to indicate that the operation generated a result equal to zero. For the add-with-carry (ADDC) and subtract-with-borrow (SUBC) operations the Z flag is cleared if the result is non-zero but is never set. These two instructions are normally used in conjunction with the ADD and SUB instructions to perform multiple precision arithmetic. The operation of the Z flag for these instructions leaves it indicating the proper result for the entire multiple precision calculation.

N. The N (Negative) flag is set to indicate that the operation generated a negative result. Note that the N flag will be set to the algebraically correct state even if the calculation overflows.

V. The V (overflow) flag is set to indicate that the operation generated a result which is outside the range that can be expressed in the destination data type. For the SHL, SHLB and SHLL instructions, the V flag will be set if the most significant bit of the operand changes at any time during the shift.

VT. The VT (oVerflow Trap) flag is set whenever the V flag is set but can only be cleared by an instruction which explicitly operates on it such as the CLRVT or JVT instructions. The operation of the VT flag allows for the testing for a possible overflow condition at the end of a sequence of related arithmetic operations. This is normally more efficient than testing the V flag after each instruction.

C. The C (Carry) flag is set to indicate the state of the arithmetic carry from the most significant bit of the ALU for an arithmetic operation or the state of the last bit shifted out of the operand for a shift. Arithmetic Borrow after a subtract operation is the complement of the C flag (i.e. if the operation generated a borrow then C = 0).

ST. The ST (STicky bit) flag is set to indicate that during a right shift a 1 has been shifted first into the C flag and then been shifted out. The ST flag is undefined after a multiply operation. The ST flag can be used along with the C flag to control rounding after a right shift. Consider multiplying two eight bit quantities and then scaling the result down to 12 bits:

```
MULUB   AX,CL,DL    ;AX:=CL*DL
SHR     AX,#4       ;Shift right 4 places
```

If the C flag is set after the shift, it indicates that the bits shifted off the end of the operand were greater-than or equal-to one half the least significant bit (LSB) of the result. If the C flag is clear after the shift, it indicates that the bits shifted off the end of the operand were less than half the LSB of the result. Without the ST flag, the rounding decision must be made on the basis of this information alone. (Normally the result would be rounded up if the C flag is set.) The ST flag allows a finer resolution in the rounding decision:

C ST	Value of the Bits Shifted Off
0 0	Value = 0
0 1	0 < Value < ½ LSB
1 0	Value = ½ LSB
1 1	Value > ½ LSB

Figure 17. Rounding Alternatives

Imprecise rounding can be a major source of error in a numerical calculation; use of the ST flag improves the options available to the programmer.

3.4 Instruction Set

The MCS-96 instruction set contains a full set of arithmetic and logical operations for the 8-bit data types BYTE and SHORT INTEGER and for the 16-bit data types WORD and INTEGER. The DOUBLE-WORD and LONG data types (32 bits) are supported for the products of 16 by 16 multiplies and the dividends of 32 by 16 divides and for shift operations. The remaining operations on 32-bit variables can be implemented by combinations of 16-bit operations. As an example the sequence:

```
ADD     AX,CX
ADDC    BX,DX
```

performs a 32-bit addition, and the sequence

```
SUB     AX,CX
SUBC    BX,DX
```

performs a 32-bit subtraction. Operations on REAL (i.e. floating point) variables are not supported directly by the hardware but are supported by the floating point library for the 8096BH (FPAL-96) which implements a single precision subset of the proposed IEEE standard for floating point operations. The performance of this software is significantly improved by the 8096BH NORML instruction which normalizes a 32-bit variable and by the existence of the ST flag in the PSW.

In addition to the operations on the various data types, the 8096BH supports conversions between these types. LDBZE (load byte zero extended) converts a BYTE to a WORD and LDBSE (load byte sign extended) converts a SHORT-INTEGER into an INTEGER. WORDS can be converted to DOUBLE-WORDS by simply clearing the upper WORD of the DOUBLE-WORD (CLR) and INTEGERS can be converted to LONGS with the EXT (sign extend) instruction.

The MCS-96 instructions for addition, subtraction, and comparison do not distinguish between unsigned words and signed integers. Conditional jumps are provided to allow the user to treat the results of these operations as either signed or unsigned quantities. As an example, the CMPB (compare byte) instruction is used to compare both signed and unsigned eight bit quantities. A JH (jump if higher) could be used following the compare if unsigned operands were involved or a JGT (jump if greater-than) if signed operands were involved.

Table 3 summarizes the operation of each of the instructions. Complete descriptions of each instruction and its timings can be found in the Instruction Set chapter. A summary of instruction opcodes and timing is included in the quick reference section at the end of this chapter.

Table 3. Instruction Summary

Mnemonic	Oper-ands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
ADD/ADDB	2	D ← D + A	✓	✓	✓	✓	↑	—	
ADD/ADDB	3	D ← B + A	✓	✓	✓	✓	↑	—	
ADDC/ADDCB	2	D ← D + A + C	↓	✓	✓	✓	↑	—	
SUB/SUBB	2	D ← D − A	✓	✓	✓	✓	↑	—	
SUB/SUBB	3	D ← B − A	✓	✓	✓	✓	↑	—	
SUBC/SUBCB	2	D ← D − A + C − 1	↓	✓	✓	✓	↑	—	
CMP/CMPB	2	D − A	✓	✓	✓	✓	↑	—	
MUL/MULU	2	D, D + 2 ← D * A	—	—	—	—	—	?	2
MUL/MULU	3	D, D + 2 ← B * A	—	—	—	—	—	?	2
MULB/MULUB	2	D, D + 1 ← D * A	—	—	—	—	—	?	3
MULB/MULUB	3	D, D + 1 ← B * A	—	—	—	—	—	?	3
DIVU	2	D ← (D, D + 2)/A, D + 2 ← remainder	—	—	—	✓	↑	—	2
DIVUB	2	D ← (D, D + 1)/A, D + 1 ← remainder	—	—	—	✓	↑	—	3
DIV	2	D ← (D, D + 2)/A, D + 2 ← remainder	—	—	—	?	↑	—	
DIVB	2	D ← (D, D + 1)/A, D + 1 ← remainder	—	—	—	?	↑	—	
AND/ANDB	2	D ← D and A	✓	✓	0	0	—	—	
AND/ANDB	3	D ← B and A	✓	✓	0	0	—	—	
OR/ORB	2	D ← D or A	✓	✓	0	0	—	—	
XOR/XORB	2	D ← D (excl. or) A	✓	✓	0	0	—	—	
LD/LDB	2	D ← A	—	—	—	—	—	—	
ST/STB	2	A ← D	—	—	—	—	—	—	
LDBSE	2	D ← A; D + 1 ← SIGN(A)	—	—	—	—	—	—	3, 4
LDBZE	2	D ← A; D + 1 ← 0	—	—	—	—	—	—	3, 4
PUSH	1	SP ← SP − 2; (SP) ← A	—	—	—	—	—	—	
POP	1	A ← (SP); SP ← SP + 2	—	—	—	—	—	—	
PUSHF	0	SP ← SP − 2; (SP) ← PSW; PSW ← 0000H I ← 0	0	0	0	0	0	0	
POPF	0	PSW ← (SP); SP ← SP + 2; I ← ✓	✓	✓	✓	✓	✓	✓	
SJMP	1	PC ← PC + 11-bit offset	—	—	—	—	—	—	5
LJMP	1	PC ← PC + 16-bit offset	—	—	—	—	—	—	5
BR [indirect]	1	PC ← (A)	—	—	—	—	—	—	
SCALL	1	SP ← SP − 2; (SP) ← PC; PC ← PC + 11-bit offset	—	—	—	—	—	—	5
LCALL	1	SP ← SP − 2; (SP) ← PC; PC ← PC + 16-bit offset	—	—	—	—	—	—	5
RET	0	PC ← (SP); SP ← SP + 2	—	—	—	—	—	—	
J (conditional)	1	PC ← PC + 8-bit offset (if taken)	—	—	—	—	—	—	5
JC	1	Jump if C = 1	—	—	—	—	—	—	5
JNC	1	Jump if C = 0	—	—	—	—	—	—	5
JE	1	Jump if Z = 1	—	—	—	—	—	—	5

NOTES:

1. If the mnemonic ends in "B", a byte operation is performed, otherwise a word operation is done. Operands D, B, and A must conform to the alignment rules for the required operand type. D and B are locations in the register file; A can be located anywhere in memory.
2. D, D + 2 are consecutive WORDS in memory; D is DOUBLE-WORD aligned.
3. D, D + 1 are consecutive BYTES in memory; D is WORD aligned.
4. Changes a byte to a word.
5. Offset is a 2's complement number.

Table 3. Instruction Summary (Continued)

Mnemonic	Oper-ands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
JNE	1	Jump if Z = 0	—	—	—	—	—	—	5
JGE	1	Jump if N = 0	—	—	—	—	—	—	5
JLT	1	Jump if N = 1	—	—	—	—	—	—	5
JGT	1	Jump if N = 0 and Z = 0	—	—	—	—	—	—	5
JLE	1	Jump if N = 1 or Z = 1	—	—	—	—	—	—	5
JH	1	Jump if C = 1 and Z = 0	—	—	—	—	—	—	5
JNH	1	Jump if C = 0 or Z = 1	—	—	—	—	—	—	5
JV	1	Jump if V = 1	—	—	—	—	—	—	5
JNV	1	Jump if V = 0	—	—	—	—	—	—	5
JVT	1	Jump if VT = 1; Clear VT	—	—	—	—	0	—	5
JNVT	1	Jump if VT = 0; Clear VT	—	—	—	—	0	—	5
JST	1	Jump if ST = 1	—	—	—	—	—	—	5
JNST	1	Jump if ST = 0	—	—	—	—	—	—	5
JBS	3	Jump if Specified Bit = 1	—	—	—	—	—	—	5, 6
JBC	3	Jump if Specified Bit = 0	—	—	—	—	—	—	5, 6
DJNZ	1	D ← D − 1; if D ≠ 0 then PC ← PC + 8-bit offset	—	—	—	—	—	—	5
DEC/DECB	1	D ← D − 1	✓	✓	✓	✓	↑	—	
NEG/NEGB	1	D ← 0 − D	✓	✓	✓	✓	↑	—	
INC/INCB	1	D ← D + 1	✓	✓	✓	✓	↑	—	
EXT	1	D ← D; D + 2 ← Sign (D)	✓	✓	0	0	—	—	2
EXTB	1	D ← D; D + 1 ← Sign(D)	✓	✓	0	0	—	—	3
NOT/NOTB	1	D ← Logical Not (D)	✓	✓	0	0	—	—	
CLR/CLRB	1	D ← 0	1	0	0	0	—	—	
SHL/SHLB/SHLL	2	C ← msb — — — — — — lsb ← 0	✓	?	✓	✓	↑	—	7
SHR/SHRB/SHRL	2	0 → msb — — — — — — lsb → C	✓	?	✓	0	—	✓	7
SHRA/SHRAB/SHRAL	2	msb → msb — — — — — — lsb → C	✓	✓	✓	0	—	✓	7
SETC	0	C ← 1	—	—	1	—	—	—	
CLRC	0	C ← 0	—	—	0	—	—	—	
CLRVT	0	VT ← 0	—	—	—	—	0	—	
RST	0	PC ← 2080H	0	0	0	0	0	0	8
DI	0	Disable All Interrupts (I ← 0)	—	—	—	—	—	—	
EI	0	Enable All Interrupts (I ← 1)	—	—	—	—	—	—	
NOP	0	PC ← PC + 1	—	—	—	—	—	—	
SKIP	0	PC ← PC + 2	—	—	—	—	—	—	
NORML	2	Left shift till msb = 1; D ← shift count	✓	?	0	—	—	—	7
TRAP	0	SP ← SP − 2; (SP) ← PC PC ← (2010H)	—	—	—	—	—	—	9

NOTES:

1. If the mnemonic ends in "B", a byte operation is performed, otherwise a word operation is done. Operands D, B and A must conform to the alignment rules for the required operand type. D and B are locations in the register file; A can be located anywhere in memory.
5. Offset is a 2's complement number.
6. Specified bit is one of the 2048 bits in the register file.
7. The "L" (Long) suffix indicates double-word operation.
8. Initiates a Reset by pulling RESET low. Software should re-initialize all the necessary registers with code starting at 2080H.
9. The assembler will not accept this mnemonic.

3.5 Software Standards and Conventions

For a software project of any size it is a good idea to modularize the program and to establish standards which control the communication between these modules. The nature of these standards will vary with the needs of the final application. A common component of all of these standards, however, must be the mechanism for passing parameters to procedures and returning results from procedures. In the absence of some overriding consideration which prevents their use, it is suggested that the user conform to the conventions adopted by the PLM-96 programming language for procedure linkage. It is a very usable standard for both the assembly language and PLM-96 environment and it offers compatibility between these environments. Another advantage is that it allows the user access to the same floating point arithmetics library that PLM-96 uses to operate on REAL variables.

REGISTER UTILIZATION

The MCS-96 architecture provides a 256 byte register file. Some of these registers are used to control register-mapped I/O devices and for other special functions such as the ZERO register and the stack pointer. The remaining bytes in the register file, some 230 of them, are available for allocation by the programmer. If these registers are to be used effectively, some overall strategy for their allocation must be adopted. PLM-96 adopts the simple and effective strategy of allocating the eight bytes between addresses 1CH and 23H as temporary storage. The starting address of this region is called PLMREG. The remaining area in the register file is treated as a segment of memory which is allocated as required.

ADDRESSING 32-BIT OPERANDS

These operands are formed from two adjacent 16-bit words in memory. The least significant word of the double word is always in lower address, even when the data is in the stack (which means that the most significant word must be pushed into the stack first). A double word is addressed by the address of its least significant byte. Note that the hardware supports some operations on double words (e.g. normalize and divide). For these operations the double word must be in the internal register file and must have an address which is evenly divisible by four.

SUBROUTINE LINKAGE

Parameters are passed to subroutines in the stack. Parameters are pushed into the stack in the order that they are encountered in the scanning of the source text. Eight-bit parameters (BYTES or SHORT-INTEGERS) are pushed into the stack with the high order byte undefined. Thirty-two bit parameters (LONG-INTEGERS, DOUBLE-WORDS, and REALS) are pushed into the stack as two 16-bit values; the most significant half of the parameter is pushed into the stack first.

As an example, consider the following PLM-96 procedure:

```
example_procedure: PROCEDURE
(param1,param2,param3);
  DECLARE param1 BYTE,
          param2 DWORD,
          param3 WORD;
```

When this procedure is entered at run time the stack will contain the parameters in the following order:

?????? : param1	
high word of param2	
low word of param2	
param3	
return address	← Stack_pointer

Figure 18. Stack Image

If a procedure returns a value to the calling code (as opposed to modifying more global variables) then the result is returned in the variable PLMREG. PLMREG is viewed as either an 8-, 16- or 32-bit variable depending on the type of the procedure.

The standard calling convention adopted by PLM-96 has several key features:

a) Procedures can always assume that the eight bytes of register file memory starting at PLMREG can be used as temporaries within the body of the procedure.

b) Code which calls a procedure must assume that the eight bytes of register file memory starting at PLMREG are modified by the procedure.

c) The Program Status Word (PSW—see Section 3.3) is not saved and restored by procedures so the calling code must assume that the condition flags (Z, N, V, VT, C, and ST) are modified by the procedure.

d) Function results from procedures are always returned in the variable PLMREG.

PLM-96 allows the definition of INTERRUPT procedures which are executed when a predefined interrupt occurs. These procedures do not conform to the rules of a normal procedure. Parameters cannot be passed to these procedures and they cannot return results. Since they can execute essentially at any time (hence the term interrupt), these procedures must save the PSW and PLMREG when they are entered and restore these values before they exit.

4.0 INTERRUPT STRUCTURE

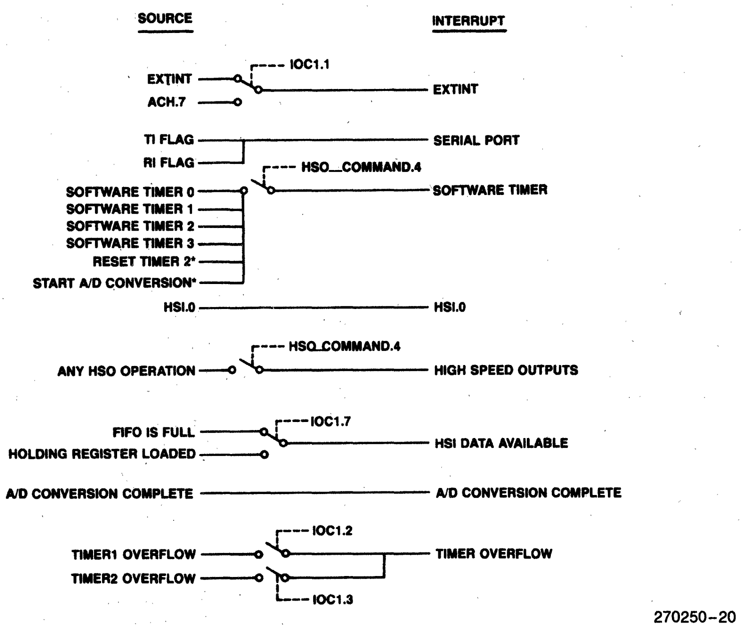
There are 21 sources of interrupts on the 8096BH. These sources are gathered into 8 interrupt types as indicated in Figure 19. The I/O control registers which control some of the sources are indicated in the figure. Each of the eight types of interrupts has its own interrupt vector as listed in Figure 20. In addition to the 8 standard interrupts, there is a TRAP instruction which acts as a software generated interrupt. This instruction is not currently supported by the MCS-96 Assembler and is reserved for use in Intel development systems.

The programmer must initialize the interrupt vector table with the starting address of the appropriate interrupt service routine. It is suggested that any unused interrupts be vectored to an error handling routine. The error routine should contain recovery code that will not further corrupt an already erroneous situation. In a debug environment, it may be desirable to have the routine lock into a jump to self loop which would be easily traceable with emulation tools. More sophisticated routines may be appropriate for production code recoveries.

Three registers control the operation of the interrupt system: Interrupt Pending, Interrupt Mask, and the PSW which contains a global disable bit. A block diagram of the system is shown in Figure 21. The transition detector looks for 0 to 1 transitions on any of the sources. External sources have a maximum transition speed of one edge every state time. If this is exceeded the interrupt may not be detected.

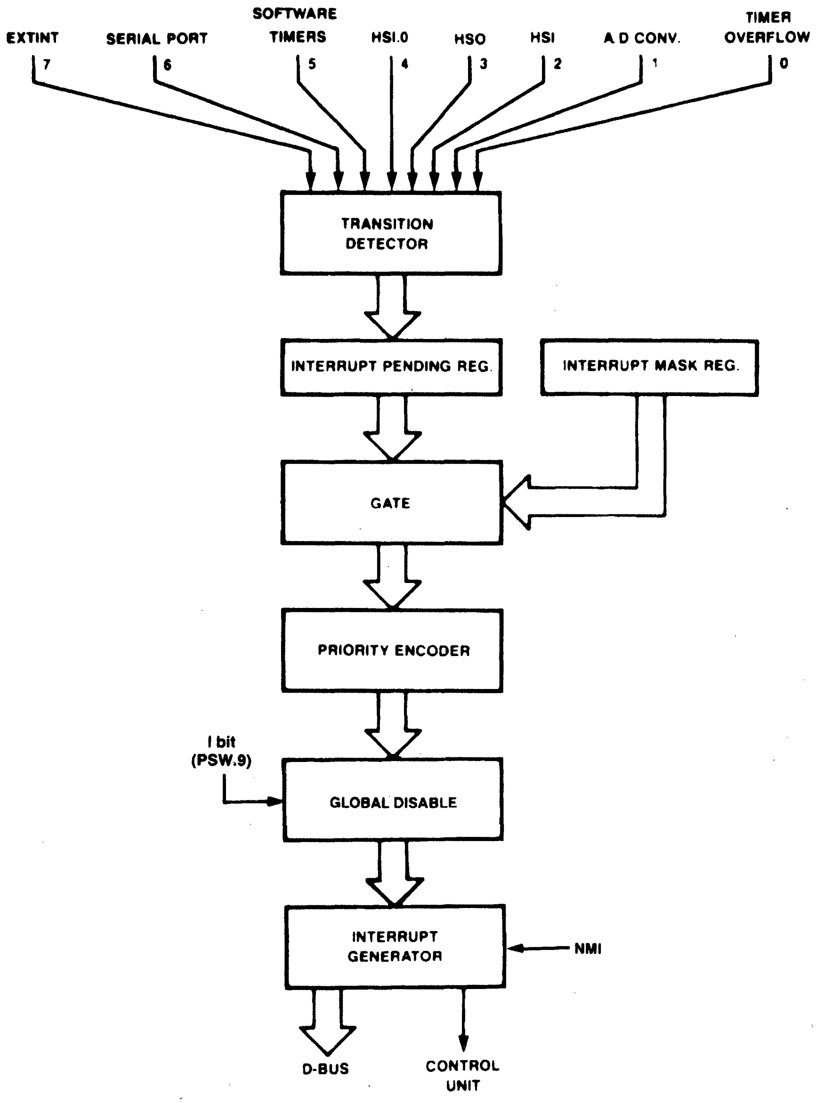
Vector	Vector Location		Priority
	(High Byte)	(Low Byte)	
Software Trap	2011H	2010H	Not Applicable
Extint	200FH	200EH	7 (Highest)
Serial Port	200DH	200CH	6
Software Timers	200BH	200AH	5
HSI.0	2009H	2008H	4
High Speed Outputs	2007H	2006H	3
HSI Data Available	2005H	2004H	2
A/D Conversion Complete	2003H	2002H	1
Timer Overflow	2001H	2000H	0 (Lowest)

Figure 20. Interrupt Vector Locations



NOTE:
*Only when initiated by the HSO unit.

Figure 19. All Possible Interrupt Sources



270250–21

Figure 21. Block Diagram of Interrupt System

4.1 Interrupt Control

Interrupt Pending Register

When the hardware detects one of the eight interrupts it sets the corresponding bit in the pending interrupt register (INT_PENDING-09H). When the interrupt vector is taken, the pending bit is cleared. This register, the format of which is shown in Figure 22, can be read or modified as a byte register. It can be read to determine which of the interrupts are pending at any given time or modified to either clear pending interrupts or generate interrupts under software control. Any software which modifies the INT_PENDING register should ensure that the entire operation is indivisible. The easiest way to do this is to use the logical instructions in the two or three operand format, for example:

```
ANDB  INT_PENDING,#11111101B
             ; Clears the A/D Interrupt
ORB   INT_PENDING,#00000010B
             ; Sets the A/D Interrupt
```

Caution must be used when writing to the pending register to clear interrupts. If the interrupt has already been acknowledged when the bit is cleared, a 4 state time "partial" interrupt cycle will occur. This is because the 8096BH will have to fetch the next instruction of the normal instruction flow, instead of proceeding with the interrupt processing as it was going to. The effect on the program will be essentially that of an extra NOP. This can be prevented by clearing the bits using a 2 operand immediate logical, as the 8096BH holds off acknowledging interrupts during these "read/modify/write" instructions.

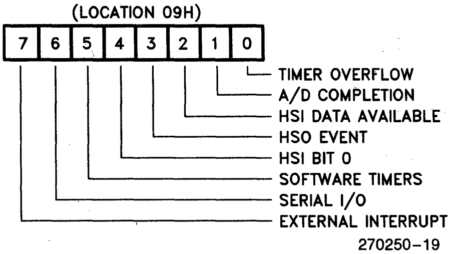


Figure 22. Interrupt Pending Register

Interrupt Mask Register

Individual interrupts can be enabled or disabled by setting or clearing bits in the interrupt mask register (INT_MASK-08H). The format of this register is the same as that of the Interrupt Pending Register shown in Figure 22.

The INT_MASK register can be read or written as byte register. A one in any bit position will enable the corresponding interrupt source and a zero will disable the source. The hardware will save any interrupts that occur by setting bits in the pending register, even if the interrupt mask bit is cleared. The INT_MASK register also can be accessed as the lower eight bits of the PSW so the PUSHF and POPF instructions save and restore the INT_MASK register as well as the global interrupt lockout and the arithmetic flags.

GLOBAL DISABLE

The processing of all interrupts can be disabled by clearing the I bit in the PSW. Setting the I bit will enable interrupts that have mask register bits which are set. The I bit is controlled by the EI (Enable Interrupts) and DI (Disable Interrupts) instructions. Note that the I bit only controls the actual servicing of interrupts. Interrupts that occur during periods of lockout will be held in the pending register and serviced on a prioritized basis when the lockout period ends.

4.2 Interrupt Priorities

The priority encoder looks at all of the interrupts which are both pending and enabled, and selects the one with the highest priority. The priorities are shown in Figure 20 (7 is highest, 0 is lowest). The interrupt generator then forces a call to the location in the indicated vector location. This location would be the starting location of the Interrupt Service Routine (ISR).

This priority selection controls the order in which pending interrupts are passed to the software via interrupt calls. The software can then implement its own priority structure by controlling the mask register (INT_MASK). To see how this is done, consider the case of a serial I/O service routine which must run at a priority level which is lower than the HSI data available interrupt but higher than any other source. The "preamble" and exit code for this interrupt service routine would look like this:

```
serial_io_isr:
 PUSHF           ; Save the PSW
                  (Includes INT_MASK)
 LDB    INT_MASK,#00000100B
 EI              ; Enable interrupts again
 ;  ⎫
 ;  ⎪
 ;  ⎪
 ;  ⎬    Service the interrupt
 ;  ⎪
 ;  ⎪
 ;  ⎭
 POPF            ; Restore the PSW
 RET
```

Note that location 200CH in the interrupt vector table would have to be loaded with the value of the label serial_io_isr and the interrupt be enabled for this routine to execute.

There is an interesting chain of instruction side-effects which makes this (or any other) 8096 interrupt service routine execute properly:

a) After the hardware decides to process an interrupt, it generates and executes a special interrupt-call instruction, which pushes the current program counter onto the stack and then loads the program counter with the contents of the vector table entry corresponding to the interrupt. The hardware will not allow another interrupt to be serviced immediately following the interrupt-call. This guarantees that once the interrupt-call starts, the first instruction of the interrupt service routine will execute.

b) The PUSHF instruction, which is now guaranteed to execute, saves the PSW in the stack and then clears the PSW. The PSW contains, in addition to the arithmetic flags, the INT_MASK register and the global disable flag (I). The hardware will not allow an interrupt following a PUSHF instruction and, by the time the LD instruction starts, all of the interrupt enable flags will be cleared. Now there is guaranteed execution of the LD INT_MASK instruction.

c) The LD INT_MASK instruction enables those interrupts that the programmer chooses to allow to interrupt the serial I/O interrupt service routine. In this example only the HSI data available interrupt will be allowed to do this but any interrupt or combination of interrupts could be enabled at this point, even the serial interrupt. It is the loading of the INT_MASK register which allows the software to establish its own priorities for interrupt servicing independently from those that the hardware enforces.

d) The EI instruction reenables the processing of interrupts.

e) The actual interrupt service routine executes within the priority structure established by the software.

f) At the end of the service routine the POPF instruction restores the PSW to its state when the interrupt-call occurred. The hardware will not allow interrupts to be processed following a POPF instruction so the execution of the last instruction (RET) is guaranteed before further interrupts can occur. The reason that this RET instruction must be protected in this fashion is that it is quite likely that the POPF instruction will reenable an interrupt which is already pending. If this interrupt were serviced before the RET instruction, then the return address to the code that was executing when the original interrupt occurred would be left on the stack. While this does not present a problem to the program flow, it could result in a stack overflow if interrupts are occurring at a high frequency. The POPF instruction also pops the INT_MASK register (part of the PSW), so any changes made to this register during a routine which ends with a POPF will be lost.

Notice that the "preamble" and exit code for the interrupt service routine does not include any code for saving or restoring registers. This is because it has been assumed that the interrupt service routine has been allocated its own private set of registers from the onboard register file. The availability of some 230 bytes of register storage makes this quite practical.

4.3 Critical Regions

Interrupt service routines must share some data with other routines. Whenever the programmer is coding those sections of code which access these shared pieces of data, great care must be taken to ensure that the integrity of the data is maintained. Consider clearing a bit in the interrupt pending register as part of a non-interrupt routine:

```
LDB       AL,INT_PENDING
ANDB      AL,#bit_mask
STB       AL,INT_PENDING
```

This code works if no other routines are operating concurrently, but will cause occasional but serious problems if used in a concurrent environment. (All programs which make use of interrupts must be considered to be part of a concurrent environment.) To demonstrate this problem, assume that the INT_PENDING register contains 00001111B and bit 3 (HSO event interrupt pending) is to be reset. The code does work for this data pattern but what happens if an HSI interrupt occurs somewhere between the LDB and the STB instructions? Before the LDB instruction INT_PENDING contains 00001111B and after the LDB instruction so does AL. If the HSI interrupt service routine executes at this point then INT_PENDING will change to 00001011B. The ANDB changes AL to 00000111B and the STB changes INT_PENDING to 00000111B. It should be 00000011B. This code sequence has manged to generate a false HSI interrupt The same basic process can generate an amazing assortment of problems and headaches. These problems can be avoided by assuring mutual exclusion which basically means that if more than one routine can change a variable, then the programmer must ensure exclusive access to the variable during the entire operation on the variable.

In many cases the instruction set of the 8096 allows the variable to be modified with a single instruction. The code in the above example can be implemented with a single instruction.

```
ANDB        INT_PENDING,#bit_mask
```

Instructions are indivisible so mutual exclusion is ensured in this case. Changes to the INT_PENDING register must be made as a single instruction, since bits can be changed in this register even if interrupts are disabled. Depending on system configurations, several other SFRs might also need to be changed in a single instruction for the same reason.

When variables must be modified without interruption, and a single instruction can not be used, the programmer must create what is termed a critical region in which it is safe to modify the variable. One way to do this is to simply disable interrupts with a DI instruction, perform the modification, and then re-enable interrupts with an EI instruction. The problem with this approach is that it leaves the interrupts enabled even if they were not enabled at the start. A better solution is to enter the critical region with a PUSHF instruction which saves the PSW and also clears the interrupt enable flags. The region can then be terminated with a POPF instruction which returns the interrupt enable to the state it was in before the code sequence. It should be noted that some system configurations might require more protection to form a critical region. An example is a system in which more than one processor has access to a common resource such as memory or external I/O devices.

4.4 Interrupt Timing

Interrupts are not always acknowledged immediately. If the interrupt signal does not occur prior to 4 state-times before the end of an instruction, the interrupt will not be acknowledged until after the next instruction has been executed. This is because an instruction is fetched and prepared for execution a few state times before it is actually executed.

There are 6 instructions which always inhibit interrupts from being acknowledged until after the next instruction has been executed. These instructions are:

EI, DI	— Enable and Disable Interrupts
POPF, PUSHF	— Pop and Push Flags
SIGND	— Prefix to perform signed multiply and divide (Note that this is not an ASM-96 Mnemonic, but is used for signed multiply and divide)
SOFTWARE TRAP	— Software interrupt

When an interrupt is acknowledged, the interrupt pending bit is cleared, and a call is forced to the location indicated by the specified interrupt vector. This call occurs after the completion of the instruction in process, except as noted above. The procedure of getting the vector and forcing the call requires 21 state times. If the stack is in external RAM an additional 3 state times are required.

The maximum number of state times required from the time an interrupt is generated (not acknowledged) until the 8096 begins executing code at the desired location is the time of the longest instruction, NORML (Normalize — 42 state times), plus the 4 state times prior to the end of the previous instruction, plus the response time (21 to 24 state times). Therefore, the maximum response time is 70 (42 + 4 + 24) state times. This does not include the 12 state times required for PUSHF if it is used as the first instruction in the interrupt routine or additional latency caused by having the interrupt masked or disabled. Refer to Figure 22A, Interrupt Response Time, to visualize an example of a worst case scenario.

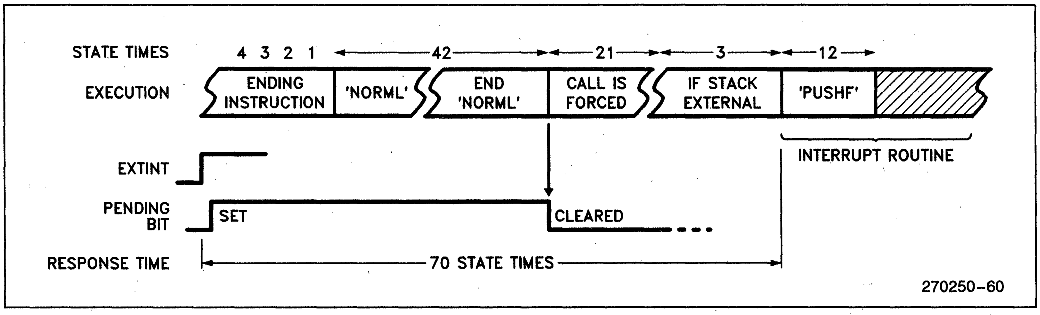


Figure 22A. Interrupt Response Time

Interrupt latency time can be reduced by careful selection of instructions in areas of code where interrupts are expected. Using 'EI' followed immediately by a long instruction (e.g. MUL, NORML, etc.) will increase the maximum latency by 4 state times, as an interrupt cannot occur between EI and the instruction following EI. The "DI", "PUSHF", "POPF" and "TRAP" instructions will also cause the same situation. Typically the PUSHF, POPF and TRAP instructions would only effect latency when one interrupt routine is already in process, as these instructions are seldom used at other times.

5.0 TIMERS

Two 16-bit timers are available for use on the 8096. The first is designated "Timer 1", the second, "Timer 2". Timer 1 is used to synchronize events to real time, while Timer 2 can be clocked externally and synchronizes events to external occurrences.

5.1 Timer 1

Timer 1 is clocked once every eight state times and can be cleared only by executing a reset. The only other way to change its value is by writing to 000CH but this is a test mode which sets both timers to 0FFFXH and should not be used in programs.

5.2 Timer 2

Timer 2 can be incremented by transitions (one count each transition, rising *and* falling) on either T2CLK or HSI.1. The multiple functionality of the timer is determined by the state of I/O Control Register 0, bit 7 (IOC0.7). To ensure that all CAM entries are checked each count of Timer 2, the maximum transition speed is limited to once per eight state times. Timer 2 can be cleared by: executing a reset, by setting IOC0.1, by triggering HSO channel 0EH, or by pulling T2RST or HSI.0 high. The HSO and CAM are described in Section 7 and 8. IOC0.3 and ICO0.5 control the resetting of Timer 2. Figure 23 shows the different ways of manipulating Timer 2. It is recommended that the IOC0 register only be used once during power on reset to initialize the timers and pins. This initialization will be followed by an HSO command 14 to clear Timer 2 internally; or externally cleared by the T2RST or HSI.0 pins.

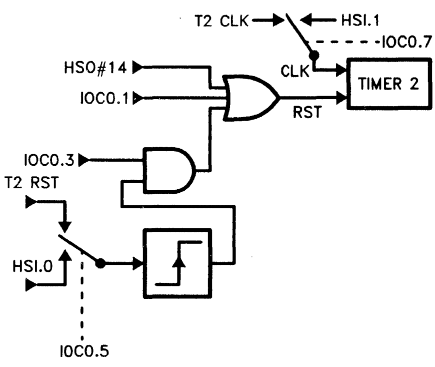


Figure 23. Timer 2 Clock and Reset Options

5.3 Timer Interrupts

Both Timer 1 and Timer 2 can be used to trigger a timer overflow interrupt and set a flag in the I/O Status Register 1 (IOS1). The interrupts are controlled by IOC1.2 and IOC1.3 respectively. The flags are set in IOS1.5 and IOS1.4, respectively.

Caution must be used when examining the flags, as any access (including Compare and Jump on Bit) of IOS1 clears bits 0 through 5 including the software timer flags. It is, therefore, recommended to write the byte to a temporary register before testing bits. The general enabling and disabling of the timer interrupts are controlled by the Interrupt Mask Register bit 0. In all cases, setting a bit enables a function, while clearing a bit disables it.

5.4 Timer Related Sections

The High Speed I/O unit is coupled to the timers in that the HSI records the value on Timer 1 when transitions occur and the HSO causes transitions to occur based on values of either Timer 1 or Timer 2. The baud rate generator can use the T2CLK pin as input to its counter. a complete listing of the functions of IOS1, IOC0, and IOC1 are in Section 11.

6.0 HIGH SPEED INPUTS

The High Speed Input Unit (HSI), can be used to record the time at which an event occurs with respect to Timer 1. There are 4 lines (HSI.0 through HSI.3) which can be used in this mode and up to a total of 8 events can be recorded. HSI.2 and HSI.3 are bidirectional pins which can also be used as HSO.4 and HSO.5. The I/O Control Registers (IOC0 and IOC1) are used to determine the functions of these pins. A block diagram of the HSI unit is shown in Figure 24.

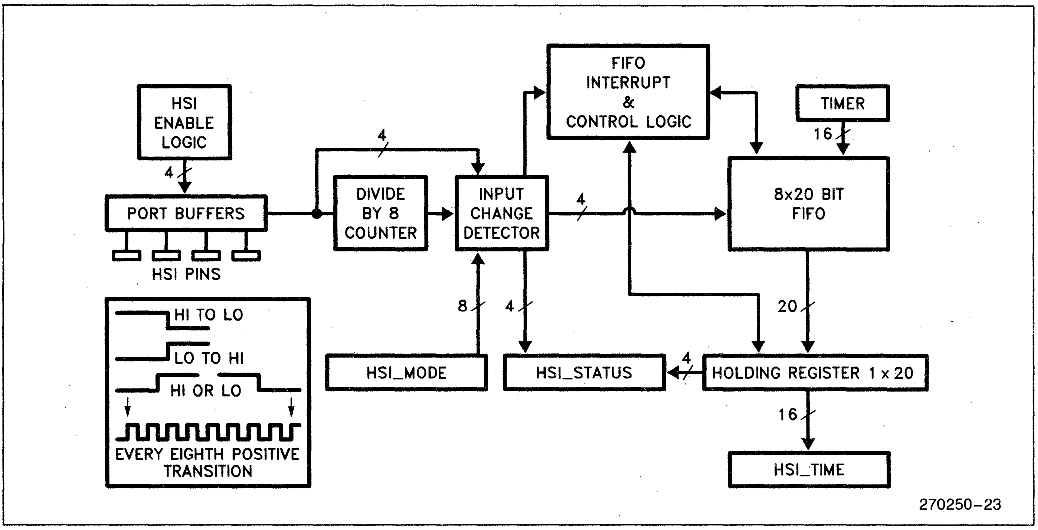


Figure 24. High Speed Input Unit

6.1 HSI Modes

There are 4 possible modes of operation for each of the HSI pins. The HSI mode register is used to control which pins will look for what type of events. The 8-bit register is set up as shown in Figure 25.

High and low levels each need to be held for at least 1 state time to ensure proper operation. The maximum input speed is 1 event every 8 state times except when the 8 transition mode is used, in which case it is 1 transition per state time. The divide by eight counter can only be zeroed in mid-count by performing a hardware reset on the 8096BH.

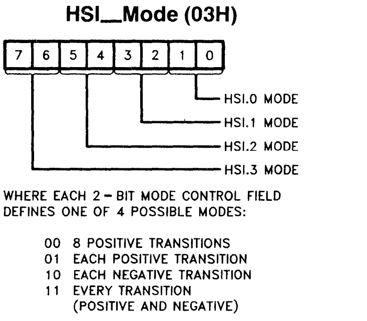

HSI_Mode (03H)
7
6
5
4
3
2
1
0
HSI.0 MODE
HSI.1 MODE
HSI.2 MODE
HSI.3 MODE
WHERE EACH 2-BIT MODE CONTROL FIELD
DEFINES ONE OF 4 POSSIBLE MODES:
00 8 POSITIVE TRANSITIONS
01 EACH POSITIVE TRANSITION
10 EACH NEGATIVE TRANSITION
11 EVERY TRANSITION
(POSITIVE AND NEGATIVE)
270250-24

Figure 25. HSI Mode Register Diagram

The HSI lines can be individually enabled and disabled using bits in IOC0, at location 0015H. Figure 26 shows the bit locations which control the HSI pins. If the pin is disabled, transitions will not be entered in the FIFO.

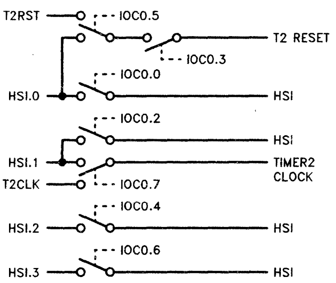


Figure 26. IOC0 Control of HSI Pin Functions

6.2 HSI FIFO

When an HSI event occurs, a 9×20 FIFO stores the 16 bits of Timer 1 and the 4 bits indicating which pins had events. It can take up to 8 state times for this information to reach the holding register. For this reason, 8 state times must be allowed between consecutive reads of HSI_TIME. When the FIFO is full, for a total of 8 events, were be stored by considering the holding register part of the FIFO. If the FIFO and holding register are full, any additional events will cause an overflow condition. Any eight consecutive events will overflow on the ninth event if the program does not clear all entries in the FIFO before the ninth event occurs.

6.3 HSI Interrupts

Interrupts can be generated by the HSI unit in three ways; two FIFO related interrupts and 0 to 1 transitions on the HSI.0 pin. The HSI.0 pin can generate interrupts even if it is not enabled to the HSI FIFO. Interrupts generated by this pin cause a vector through location 2008H. The FIFO related interrupts are controlled by bit 7 of I/O Control Register 1, (IOC1.7). If the bit is a 0, then an interrupt will be generated every time a value is loaded into the holding register. If it is a 1, an interrupt will only be generated when the FIFO, (independent of the holding register), has six entries in it. Since all interrupts are rising edge triggered, if IOC1.7 = 1, the processor will not be re-interrupted until the FIFO first contains 5 or less records, then contains six or more.

6.4 HSI Status

Bits 6 and 7 of the I/O Status register 1 (IOS1) indicate the status of the HSI FIFO. If bit 6 is a 1, the FIFO contains at least six entries. If bit 7 is a 1, the FIFO contains at least 1 entry and the HSI holding register has data available to be read. The FIFO may be read after verifying that it contains valid data. Caution must be used when reading or testing bits in IOS1, as this action clears bits 0–5, including the software and hardware timer overflow flags. It is best to store the byte and then test the stored value. See Section 11.

Reading the HSI is done in two steps. First, the HSI Status register is read to obtain the current state of the HSI pins and which pins had changed at the recorded time. The format of the HSI_STATUS Register is shown in Figure 27. Second, the HSI Time register is read. Reading the Time register unloads one level of the FIFO, so if the Time register is read before the Status register, the event information in the Status register will be lost. The HSI Status register is at location 06H and the HSI Time registers are in locations 04H and 05H.

If the HSI_TIME register is read without the holding register being loaded, the returned value will be indeterminate. Under the same conditions, the four bits in

HSI_STATUS indicating which events have occurred will also be indeterminate. The four HSI_STATUS bits which indicate the current state of the pins will always return the correct value.

It should be noted that many of the Status register conditions are changed by a reset, see Section 13. A complete listing of the functions of IOS0, IOS1, and IOC1 can be found in Section 11.

7.0 HIGH SPEED OUTPUTS

The High Speed Output unit, (HSO), is used to trigger events at specific times with minimal CPU overhead. These events include: starting an A to D conversion, resetting Timer 2, setting 4 software flags, and switching 6 output lines (HSO.0 through HSO.5). Up to eight events can be pending at one time and interrupts can be generated whenever any of these events are triggered. HSO.4 and HSO.5 are bidirectional pins which can also be used as HSI.2 and HSI.3 respectively. Bits 4 and 6 of I/O Control Register 1, (IOC1.4, IOC1.6), enable HSO.4 and HSO.5 as outputs.

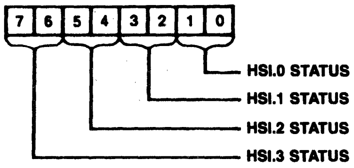
The HSO unit can generate two types of interrupts. The HSO execution interrupt (vector = (2006H)) is generated (if enabled) for HSO commands which operate one or more of the six output pins. The other HSO interrupt is the software timer interrupt (vector = (200BH)) which is generated (if enabled) by any other HSO command, (e.g. triggering the A/D, resetting Timer 2 or generating a software time delay).

7.1 HSO CAM

A block diagram of the HSO unit is shown in Figure 28. The Content Addressable Memory (CAM) file is the center of control. One CAM register is compared with the timer values every state time, taking 8 state times to compare all CAM registers with the timers. This defines the time resolution of the HSO to be 8 state times (2.0 microseconds at an oscillator frequency of 12 MHz).

HSI Status Register (HSI_Status)

LOCATION 06H



270250-26

Where for each 2-bit status field the lower bit indicates whether or not an event has occurred on this pin at the time in HSI_TIME and the upper bit indicates the current status of the pin.

Figure 27. HSI Status Register Diagram

Each CAM register is 23 bits wide. Sixteen bits specify the time at which the action is to be carried out and 7 bits specify both the nature of the action and whether Timer 1 or Timer 2 is the reference. The format of the

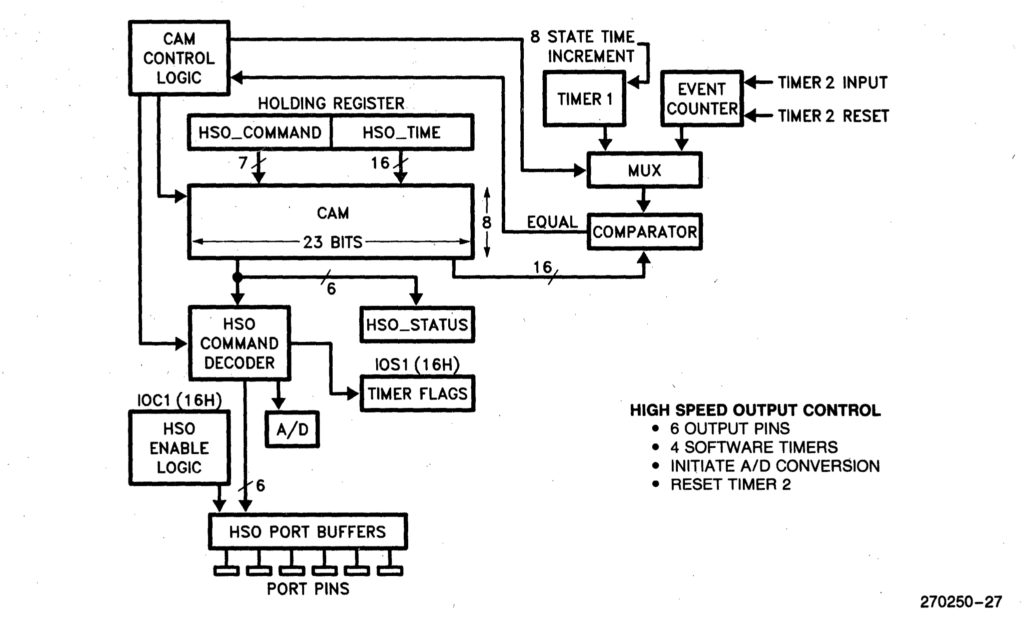


Figure 28. High Speed Output Unit

command to the HSO unit is shown in Figure 29. Note that bit 5 is ignored for command channels 8 through 0FH.

To enter a command into the CAM file, write the 7-bit "Command Tag" into location 0006H followed by the time at which the action is to be carried out into word address 0004H. The typical code would be:

```
LDB HSO_COMMAND,#what_to_do
ADD HSO_TIME,TIMER1,#when_to_do_it
```

Writing the time value loads the HSO Holding Register with both the time and the last written command tag. The command does not actually enter the CAM file until an empty CAM register becomes available.

Commands in the holding register will not execute even if their time tag is reached. Commands must be in the CAM for this to occur. Commands in the holding register can also be overwritten. Since it can take up to 8 state times for a command to move from the holding register to the CAM, 8 states must be allowed between successive writes to the CAM.

To provide proper synchronization, the minimum time that should be loaded to Timer 1 is Timer 1 + 2. Smaller values may cause the Timer match to occur 65,636 counts later than expected. A similar restriction applies if Timer 2 is used.

Care must be taken when writing the command tag for the HSO. If an interrupt occurs during the time between writing the command tag and loading the time value, and the interrupt service routine writes to the HSO time register, the command tag used in the interrupt routine will be written to the CAM at both the time specified by the interrupt routine and the time specified by the main program. The command tag from the main program will not be executed. One way of avoiding this problem would be to disable interrupts when writing commands and times to the HSO unit. See also Section 4.5.

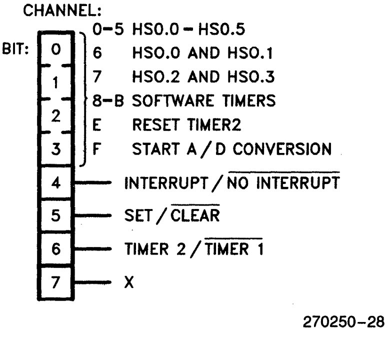


Figure 29. HSO Command Tag Format

7.2 HSO Status

Before writing to the HSO, it is desirable to ensure that the Holding Register is empty. If it is not, writing to the HSO will overwrite the value in the Holding Register. I/O Status Register 0 (IOS0) bits 6 and 7 indicate the status of the HSO unit. This register is described in Section 11. If IOS0.6 equals 0, the holding register is empty and at least one CAM register is empty. If IOS0.7 equals 0, the holding register is empty.

The programmer should carefully decide which of these two flags is the best to use for each application.

7.3 Clearing the HSO

All 8 CAM locations of the HSO are compared before any action is taken. This allows a pending external event to be cancelled by simply writing the opposite event to the CAM. However, once an entry is placed in the CAM, it cannot be removed until either the specified timer matches the written value or the chip is reset. If, as an example, a command has been issued to set HSO.1 when TIMER 1 = 1234, then entering a second command which clears HSO.1 when TIMER 1 = 1234 will result in no operation on HSO.1. Both commands will remain in the CAM until TIMER 1 = 1234.

Internal events are not synchronized to Timer 1, and therefore cannot be cleared. This includes events on HSO channels 8 through F and all interrupts. Since interrupts are not synchronized it is possible to have multiple interrupts at the same time value.

7.4 Using Timer 2 with the HSO

Timer 1 is incremented only once every 8 state-times. When it is being used as the reference timer for an HSO action, the comparator has a chance to look at all 8 CAM registers before Timer 1 changes its value. Following the same reasoning, Timer 2 has been synchronized to allow it to change at a maximum rate of once per 8 state-times. Timer 2 increments on both edges of the input signal.

When using Timer 2 as the HSO reference, caution must be taken that Timer 2 is not reset prior to the highest value for a Timer 2 match in the CAM. This is because the HSO CAM will hold an event pending until a time match occurs, if that match is to a time value on Timer 2 which is never reached, the event will remain pending in the CAM until the device is reset.

Additional caution must be used when Timer 2 is being reset using the HSO unit, since resetting Timer 2 using the HSO is an internal event and can therefore happen at any time within the eight-state-time window. This situation arises when the event is set to occur when

Timer 2 is equal to zero. If HSI.0 or the T2RST pin is used to clear Timer 2, and Timer 2 equal to zero triggers the event, then the event may not occur. This is because HSI.0 and T2RST clear Timer 2 asynchronously, and Timer 2 may then be incremented to one before the HSO CAM entry can be read and acted upon. This can be avoided by setting the event to occur when Timer 2 is equal to one. This method will ensure that there is enough time for the CAM entry recognition.

The same asynchronous nature can affect events scheduled to occur at the same time as an internal Timer 2 reset. These events should be logged into the CAM with a Timer 2 value of zero. When using this method to make a programmable modulo counter, the count will stay at the maximum Timer 2 value only until the Reset T2 command is recognized. The count will stay at zero for the transition which would have changed the count from "N" to zero, and then changed to a one on the next transition.

7.5 Software Timers

The HSO can be programmed to generate interrupts at preset times. Up to four such "Software Timers" can be in operation at a time. As each preprogrammed time is reached, the HSO unit sets a Software Timer Flag. If the interrupt bit in the command tag was set then a Software Timer Interrupt will also be generated. The interrupt service routine can then examine I/O Status register 1 (IOS1) to determine which software timer expired and caused the interrupt. When the HSO resets Timer 2 or starts an A to D conversion, it can also be programmed to generate a software timer interrupt but there is no flag to indicate that this has occurred.

If more than one software timer interrupt occurs in the same time frame it is possible that multiple software timer interrupts will be generated.

Each read or test of any bit in IOS1 will clear bits 0 through 5. Be certain to save the byte before testing it unless you are only concerned with 1 bit. See also Section 11.5.

A complete listing of the functions of IOS0, IOS1, and IOC1 can be found in Section 11. The Timers are described in Section 5 and the HSI is described in Section 6.

8.0 ANALOG INTERFACE

The 8096H can easily interface to analog signals using its Analog to Digital Converter and its Pulse-Width-Modulated (PWM) output and HSO Unit. Analog inputs are accepted by the 8-input, 10-bit A to D converter. The PWM and HSO units provide digital signals which can be filtered for use as analog outputs.

8.1 Analog Inputs

A to D conversion is performed on one of the 8 inputs at a time using successive approximation with a result equal to the ratio of the input voltage divided by the analog supply voltage. If the ratio is 1.00, then the result will be all ones. The A/D converter is available on selected members of the MCS-96 family. See Section 14 for the device selection matrix.

Each conversion on the 8096BH requires 88 state-times (22 μs at 12 MHz) independent of the accuracy desired or value of input voltage. The input voltage must be in the range of 0 to V_{REF}, the analog reference and supply voltage. For proper operation, V_{REF} (the reference voltage and analog power supply) must be held nominally at 5V. The A/D result is calculated from the formula:

$$1023 \times (\text{input voltage-ANGND})/(V_{REF}\text{-ANGND})$$

It can be seen from this formula that changes in V_{REF} or ANGND effect the output of the converter. This can be advantageous if a ratiometric sensor is used since these sensors have an output that can be measured as a proportion of V_{REF}.

ANGND must be tied to V_{SS} (digital ground) in order for the 8096BH to operate properly. This common connection should be made as close to the chip as possible, and using good bulk and high frequency by-pass capacitors to decouple power supply variations and noise from the circuit. Analog design rules call for one and only one common connection between analog and digital returns to eliminate unwanted ground variations.

The A/D converter has sample and hold. The sampling window is open for 4 state times which are included in the 88 state-time conversion period. The exact timings of the A/D converter can be found in Section 3 of the Hardware Design chapter.

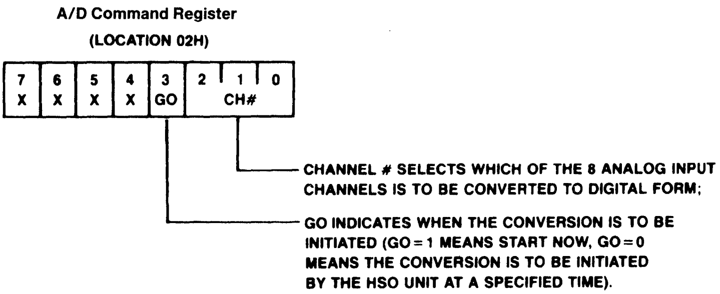
8.2 A/D Commands

Analog signals can be sampled by any one of the 8 analog input pins (ACH0 through ACH7) which are shared with Port 0. ACH7 can also be used as an external interrupt if IOC1.1 is set (see Sections 4 and 11). The A/D Command Register, at location 02H, selects which channel is to be converted and whether the conversion should start immediately or when the HSO (Channel #0FH) triggers it. The A/D command register must be written to for each conversion, even if the HSO is used as the trigger. A to D commands are formatted as shown in Figure 30.

The command register is double buffered so it is possible to write a command to start a conversion triggered by the HSO while one is still in progress. Care must be taken when this is done since if a new conversion is started while one is already in progress, the conversion in progress is cancelled and the new one is started. When a conversion is started, the result register is cleared. For this reason the result register must be read before a new conversion is started or data will be lost.

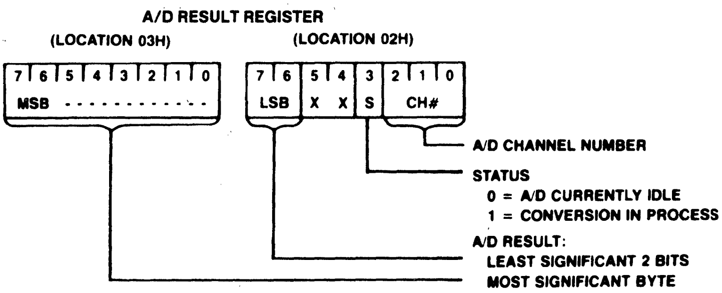
8.3 A/D Results

Results of the analog conversions are read from the A/D Result Register at locations 02H and 03H. Although these addresses are on a word boundary, they must be read as individual bytes. Information in the A/D Result register is formatted as shown in Figure 31. Note that the status bit may not be set until 8 state


A/D Command Register
(LOCATION 02H)
7 6 5 4 3 2 1 0
X X X X GO CH#
CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM;
GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).

270250-29

Figure 30. A/D Command Register



270250-30

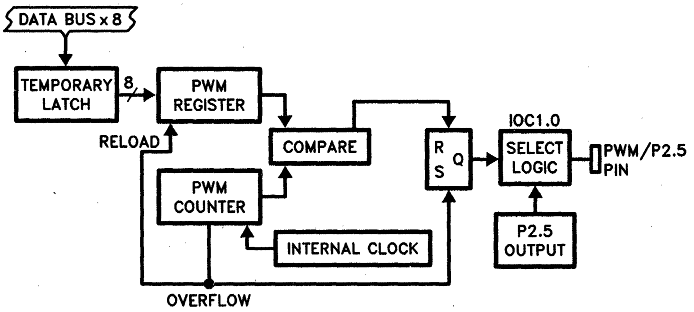
Figure 31. A/D Result Register

times after the go command, so it is necessary to wait 8 state times before testing it. Information on using the HSO is in Section 7.

8.4 Pulse Width Modulation Output (D/A)

Digital to analog conversion can be done with the Pulse Width Modulation output; a block diagram of the circuit is shown in Figure 32. The 8-bit counter is incremented every state time. When it equals 0, the PWM output is set to a one. When the counter matches the value in the PWM register, the output is switched low. When the counter overflows, the output is once again switched high. A typical output waveform is shown in Figure 33. Note that when the PWM register equals 00, the output is always low. Additionally, the PWM register will only be reloaded from the temporary latch when the counter overflows. This means that the compare circuit will not recognize a new value to compare against until the counter has expired the remainder of the current 8-bit count.

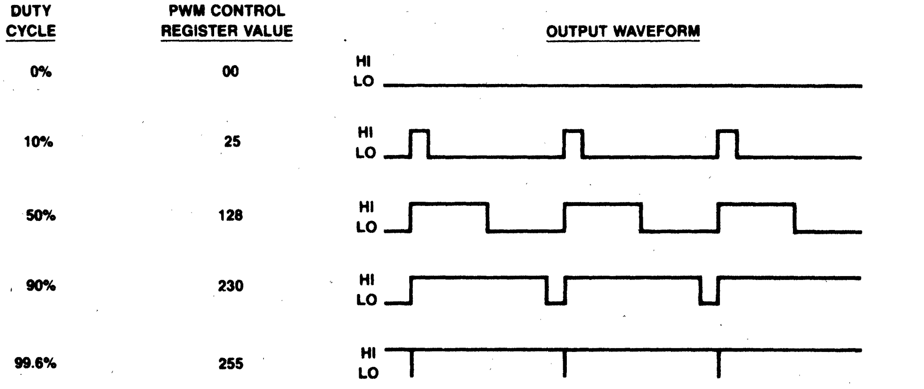
The output waveform is a variable duty cycle pulse which repeats every 256 state times (64 μs at 12 MHz). Changes in the duty cycle are made by writing to the PWM register at location 17H. There are several types of motors which require a PWM waveform for most efficient operation. Additionally, if this waveform is integrated it will produce a DC level which can be changed in 256 steps by varying the duty cycle.



270250-31

- PWM Period (XTAL = 12 MHz) = 64 μs, Frequency = 15.625 KHz
- Duty Cycle Programmable in 256 Steps

Figure 32. Pulse Width Modulated (D/A) Output



DUTY CYCLE	PWM CONTROL REGISTER VALUE
0%	00
10%	25
50%	128
90%	230
99.6%	255

270250-32

Figure 33. Typical PWM Outputs

Details about the hardware required for smooth, accurate D/A conversion can be found in Section 4 of the Hardware Design chapter. Typically, some form of buffer and integrator are needed to obtain the most usefulness from this feature.

The PWM output shares a pin with Port 2, pin 5 so that these two features cannot be used at the same time. IOC1.0 equal to 1 selects the PWM function instead of the standard port function. More information on IOC1 is in Section 11.

8.5 PWM Using the HSO

The HSO unit can be used to generate PWM waveforms with very little CPU overhead. If the HSO is not being used for other purposes, a 4 line PWM unit can be made by loading the on and off times into the CAM in sets of 4. The CAM would then always be loaded and only 2 interrupts per PWM period would be needed.

9.0 SERIAL PORT

The serial port on the 8096BH has 3 asynchronous and one synchronous mode. The asynchronous modes are full duplex, meaning they can transmit and receive at the same time. The receiver is double buffered so that the reception of a second byte can begin before the first byte has been read. The port is functionally compatible with the serial port on the MCS-51 family of microcontrollers, although the software used to control the ports is different.

Control of the serial port is handled through the Serial Port Control/Status Register at location 11H. Figure 37 shows the layout of this register. The details of using it to control the serial port will be discussed in Section 9.2.

Data to and from the serial port is transferred through SBUF (rx) and SBUF (tx), both located at 07H. Although these registers share the same address, they are physically separate, with SBUF (rx) containing the data received by the serial port and SBUF (tx) used to hold data ready for transmission. The program cannot write to SBUF (rx) or read from SBUF (tx).

The baud rate at which the serial port operates is controlled by an independent baud rate generator. The inputs to this generator can be either the XTAL1 or the T2CLK pin. Details on setting up the baud rate are given in Section 9.3.

9.1 Serial Port Modes

MODE 0

Mode 0 is a synchronous mode which is commonly used for shift register based I/O expansion. In this mode the TXD pin outputs a set of 8 pulses while the RXD pin either transmits or receives data. Data is transferred 8 bits at a time with the LSB first. A diagram of the relative timing of these signals is shown in Figure 34. Note that this is the only mode which uses RXD as an output.

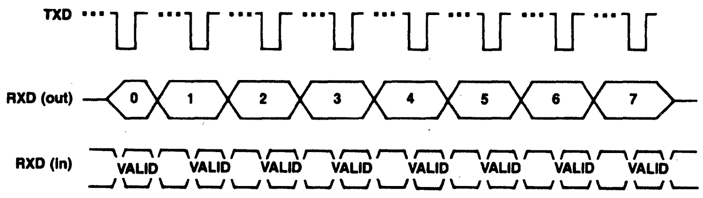


Figure 34. Serial Port Mode 0 Timing

Although it is not possible to transmit and receive at the same time using this mode, two external gates and a port pin can be used to time-multiplex the two functions. An example of multiplexing transmit and receive is discussed in Section 6.1 of the Hardware Design chapter.

MODE 1

Mode 1 is the standard asynchronous communications mode. The data frame used in this mode is shown in Figure 35. It consists of 10 bits; a start bit (0), 8 data bits (LSB first), and a stop bit (1). If parity is enabled, (the PEN bit is set to a 1), an even parity bit is sent instead of the 8th data bit and parity is checked on reception.

MODE 2

Mode 2 is the asynchronous 9th bit recognition mode. This mode is commonly used with Mode 3 for multiprocessor communications. Figure 36 shows the data frame used in this mode. It consists of a start bit (0), 9 data bits (LSB first), and a stop bit (1). When transmitting, the 9th bit can be set to a one by setting the TB8 bit in the control register before writing to SBUF (tx). The TB8 bit is cleared on every transmission, so it must be set prior to writing to SBUF (tx) each time it is desired. During reception, the serial port interrupt and the Receive Interrupt (RI) bit will not be set unless the 9th bit being received is set. This provides an easy way to have selective reception on a data link. Parity cannot be enabled in this mode.

MODE 3

Mode 3 is the asynchronous 9th bit mode. The data frame for this mode is identical to that of Mode 2. The transmission differences between Mode 3 and Mode 2 are that parity can be enabled (PEN = 1) and cause the 9th data bit to take the even parity value. The TB8 bit can still be used if parity is not enabled (PEN = 0). When in Mode 3, a reception always causes an interrupt, regardless of the state of the 9th bit. The 9th bit is stored if PEN = 0 and can be read in bit RB8. If PEN = 1 then RB8 becomes the Receive Parity Error (RPE) flag.

9.2 Controlling the Serial Port

Control of the serial port is done through the Serial Port Control (SP_CON) and Serial Port Status (SP_STAT) registers shown in Figure 37. Writing to location 11H accesses SP_CON while reading it access SP_STAT. Note that reads of SP_STAT will return indeterminate data in the lower 5 bits and writing to the upper 3 bits of SP_CON has no effect on chip functionality. The TB8 bit is cleared after each transmission and both TI and RI are cleared whenever SP_STAT (not SP_CON) is accessed. Whenever the TXD pin is used for the serial port it must be enabled by setting IOC1.5 to a 1. IOC1 is discussed further in Section 11.3. Information on the hardware connections and timing of the serial port is in Section 6 of the Hardware Design chapter.

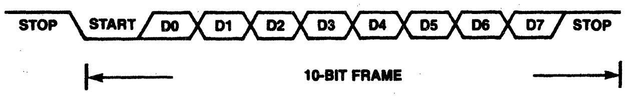


Figure 35. Serial Port Frame—Mode 1

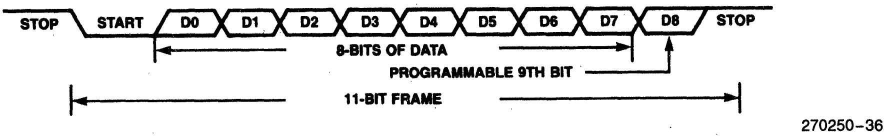
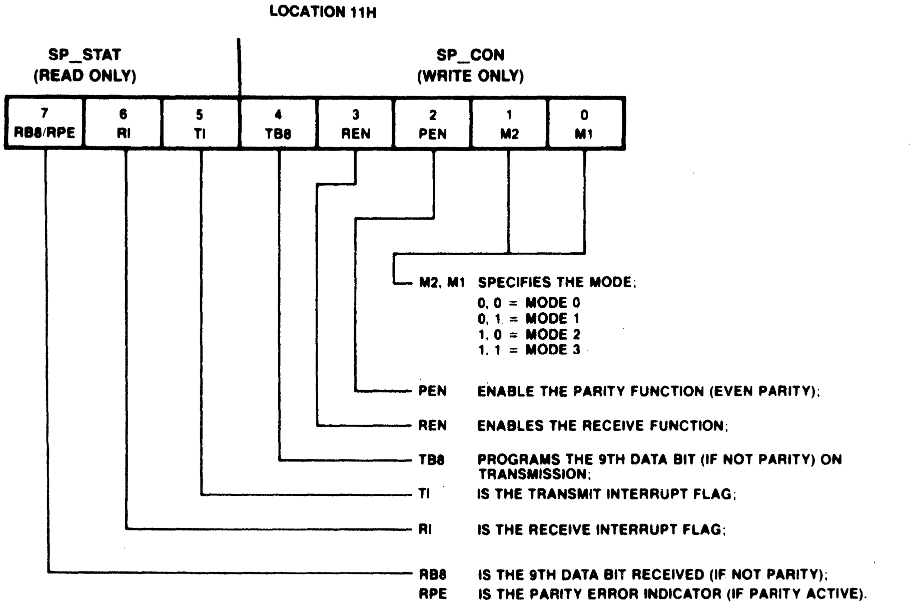


Figure 36. Serial Port Frame Modes 2 and 3


LOCATION 11H
SP_STAT
(READ ONLY)
SP_CON
(WRITE ONLY)
7 RB8/RPE
6 RI
5 TI
4 TB8
3 REN
2 PEN
1 M2
0 M1
M2, M1 SPECIFIES THE MODE;
0, 0 = MODE 0
0, 1 = MODE 1
1, 0 = MODE 2
1, 1 = MODE 3
PEN ENABLE THE PARITY FUNCTION (EVEN PARITY);
REN ENABLES THE RECEIVE FUNCTION;
TB8 PROGRAMS THE 9TH DATA BIT (IF NOT PARITY) ON TRANSMISSION;
TI IS THE TRANSMIT INTERRUPT FLAG;
RI IS THE RECEIVE INTERRUPT FLAG;
RB8 IS THE 9TH DATA BIT RECEIVED (IF NOT PARITY);
RPE IS THE PARITY ERROR INDICATOR (IF PARITY ACTIVE).

270250-33

NOTE:
TI and RI are cleared when SP_STAT is read.

Figure 37. Serial Port Control/Status Register

In Mode 0, if REN = 0, writing to SBUF (tx) will start a transmission. Causing a rising edge on REN, or clearing RI with REN = 1, will start a reception. Setting REN = 0 will stop a reception in progress and inhibit further receptions. To avoid a partial or complete undesired reception, REN must be set to zero before RI is cleared. This can be handled in an interrupt environment by using software flags or in straight-line code by using the Interrupt Pending register to signal the completion of a reception.

In the asynchronous modes, writing to SBUF (tx) starts a transmission. A falling edge on RXD will begin a reception if REN is set to 1. New data placed in SBUF (tx) is held and will not be transmitted until the end of the stop bit has been sent.

In all modes, the RI flag is set after the last data bit is sampled approximately in the middle of the bit time. Also for all modes, the TI flag is set after the last data bit (either 8th or 9th) is sent, also in the middle of the bit time. The flags clear when SP_STAT is read, but do not have to be clear for the port to receive or transmit. The serial port interrupt bit is set as a logical OR of the RI and TI bits. Note that changing modes will reset the Serial Port and abort any transmission or reception in progress on the channel. If the T_X and R_X pins are tied together for loopback testing, the RI flag will be written first.

9.3 Determining Baud Rates

Baud rates in all modes are determined by the contents of a 16-bit register at location 000EH. This register must be loaded sequentially with 2 bytes (least significant byte first). The serial port will not function between the loading of the first and second bytes. The MSB of this register selects one of two sources for the input frequency to the baud rate generator. If it is a 1, the frequency on the XTAL1 pin is selected, if not, the external frequency from the T2CLK pin is used. It should be noted that the maximum speed of T2CLK is one transition every 2 state times, with a minimum period of 16 XTAL1 cycles. This provides the needed synchronization to the internal serial port clocks.

The unsigned integer represented by the lower 15 bits of the baud rate register defines a number B, where B has a maximum value of 32767. The baud rate for the four serial modes using either XTAL1 or T2CLK as the clock source is given by:

Using XTAL1:

$$\text{Mode 0: } \begin{matrix}\text{Baud}\\ \text{Rate}\end{matrix} = \frac{\text{XTAL1 frequency}}{4 * (B + 1)}; \quad B \neq 0$$

$$\text{Others: } \frac{\text{Baud}}{\text{Rate}} = \frac{\text{XTAL1 frequency}}{64 * (B + 1)}$$

Using T2CLK:

$$\text{Mode 0: } \frac{\text{Baud}}{\text{Rate}} = \frac{\text{T2CLK frequency}}{B}; \quad B \neq 0$$

$$\text{Others: } \frac{\text{Baud}}{\text{Rate}} = \frac{\text{T2CLK frequency}}{16 * B}; \quad B \neq 0$$

Note that B cannot equal 0, except when using XTAL1 in other than mode 0.

Common baud rate values, using XTAL1 at 12 MHz, are shown below.

Baud Rate	Baud Register Value	
	Mode 0	Others
9600	8137H	8013H
4800	8270H	8026H
2400	84E1H	804DH
1200	89C3H	809BH
300	A70FH	8270H

The maximum baud rates are 1.5 Mbaud synchronous and 187.5 Kbaud asynchronous with 12 MHz on XTAL1.

9.4 Multiprocessor Communications

Mode 2 and 3 are provided for multiprocessor communications. In Mode 2 if the received 9th data bit is not 1, the serial port interrupt is not activated. The way to use this feature in multiprocessor systems is described below.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address frame which identifies the target slave. An address frame will differ from a data frame in that the 9th data bit is 1 in an address frame and 0 in a data frame. Slaves in Mode 2 will not be interrupted by a data frame. An address frame, however, will interrupt all slaves so that each slave can examine the received byte and see if it is being addressed. The addressed slave switches to Mode 3 to receive the coming data frames, while the slaves that were not addressed stay in Mode 2 and go on about their business.

10.0 I/O PORTS

There are five 8-bit I/O ports on the 8096. Some of these ports are input only, some are output only, some are bidirectional and some have alternate functions. In addition to these ports, the HSI/O unit can be used to provide extra I/O lines if the timer related features of these lines are not needed.

Input ports connect to the internal bus through an input buffer. Output ports connect through an output buffer to an internal register that hold the bits to be output. Bidirectional ports consist of an internal register, an input buffer, and an output buffer.

Port 0 is an input port which is also used as the analog input for the A to D converter. Port 1 is a quasi-bidirectional port. Port 2 contains three types of port lines: quasi-bidirectional, input and output. The input and output lines are shared with other functions in the 8096BH as shown in Table 4. Ports 3 and 4 are open-drain bidirectional ports which share their pins with the address/data bus.

Table 4. Port 2 Alternate Functions

Port	Function	Alternate Function	Controlled by
P2.0	Output	TXD (Serial Port Transmit)	IOC1.5
P2.1	Input	RXD (Serial Port Receive)	N/A
P2.2	Input	EXTINT (External Interrupt)	IOC1.1
P2.3	Input	T2CLK (Timer 2 Input)	IOC0.7
P2.4	Input	T2RST (Timer 2 Reset)	IOC0.5
P2.5	Output	PWM (Pulse-Width Modulation)	IOC1.0
P2.6	Quasi-Bidirectional		
P2.7	Quasi-Bidirectional		

Section 2 of the Hardware Design chapter contains additional information on the timing, drive capabilities, and input impedances of I/O pins.

10.1 Input Ports

Input ports and pins can only be read. There are no output drivers on these pins. The input leakage of these pins is in the microamp range. The specific values can be found in the data sheet for the device being considered.

In addition to acting as a digital input, each line of Port 0 can be selected to be the input of the A to D converter as discussed in Section 8. The pins on Port 0 are tested to have D.C. leakage of 3 microamps or less, as specified in the data sheet for the device being considered. The capacitance on these pins is approximately 5 pF and will instantaneously increase by around 5 pF when the pin is being sampled by the A to D converter.

The 8096BH samples the input to the A/D for 4 state times at the beginning of the conversion. Details on the A to D converter can be found in Section 8 of this chapter and in Section 3 of the Hardware Design chapter.

10.2 Quasi-Bidirectional Ports

Port 1, Port 2.6 and Port 2.7 are quasi-bidirectional ports. "Quasi-bidirectional" means that the port pin has a weak internal pullup that is always active and an internal pulldown which can be on to output a 0, or off to output a 1. If the internal pulldown is left off (by writing a 1 to the pin), the pin's logic level can be controlled by an external pulldown. If the external pulldown is on, it will input a 0 to the 8096BH, if it is off, a 1 will be input. From the user's point of view, the main difference between a quasi-bidirectional port and a standard input port is that the quasi-bidirectional port will source current if externally pulled low. It will also pull itself high if left unconnected.

In parallel with the weak internal pullup is a much stronger internal pullup that is activated for one state time when the pin is internally driven from 0 to 1. This is done to speed up the 0-to-1 transition time. When this pullup is on the pin can typically source 30 milliamps to V_{SS}.

When the processor writes to the pins of a quasi-bidirectional port it actually writes into a register which in turn drives the port pin. When the processor reads these ports, it senses the status of the pin directly. If a port pin is to be used as an input then the software should write a one to its associated SFR bit, this will cause the low-impedance pull-down device to turn off and leave the pin pulled up with a relatively high impedance pullup device which can be easily driven down by the device driving the input.

If some pins of a port are to be used as inputs and some are to be used as outputs the programmer should be careful when writing to the port.

Particular care should be exercised when using XOR opcodes or any opcode which is a read-modify-write instruction. It is possible for a Quasi-Bidirectional Pin to be written as a one, but read back as a zero if an external device (i.e., a transistor base) is pulling the pin below V_{IH}. See the Hardware Design Chapter Section 2.2 for further details on using the Quasi-Bidirectional Ports.

10.3 Output Ports

Output pins include the bus control lines, the HSO lines, and some of Port 2. These pins can only be used as outputs as there are no input buffers connected to them. It is not possible to use immediate logical instructions such as XOR PORT2, #00111B to toggle these pins. The output currents on these ports is higher than that of the quasi-bidirectional ports.

10.4 Ports 3 and 4/AD0-15

These pins have two functions. They are either bidirectional ports with open-drain outputs or System Bus pins which the memory controller uses when it is accessing off-chip memory. If the $\overline{\text{EA}}$ line is low, the pins always act as the System Bus. Otherwise they act as bus pins only during a memory access. If these pins are being used as ports and bus pins, ones must be written to them prior to bus operations.

Accessing Port 3 and 4 as I/O is easily done from internal registers. Since the LD and ST instructions require the use of internal registers, it may be necessary to first move the port information into an internal location before utilizing the data. If the data is already internal, the LD is unnecessary. For instance, to write a word value to Port 3 and 4 . . .

```
LD intreg, portdata   ; register ← data
                      ; not needed if already internal

ST intreg, 1FFEH      ; register → Port 3 and 4
```

To read Port 3 and 4 requires that "ones" be written to the port registers to first setup the input port configuration circuit. Note that the ports are reset to this input condition, but if zeroes have been written to the port, then ones must be re-written to any pins which are to be used as inputs. Reading Port 3 and 4 from a previously written zero condition is as follows . . .

```
LD intregA, #0FFFFH ; setup port change mode pattern

ST intregA, 1FFEH   ; register --> Port 3 and 4
                    ; LD & ST not needed if previously
                    ; written as ones

LD intregB, 1FFEH   ; register <-- Port 3 and 4
```

Note that while the format of the LD and ST instructions are similar, the source and destination directions change.

When acting as the system bus the pins have strong drivers to both V_{CC} and V_{SS}. These drivers are used whenever data is being output on the system bus and are not used when data is being output by Ports 3 and 4. Only the pins and input buffers are shared between the bus and the ports. The ports use different output buffers which are configured as open-drain, and require pullup resistors. (open-drain is the MOS version of open-collector.) The port pins and their system bus functions are shown in Table 5.

Table 5. P3,4/AD0-15 Pins

Port Pin	System Bus Function
P3.0	AD0
P3.1	AD1
P3.2	AD2
P3.3	AD3
P3.4	AD4
P3.5	AD5
P3.6	AD6
P3.7	AD7
P4.0	AD8
P4.1	AD9
P4.2	AD10
P4.3	AD11
P4.4	AD12
P4.5	AD13
P4.6	AD14
P4.7	AD15

11.0 STATUS AND CONTROL REGISTERS

There are two I/O Control registers, IOC0 and IOC1. IOC0 controls Timer 2 and the HSI lines. IOC1 controls some pin functions, interrupt sources and 2 HSO pins.

Whenever input lines are switched between two sources, or enabled, it is possible to generate transitions on these lines. This could cause problems with respect to edge sensitive lines such as the HSI lines, Interrupt line, and Timer 2 control lines.

11.1 I/O Control Register 0 (IOC0)

IOC0 is located at 0015H. The four HSI lines can be enabled or disabled to the HSI unit by setting or clearing bits in IOC0. Timer 2 functions including clock and reset sources are also determined by IOC0. The control bit locations are shown in Figure 38. IOC0 is for initialization only.

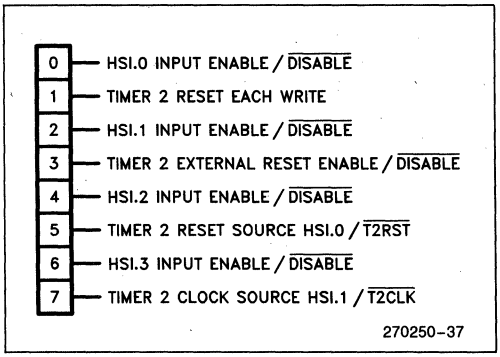

0 — HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1 — TIMER 2 RESET EACH WRITE
2 — HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3 — TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4 — HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5 — TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6 — HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7 — TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$
270250-37

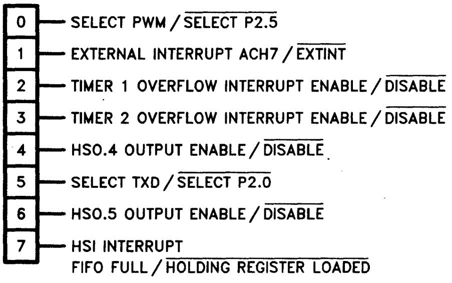
Figure 38. I/O Control Register 0 (IOC0)

11.2 I/O Control Register 1 (IOC1)

IOC1 is used to select some pin functions and enable or disable some interrupt sources. Its location is 0016H. Port pin P2.5 can be selected to be the PWM output instead of a standard output. The external interrupt source can be selected to be either EXTINT (same pin as P2.2) or Analog Channel 7 (ACH7, same pin as P0.7). Timer 1 and Timer 2 overflow interrupts can be individually enabled or disabled. The HSI interrupt can be selected to activate either when there is 1 FIFO entry or 7. Port pin P2.0 can be selected to be the TXD output. HSO.4 and HSO.5 can be enabled or disabled to the HSO unit. More information on interrupts is available in Section 4. The positions of the IOC1 control bits are shown in Figure 39.

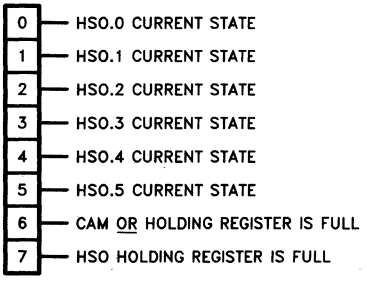
11.3 I/O Status Register 0 (IOS0)

There are two I/O Status registers, IOS0 and IOS1. IOS0, located at 0015H, holds the current status of the HSO lines and CAM. The status bits of IOS0 are shown in Figure 40.


0 SELECT PWM / SELECT P2.5
1 EXTERNAL INTERRUPT ACH7 / EXTINT
2 TIMER 1 OVERFLOW INTERRUPT ENABLE / DISABLE
3 TIMER 2 OVERFLOW INTERRUPT ENABLE / DISABLE
4 HSO.4 OUTPUT ENABLE / DISABLE
5 SELECT TXD / SELECT P2.0
6 HSO.5 OUTPUT ENABLE / DISABLE
7 HSI INTERRUPT
FIFO FULL / HOLDING REGISTER LOADED

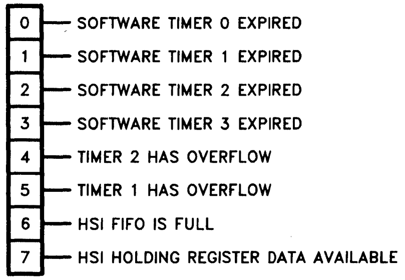
270250-38

Figure 39. I/O Control Register 1 (IOC1)



270250-39

Figure 40. I/O Status Register 0 (IOS0)


0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE

270250-40

Figure 41. HSIO Status Register 1 (IOS1)

11.4 I/O Status Register 1 (IOS1)

IOS1 is located at 016H. It contains status bits for the timers and the HSI/O. The positions of these bits are shown in Figure 41.

Whenever the processor reads this register all of the time-related flags (bits 5 through 0) are cleared. This applies not only to explicit reads such as:

```
LDB   AL,IOS1
```

but also to implicit reads such as:

```
JB    IOS1.3,somewhere_else
```

which jumps to somewhere_else if bit 3 of IOS1 is set. In most cases this situation can best be handled by having a byte in the register file which is used to maintain an image of lower five bits of the register. Any time a hardware timer interrupt or a HSO software timer interrupt occurs the byte can be updated:

```
ORB   IOS1_image,IOS1
```

leaving IOS1_image containing all the flags that were set before plus all the new flags that were read and cleared from IOS1. Any other routine which needs to sample the flags can safely check IOS1_image. Note that if these routines need to clear the flags that they have acted on, then the modification of IOS1_image must be done from inside a critical region (see Section 4.4).

12.0 WATCHDOG TIMER

The WatchDog Timer (WDT) provides a means to recover gracefully from a software upset. When the watchdog is enabled it will initiate a hardware reset unless the software clears it every 64K state times.

The WDT is implemented as an 8-bit timer with an 8-bit prescaler. The prescaler is not synchronized, so the timer will overflow between 65280 and 65535 state times after being reset. When the timer overflows it pulls down the $\overline{\text{RESET}}$ pin for at least one state time, resetting the 8096BH and any other devices tied to the $\overline{\text{RESET}}$ line. If a large capacitor is connected to the line, the pin may take a long time to go low. This will effect the length of time the pin is low and the voltage on the pin when it is finished falling. Section 1.4 of the Hardware Design chapter contains more information about reset hardware connections.

The WDT is enabled the first time it is cleared. Once it is enabled, it can only be disabled by resetting the 8096BH. The internal bit which controls the watchdog can typically maintain its state through power glitches as low as V_{SS} and as high as 7.0V for up to one millisecond.

Enabling and clearing the WDT is done by writing a "01EH" followed by a "0E1H" to the WDT register at location 0AH. This double write is used to help prevent accidental clearing of the timer.

12.1 Software Protection Hints

Glitches and noise on the PC board can cause software upsets, typically by changing either memory locations or the program counter. These changes can be internal to the chip or be caused by bad data returning to the chip.

There are both hardware and software solutions to noise problems, but the best solution is good design practice and a few ounces of prevention. The software can be designed so that the watchdog times out if the program does not progress properly. The watchdog will also time-out if the software error was due to ESD (Electrostatic Discharge) or other hardware related problems. This prevents the controller from having a malfunction for longer than 16 milliseconds if a 12 MHz oscillator is used.

When using the WDT to protect software it is desirable to reset it from only one place in code. This will lessen the chance that an undesired WDT reset will occur. The section of code that resets the WDT should monitor the other code sections for proper operation. This can be done by checking variables to make sure they are within reasonable values. Simply using a software timer to reset the WDT every 15 milliseconds will not provide much protection against minor problems.

It is also recommended that unused areas of code be filled with NOPs and periodic jumps to an error routine or RST (reset chip) instructions. This is particularly important in the code around lookup tables, since if lookup tables are executed undesired results will occur. Wherever space allows, each table should be surrounded by 7 NOPs (the longest 8096 instruction has 7 bytes) and a RST or jump to error routine instruction. Since RST is a one-byte instruction, the NOPs are not needed if RSTs are used instead of jumps to an error routine. This will help to ensure a speedy recovery should the processor have a glitch in the program flow. Since RST instruction has an opcode of 0FFH, pulling the data lines high with resistors will cause an RST to be executed if unimplemented memory is addressed.

12.2 Disabling The Watchdog

The watchdog should be disabled by software not initializing it. If this is not possible, such as during program development, the watchdog can be disabled by holding the $\overline{\text{RESET}}$ pin at 2.0V to 2.5V. Voltages over 2.5V on the pin could quickly damage the device. Even at 2.5V, using this technique for other than debugging purposes is not recommended, as it may effect long term reliability. It is further recommended that any device used in this way for more than several seconds, not be used in production versions of products. Section 1.6 of the Hardware Design chapter has more information on disabling the Watchdog Timer.

13.0 RESET

13.1 Reset Signal

As with all processors, the 8096BH must be reset each time the power is turned on. This is done by holding the $\overline{\text{RESET}}$ pin low for at least 2 state times after the power supply is within tolerance and the oscillator has stabilized. (See Figure 44, TRLPV.)

After the $\overline{\text{RESET}}$ pin is brought high, a ten state reset sequence is executed. During this time, the Chip Configuration Byte (CCB) is read from location 2018H and written to the 8096BH Chip Configuration Register (CCR). If the voltage on the $\overline{\text{EA}}$ pin selects the internal/external execution mode the CCB is read from internal ROM/EPROM. If the voltage on the $\overline{\text{EA}}$ pin selects the external execution only mode the CCB is read from external memory.

The 8096BH can be reset using a capacitor, 1-shot, or any other method capable of providing a pulse of at least 2 state times longer than required for V_{CC} and the oscillator to stabilize.

For best functionality, it is suggested that the reset pin be pulled low with an open collector device. In this way, several reset sources can be wire ORed together. Remember, the $\overline{\text{RESET}}$ pin itself can be a reset source when the RST instruction is executed or when the Watchdog Timer overflows. Details of hardware suggestions for reset can be found in Section 1.4 of the Hardware Design chapter.

13.2 Reset Status

The I/O lines and control lines of the 8096BH will be in their reset state within IOXTAL1 periods after reset is low, with V_{CC} and the oscillator stabilized (See Figure 44, TRLPV). Prior to that time, the status of the I/O lines is indeterminate. After the 10 state time reset sequence, the Special Function Registers will be set as follows:

Register	Reset Value
Port 1	XXXXXXXXB
Port 2	XX0XXXX1B
Port 3	11111111B
Port 4	11111111B
PWM Control	00H
Serial Port (Transmit)	undefined
Serial Port (Receive)	undefined
Baud Rate Register	undefined
Serial Port Control	XXXX0XXXB
Serial Port Status	X00XXXXXB
A/D Command	undefined
A/D Result	undefined
Interrupt Pending	undefined
Interrupt Mask	00000000B
Timer 1	0000H
Timer 2	0000H
Watchdog Timer	0000H
HSI Mode	XXXXXXXXB
HSI Status	undefined
IOS0	00000000B
IOS1	00000000B
IOC0	X0X0X0X0B
IOC1	X0X0XXX1B
HSI FIFO	empty
HSO CAM	empty
HSO SFR	000000B
PSW	0000H
Stack Pointer	undefined
Program Counter	2080H

Figure 42. Register Reset Status

Port 1 and Port 2.6, 2.7 reset to a floating or weak pullup condition. HSO.4 and HSO.5 reset to a floating condition as they are disabled by IOC1.4 and IOC1.6.

Other conditions following a reset are:

Pin	Reset Value
$\overline{\text{RD}}$	high
$\overline{\text{WR/WRL}}$	high
ALE/$\overline{\text{ADV}}$	high
$\overline{\text{BHE/WRH}}$	high
INST	low
HSO Lines	XX0000B

Figure 43. Bus Control Pins Reset Status

It is important to note that the Stack Pointer and Interrupt Pending Register are undefined, and need to be initialized in software. The Interrupts are disabled by both the mask register and PSW.9 after a reset.

13.3 Reset Sync Mode

The $\overline{\text{RESET}}$ line can be used to start the 8096BH at an exact state time to provide for synchronization of test equipment and multiple chip systems. $\overline{\text{RESET}}$ is active low. To synchronize devices, $\overline{\text{RESET}}$ is brought high on the rising edge of XTAL1. Complete details on synchronizing devices can be found in Section 1.5 of the Hardware Design chapter.

It is very possible that devices which start in sync may not stay that way. The best example of this would be when a "jump on I/O bit" is being used to hold the processor in a loop. If the line changes during the time it is being tested, one processor may see it as a one, while the other sees it as a zero. The result is that one processor will do an extra loop, thus putting it several states out of sync with the other.

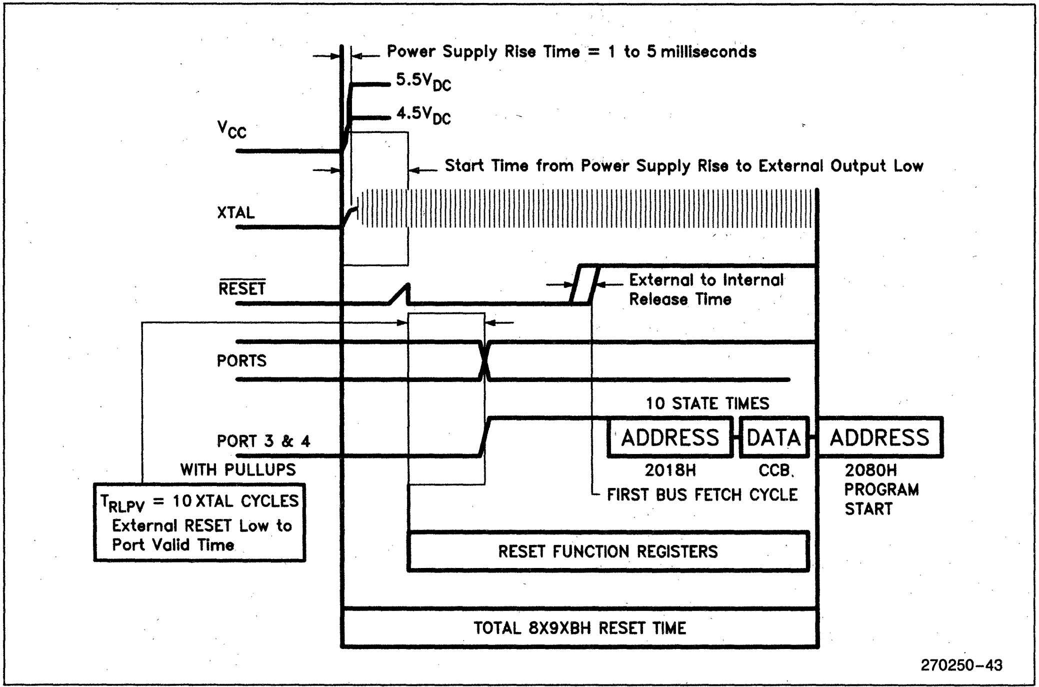

Power Supply Rise Time = 1 to 5 milliseconds
5.5V_DC
4.5V_DC
V_CC
Start Time from Power Supply Rise to External Output Low
XTAL
RESET
External to Internal Release Time
PORTS
10 STATE TIMES
PORT 3 & 4
WITH PULLUPS
ADDRESS
DATA
ADDRESS
2018H
CCB.
2080H PROGRAM START
FIRST BUS FETCH CYCLE
T_RLPV = 10 XTAL CYCLES External RESET Low to Port Valid Time
RESET FUNCTION REGISTERS
TOTAL 8X9XBH RESET TIME
270250–43

Figure 44. TRLPV

MCS®-96 8096BH Hardware Design Information and Data Sheets

2

8096BH HARDWARE DESIGN INFORMATION

OVERVIEW

This Chapter of the manual is devoted to the hardware engineer. All of the information you need to connect the correct pin to the correct external circuit is provided. Many of the special function pins have different characteristics which are under software control, therefore, it is necessary to define the system completely before the hardware is wired-up.

Frequently within this chapter a specification for a current, voltage, or time period is referred to; the values provided are to be used as an approximation only. The exact specification can be found in the latest data sheet for the particular device and temperature range that is being used.

1.0 REQUIRED HARDWARE CONNECTIONS

Although the 8096BH is a single-chip microcontroller, it still requires several external connections to make it work. Power must be applied, a clock source provided, and some form of reset circuitry must be present. We will look at each of these areas of circuitry separately. Figure 6 shows the connections that are needed for a single-chip system.

1.1 Power Supply Information

Power for the 8096BH flows through six pins; they are: three positive voltage pins—V_{CC} (digital), V_{REF} (Port 0 digital I/O and A/D power), V_{PD} (power down mode); and three common returns—two V_{SS} pins and one ANGND pin. All six of these pins **must** be connected on the 8096BH for normal operation. The V_{CC} pin, V_{REF} pin and V_{PD} pin should be tied to 5 volts. The two V_{SS} pins and the ANGND pin must be grounded. When the analog to digital converter is being used it may be desirable to connect the VREF pin to a separate power supply, or at least a separate power supply line.

The three common return pins should be connected at the chip with as short a lead as possible to avoid problems due to voltage drops across the wiring. There should be no measurable voltage difference between V_{SS1} and V_{SS2}. The two V_{SS} pins and the ANGND pin must all be nominally at 0 volts. The maximum current drain of the 8096BH is around 180 mA, with all lines unloaded.

When the analog converter is being used, clean, stable power must be provided to the analog section of the chip to assure highest accuracy. To achieve this, it may be desirable to separate the analog power supply from the digital power supply. The V_{REF} pin supplies the digital circuitry in the A/D converter and provides the 5 volt reference to the analog portion of the converter. V_{REF} and ANGND must be connected even if the A/D converter is not used. More information on the analog power supply is in Section 3.1.

1.2 Other Needed Connections

Several other connections are needed to configure the 8096BH. In normal operation the following pins should be connected to the indicated power supply.

Pin	Power Supply
NMI	V_{CC}
$\overline{EA}$	V_{CC} (to allow internal execution) V_{SS} (to force external execution)

Although the $\overline{EA}$ pin has an internal pulldown, it is best to tie this pin to the desired level. This will prevent induced noise from disturbing the system. Raising $\overline{EA}$ to + 12.75 volts will place an 8096BH in a special operating mode designed for programming and program memory verification (see Section 10).

1.3 Oscillator Information

The 8096BH requires a clock source to operate. This clock is provided to the chip through the XTAL1 input. The frequency of operation is from 6 MHz to 12 MHz.

The on-chip circuitry for the 8096BH oscillator is a single stage linear inverter as shown in Figure 1. It is intended for use as a crystal-controlled, positive reactance oscillator with external connections as shown in Figure 2. In this application, the crystal is being operated in its fundamental response mode as an inductive reactance in parallel resonance with shunt capacitance external to the crystal.

The crystal specifications and capacitance values (C1 and C2 in Figure 2) are not critical. Thirty picofarads can be used in these positions at any frequency with good quality crystals. For 0.5% frequency accuracy, the crystal frequency can be specified at series resonance or for parallel resonance with any load capacitance. (In other words, for that degree of frequency accuracy, the load capacitance simply doesn't matter.) For 0.05% frequency accuracy the crystal frequency should be specified for parallel resonance with 25 pF load capacitance, if C1 and C2 are 30 pF.

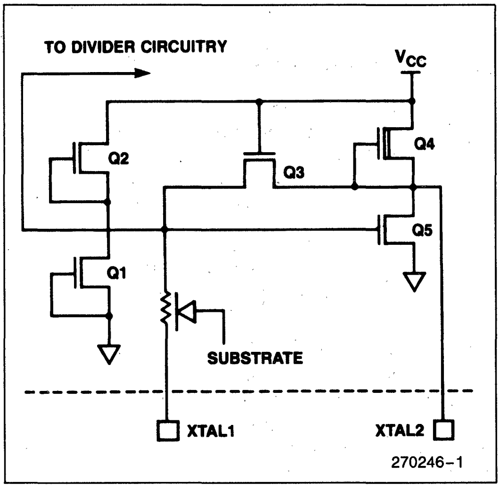


Figure 1. 8096BH Oscillator Circuit

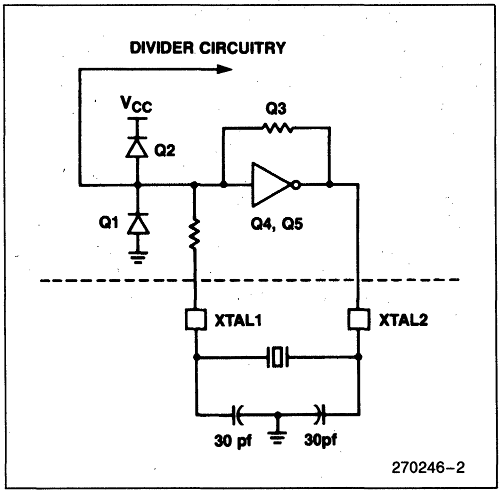


Figure 2. Crystal Oscillator Circuit

An external oscillator may encounter as much as a 100 pF load at XTAL1 when it starts up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{IL} and V_{IH} specifications the capacitance will not exceed 20 pF.

A more in-depth discussion of crystal specifications and the selection of values for C1 and C2 can be found in the Intel Application Note, AP-155, "Oscillators for Microcontrollers."

To drive the 8096BH with an external clock source, apply the external clock signal to XTAL1 and let XTAL2 float. An example of this circuit is shown in Figure 3. The required voltage levels on XTAL1 are specified in the data sheet. The signal on XTAL1 must be clean with good solid levels.

It is important that the minimum high and low times are met to avoid having the XTAL1 pin in the transition range for long periods of time. The longer the signal is in the transition region, the higher the probability that an external noise glitch could be seen by the clock generator circuitry. Noise glitches on the 8096BH internal clock lines will cause unreliable operation.

The clock generator provides a 3 phase clock output from the XTAL1 pin input. Figure 4 shows the waveforms of the major internal timing signals.

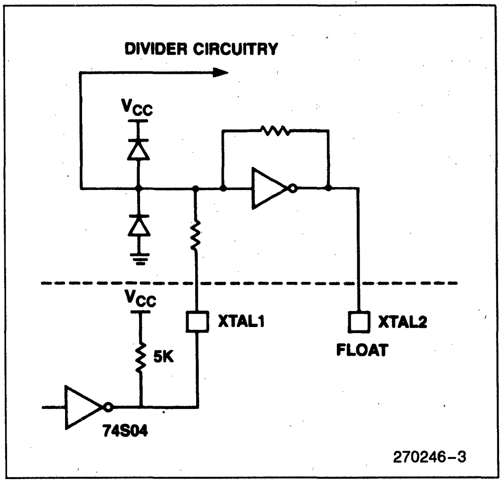


Figure 3. External Clock Drive

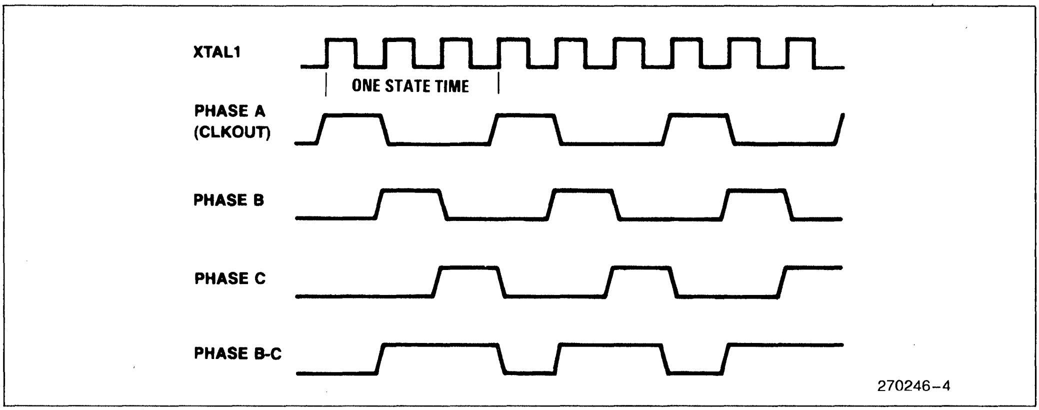


Figure 4. Internal Timings

1.4 Reset Information

In order for the 8096BH to function properly it must be reset. This is done by holding the $\overline{\text{RESET}}$ pin low for at least 10 XTAL1 cycles after the power supply is within tolerance and the oscillator has stabilized.

After the $\overline{\text{RESET}}$ pin is brought high, a ten state reset sequence is executed. During this time, the Chip Configuration Byte (CCB) is read from location 2018H and written to the 8096BH Chip Configuration Register (CCR). If the voltage on the $\overline{\text{EA}}$ pin selects the internal/external execution mode the CCB is read from internal ROM/EPROM. If the voltage on the $\overline{\text{EA}}$ pin selects the external execution only mode the CCB is read from external memory. See Figure 5, and 5A.

There are several ways to provide a good reset to an 8096BH, the simplest being just to connect a capacitor from the reset pin to ground. The capacitor should be on the order of 2 microfarads for every millisecond of reset time required. This method will only work if the rise time of V_{CC} is fast and the total reset time is less than around 50 milliseconds. It also may not work if the $\overline{\text{RESET}}$ pin is to be used to reset other devices on the board. An 8096BH with the minimum required connections is shown in Figure 6.

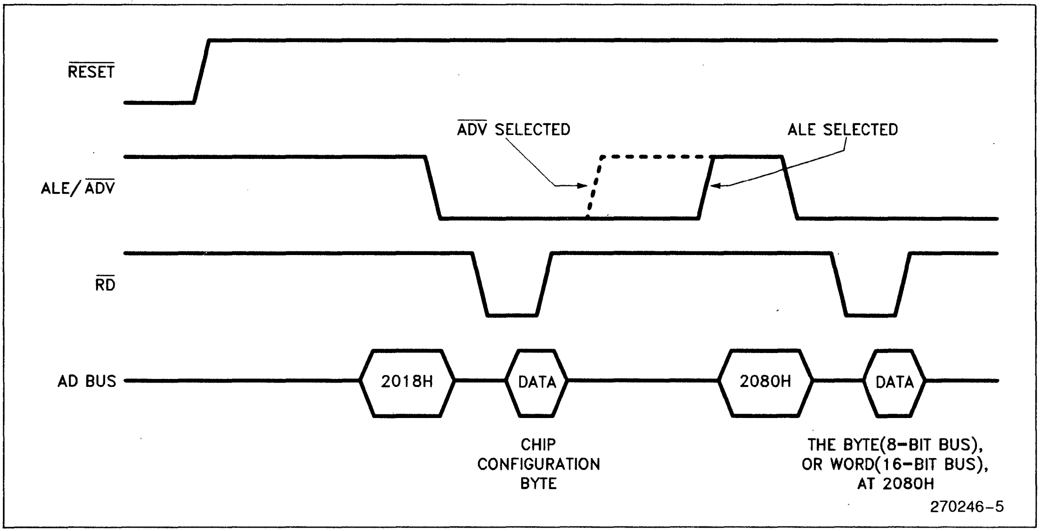


Figure 5. Reset Sequence

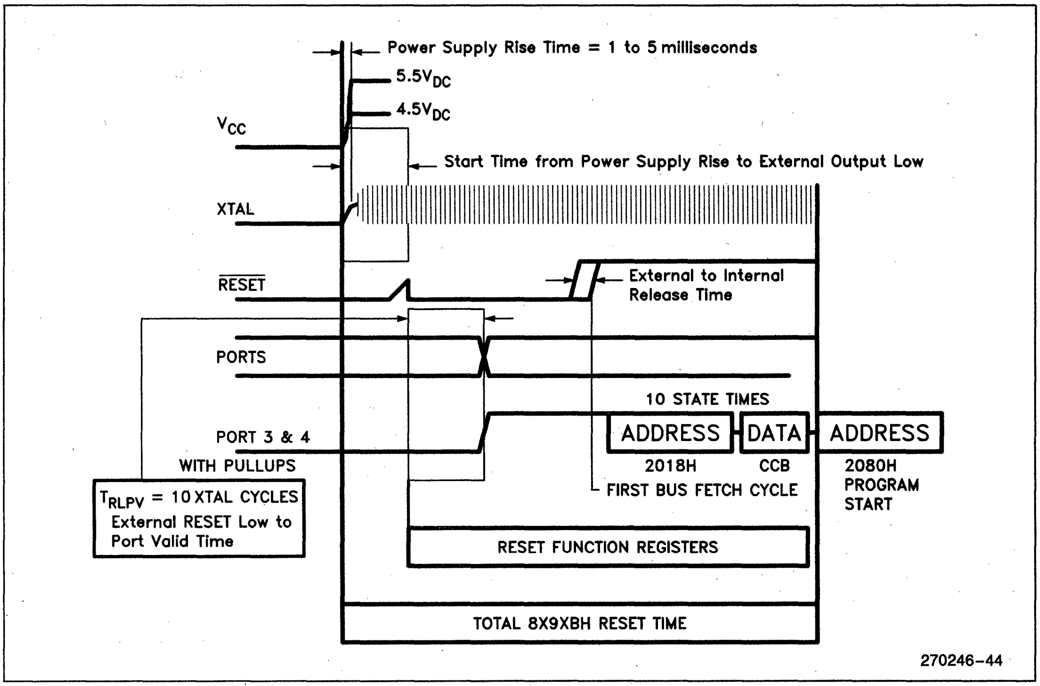
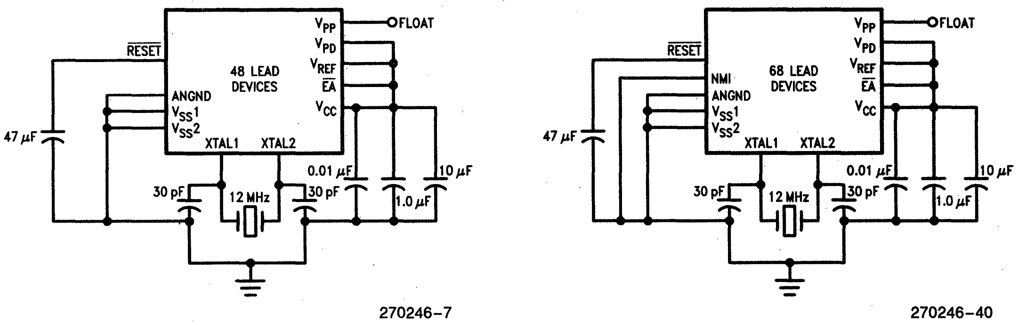

Power Supply Rise Time = 1 to 5 milliseconds
5.5V_DC
4.5V_DC
V_CC
Start Time from Power Supply Rise to External Output Low
XTAL
RESET
External to Internal Release Time
PORTS
10 STATE TIMES
PORT 3 & 4 WITH PULLUPS
ADDRESS
DATA
ADDRESS
2018H
CCB
2080H PROGRAM START
FIRST BUS FETCH CYCLE
T_RLPV = 10 XTAL CYCLES External RESET Low to Port Valid Time
RESET FUNCTION REGISTERS
TOTAL 8X9XBH RESET TIME
270246-44

Figure 5A. T_{RLPV}

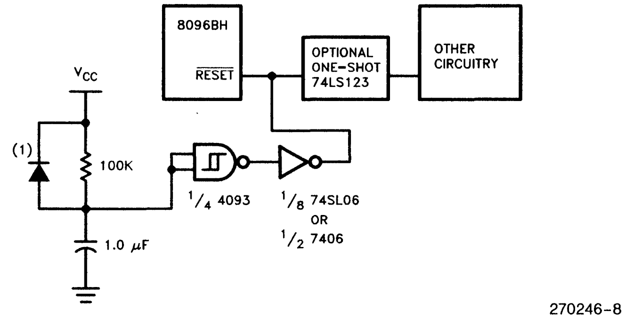


NOTES:
1. These capacitors are needed only if A to D is used.
2. V_{REF} & ANGND may be connected to the same traces as the digital power supply if the A to D is not used.

Figure 6. Minimum Hardware Connections

The 8096BH $\overline{\text{RESET}}$ pin can be used to allow other chips on the board to make use of the Watchdog Timer or the RST instruction. When this is done the reset hardware should be a one-shot with an open collector output. The reset pulse going to the other devices may have to be buffered and lengthened with a one-shot, since the $\overline{\text{RESET}}$ low duration is only one state times. If this is done, it is possible that the 8096BH will be reset and start running before the other devices on the board are out of reset. The software must account for this possible problem.

A capacitor directly connected to $\overline{\text{RESET}}$ cannot be used to reset the device if the pin is to be used as an output. If a large capacitor is used, the pin will pull-down more slowly than normal. It will continue to pull-down until the 8096BH is reset. It could fall so slowly that it never goes below the internal switch point of the reset signal (1 to 1.5 volts), a voltage which may be above the guaranteed switch point of external circuitry connected to the pin. A circuit example is shown in Figure 7.



NOTE:
1. The diode will provide a faster cycle time repetitive power-on-resets.

Figure 7. Multiple Chip Reset Circuit

1.5 Sync Mode

If $\overline{\text{RESET}}$ is brought high at the same time as or just after the rising edge of XTAL1, the device will start executing the 10 state time RST instruction exactly $6\frac{1}{2}$ XTAL1 cycles later. This feature can be used to synchronize several MCS-96 devices. A diagram of a typical connection is shown in Figure 8. It should be noted that devices that start in sync may not stay that way, due to propagation delays which may cause the synchronized devices to receive signals at slightly different times.

1.6 Disabling the Watchdog Timer

The Watchdog Timer will pull the $\overline{\text{RESET}}$ pin low when it overflows. See Figure 9. If the pin is being externally held above the low going threshold, the pulldown transistor will remain on indefinitely. This means that once the watchdog overflows, the device must be reset or $\overline{\text{RESET}}$ must be held high indefinitely. Just resetting the Watchdog Timer in software will not clear the flip-flop which keeps the $\overline{\text{RESET}}$ pulldown on.

The pulldown is capable of sinking on the order of 30 milliamps if it is held at 2.0 volts. This amount of current may cause some long term reliability problems due to localized chip heating. For this reason, devices that will be used in production should never have had the Watchdog Timer over-ridden for more than a second or two.

Whenever the reset pin is being pulled high while the pulldown is on, it should be through a resistor that will limit the voltage on $\overline{\text{RESET}}$ to 2.5 volts and the current through the pin to 40 milliamps.

If it is necessary to disable the Watchdog Timer for more than a brief test the software solution of never initiating the timer should be used. See Section 14 in the Architecture Chapter.

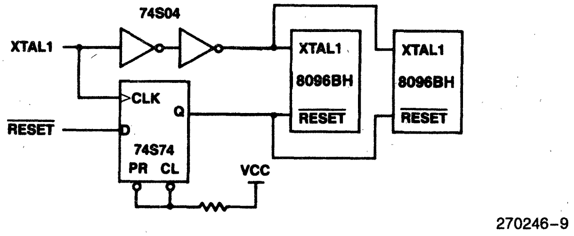


Figure 8. Reset Sync Mode

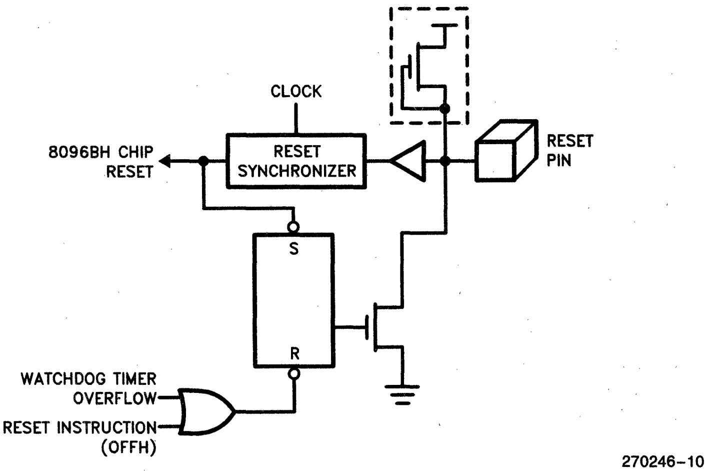


Figure 9. Reset Logic

1.7 Power Down Circuitry

Battery backup can be provided on the 8096BH with a 1 mA current drain at 5 volts. This mode will hold locations 0F0H through 0FFH valid as long as the power to the V_{PD} pin remains on. The required timings to put the device into power-down and an overview of this mode are given in Section 2.3 in the 8096BH Architecture Chapter.

A 'key' can be written into power-down RAM while the device is running. This key can be checked on reset to determine if it is a start-up from power-down or a complete cold start. In this way the validity of the power-down RAM can be verified. The length of this key determines the probability that this procedure will work, however, there is always a statistical chance that the RAM will power up with a replica of the key.

Under most circumstances, the power-fail indicator which is used to initiate a power-down condition must come from the unfiltered, unregulated section of the power supply. The power supply must have sufficient storage capacity to operate the 8096BH until it has completed its reset operation.

2.0 DRIVE AND INTERFACE LEVELS

There are five types of I/O lines on the 8096BH. Of these, two are inputs and three are outputs. All of the pins of the same type have the same current/voltage characteristics. Some of the control input pins, such as XTAL1 and $\overline{\text{RESET}}$, may have slightly different characteristics. These pins are discussed in Section 1.

While discussing the characteristics of the I/O pins some approximate current or voltage specifications will be given. The exact specifications are available in the lastest version of the data sheet that corresponds to the device being used.

2.1 Quasi-Bidirectional Ports

The Quasi-Bidirectional pins of Port 1, Port 2.6, and Port 2.7 have both input and output port configurations. They have three distinct states; low impedance current sink (Q2), low impedance current source (Q1), and high impedance current source (Q3). As a low impedance current sink, the pin has specification of sinking up to around 0.5 mA, while staying below 0.45 volts. The pin is placed in this condition by writing a '0' to the SFR (Special Function Register) controlling the pin.

Examine Figure 10. When a '1' is written to the SFR location controlling the pin, Q1 (a low impedance MOSFET pullup) is turned on for one state time, then it is turned off and the depletion pullup holds the line at a logical '1' state. The low-impedance pullup is used to shorten the rise time of the pin, and has current source capability on the order of 100 times that of the depletion pullup.

While the depletion mode pullup is the only device on, the pin may be used as an input with a leakage of around 100 microamps from 0.45 volts to V_{CC}. It is ideal for use with TTL or CMOS chips and may even be used directly with switches. However if the switch option is used, certain precautions should be taken. It is important to note that any time the pin is read, the value returned will be the value on the pin, not the value placed in the control register. This could cause logical operations made directly on these pins to indavertently write a 0 to pins being used as inputs. In order to perform logical operations on a port where a quasi-bidirectional pin is an input, it is necessary to guarantee that the bit associated with the input pin is always a one when writing to the port.

It is possible for asynchronous resets to occur at the same time the program is writing to the quasi-bidirectional port pins. If this should occur, only the high impedance pullup will be turned on and the pin will reset to a logical one more slowly than usual.

2.2 Quasi-Bidirectional Hardware Connections

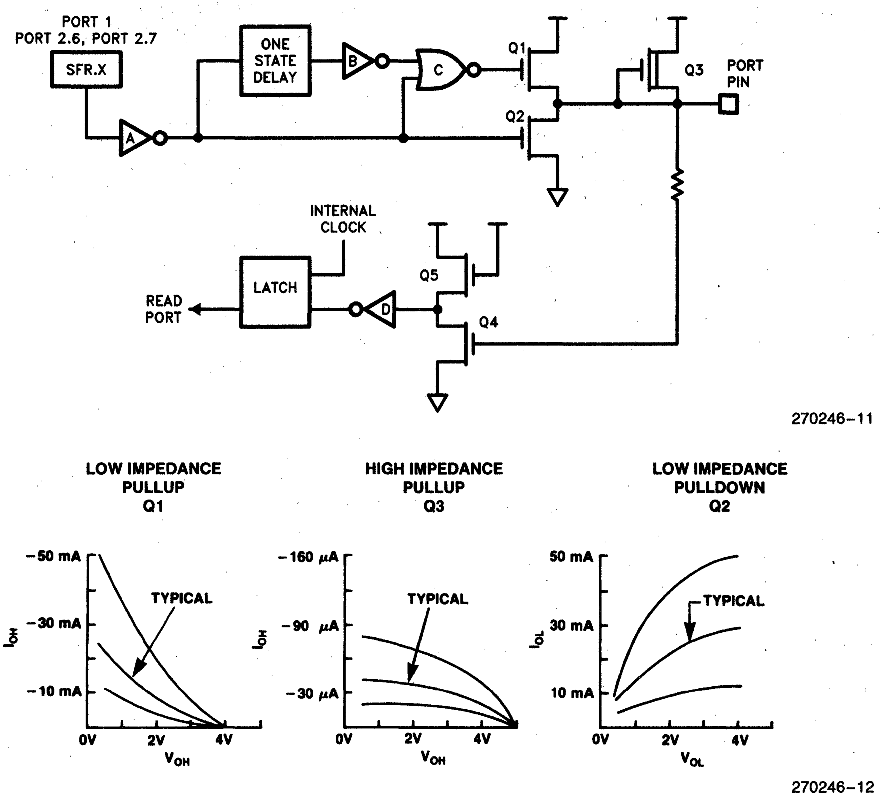
When using the quasi-bidirectional ports as inputs tied to switches, series resistors may be needed if the ports will be written to internally after the device is initialized. The amount of current sourced to ground from each pin is tyically 20 mA or more. Therefore, if all 8 pins are tied to ground, 160 mA will be sourced. This is equivalent to instantaneously doubling the power used by the chip and may cause noise in some applications.

This potential problem can be solved in hardware or software. In software, never write a zero to a pin being used as an input.

In hardware, a 1K resistor in series with each pin will limit current to a reasonable value without impeding the ability to override the high impedance pullup. If all 8 pins are tied together a 120Ω resistor would be reasonable. The problem is not quite as severe when the inputs are tied to electronic devices instead of switches, as most external pulldowns will not hold 20 mA to 0.0 volts.

Writing to a Quasi-Bidirectional Port with electronic devices attached to the pins requires special attention. Consider using P1.0 as an input and trying to toggle P1.1 as an output:

```
ORB  IOPORT1, #00000001B ; Set P1.0
                         ; for input
XORB IOPORT1, #00000010B ; Complement
                         ; P1.1
```



NOTE:
These graphs show typical pin capabilities, they are not guaranteed specifications

Figure 10. Quasi-Bidirectional Port

The first instruction will work as expected but two problems can occur when the second instruction executes. The first is that even though P1.1 is being driven high by the 8096BH it is possible that it is being held low externally. This typically happens when the port pin is used to drive the base of an NPN transistor which in turn drives whatever there is in the outside world which needs to be toggled. The base of the transistor will clamp the port pin to the transistor's Vbe above ground, typically 0.7V. The 8096BH will input this value as a zero even if a one has been written to the port pin. When this happens the XORB instruction will always write a one to the port pin's SFR and the pin will not toggle.

The second problem, which is related to the first, is that if P1.0 happens to be driven to a zero when Port 1 is read by the XORB instruction, then the XORB will write a zero to P1.0 and it will no longer be useable as an input.

The first situation can best be solved by the external driver design. A series resistor between the port pin and the base of the transistor often works by bringing up the voltage present on the port pin. The second case can be taken care of in the software fairly easily:

```
LDB  AL, IOPORT1
XORB AL, #010B
ORB  AL, #001B
STB  AL, IOPORT1
```

A software solution to both cases is to keep a byte in RAM as an image of the data to be output to the port; any time the software wants to modify the data on the port it can then modify the image byte and copy it to the port.

If a switch is used on a long line connected to a quasi-bidirectional pin, a pullup resistor is recommended to reduce the possibility of noise glitches and to decrease the rise time of the line. On extremely long lines that are handling slow signals, a capacitor may be helpful in addition to the resistor to reduce noise.

2.3 Input Only Ports

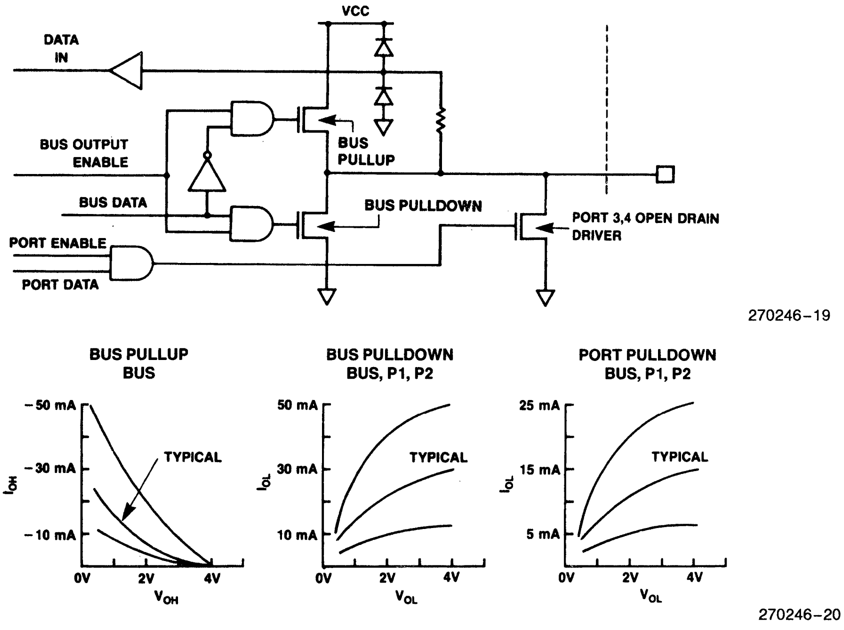
The high impedance input pins on the 8096BH have an input leakage of a few microamps and are predominantly capacitive loads on the order of 10 pF.

Port 0 pins are special in that they may individually be used as digital inputs, or as analog inputs. A Port 0 pin being used as a digital input acts as the high impedance input ports just described. However, Port 0 pins being used as analog inputs are required to provide current to the internal sample capacitor when a conversion begins. This means that the input characteristics of a pin will change if a conversion is being done on that pin. See Section 3. In either case, if Port 0 is to be used as analog or digital I/O, it will be necessary to provide power to this port through the V_{REF} pin.

2.4 Open Drain Ports

Ports 3 and 4 on the 8096BH are open drain ports. There is no pullup when these pins are used as I/O ports. These pins have different characteristics when used as bus pins as described in the next section. A diagram of the output buffers connected to Ports 3 and 4 and the bus pins is shown in Figure 11.

When Ports 3 and 4 are to be used as inputs, or as bus pins, they must first be written with a '1'. This will put the ports in a high impedance mode. When they are used as outputs, a pullup resistor must be used externally. The sink capability of these pins is on the order of 0.4 milliamps so the total pullup current to the pin must be less than this. A 15K pullup resistor will source a maximum of 0.33 milliamps, so it would be a reasonable value to choose if no other circuits with pullups were connected to the pin.



NOTE:
These graphs show typical pin capabilities, they are not guaranteed specifications.

Figure 11. Bus and Port 3 and 4 Pins

2.5 HSO Pins, Control Outputs and Bus Pins

The control outputs and HSO pins have output buffers with the same output characteristics as those of the bus pins. Included in the category of control outputs are: TXD, RXD (in Mode 0), PWM, CLKOUT, ALE, $\overline{\text{BHE}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$. The bus pins have 3 states: output high, output low, and high impedance input. As a high output, the pins are specified to source around 200 μA to 2.4 volts, but the pins can source on the order of ten times that value in order to provide the fast rise times. When used as a low output, the pins can sink around 2 mA at 0.45 volts, and considerably more as the voltage increases. When in the high impedance state, the pin acts as a capacitive load with a few microamps of leakage. Figure 11 shows the internal configuration of a bus pin.

3.0 ANALOG INPUTS

The on-chip A/D converter of the 8096BH can be used to digitize analog inputs while analog outputs can be generated with either the chip's PWM output or HSO unit. This section describes the analog input suggestions. See Section 4 for analog output.

The 8096BH's Integrated A/D converter includes an eight channel analog multiplexer, sample-and-hold circuit and 10-bit analog to digital converter (Figure 12). The 8096BH can therefore select one of eight analog inputs to convert, sample-and-hold the input voltage and convert the voltage into a digital value. Each conversion takes 22 microseconds, including the time required for the sample-hold (with XTAL1 = 12 MHz). The method of conversion is successive approximation.

Section 3.5 contains the definitions of numerous terms used in connection with the A/D converter.

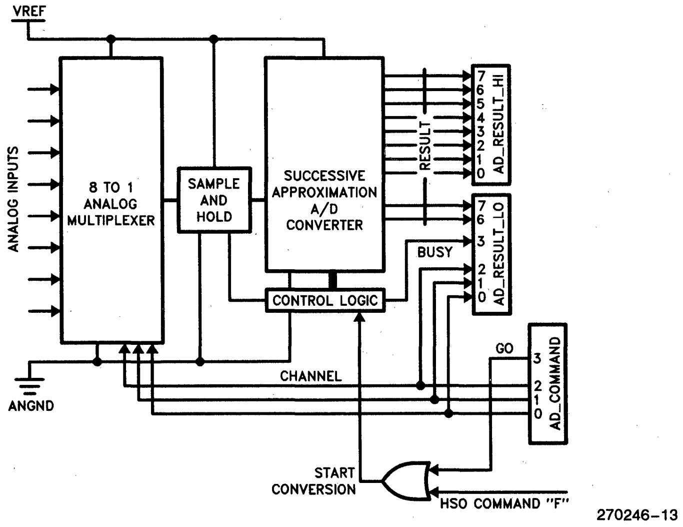


Figure 12. A/D Converter Block Diagram

3.1 A/D Overview

The conversion process is initiated by the execution of HSO command 0FH, or by writing a one to the GO Bit in the A/D Control Register. Either activity causes a start conversion signal to be sent to the A/D converter control logic. If an HSO command was used, the conversion process will begin when Timer 1 increments. This aids applications attempting to approach spectrally pure sampling, since successive samples spaced by equal Timer 1 delays will occur with a variance of about ±50 ns (assuming a stable clock on XTAL1). However, conversions initiated by writing a one to the ADCON register GO Bit will start within three state times after the instruction has completed execution resulting in a variance of about 0.75 μs (XTAL1 = 12 MHz).

Once the A/D unit receives a start conversion signal, there is a one state time delay before sampling (sample delay) while the successive approximation register is reset and the proper multiplexer channel is selected. After the sample delay, the multiplexer output is connected to the sample capacitor and remains connected for four state times (sample time). After this four state time "sample window" closes, the input to the sample capacitor is disconnected from the multiplexer so that changes on the input pin will not alter the stored charge while the conversion is in progress. The comparator is then auto-zeroed and the conversion begins. The sample delay and sample time uncertainties are each approximately ±50 ns, independent of clock speed.

To perform the actual analog-to-digital conversion the 8096BH implements a successive approximation algorithm. The converter hardware consists of a 256-resistor ladder, a comparator, coupling capacitors and a 10-bit successive approximation register (SAR) with logic that guides the process. The resistor ladder provides 20 mV steps (V_{REF} = 5.12V), while capacitive coupling is used to create 5 mV steps within the 20 mV ladder voltages. Therefore, 1024 internal reference voltages are available for comparison against the analog input to generate a 10-bit conversion result.

A successive approximation conversion is performed by comparing a sequence of reference voltages, to the analog input, in a binary search for the reference voltage that most closely matches the input. The ½ full scale reference voltage is the first tested. This corresponds to a 10-bit result where the most significant bit is zero, and all other bits are ones (0111.1111.11b). If the analog input was less than the test voltage, bit 10 of the SAR is left a zero, and a new test voltage of ¼ full scale (0011.1111.11b) is tried. If this test voltage was lower than the analog input, bit 9 of the SAR is set and bit 8 is cleared for the next test (0101.1111.11b). This binary search continues until 10 tests have occurred, at which time the valid 10-bit conversion result resides in the SAR where it can be read by software.

The total number of state times required is 88 for a 10-bit conversion. Attempting to short-cycle the 10-bit conversion process by reading A/D results before the done bit is set is not recommended.

3.2 A/D Interface Suggestions

The external interface circuitry to an analog input is highly dependent upon the application, and can impact converter characteristics. In the external circuit's design, important factors such as input pin leakage, sample capacitor size and multiplexer series resistance from the input pin to the sample capacitor must be considered.

For the 8096BH, these factors are idealized in Figure 13. The external input circuit must be able to charge a sample capacitor (C_S) through a series resistance (R_I) to an accurate voltage given a D.C. leakage (I_L). On the 8096BH, C_S is around 2 pF, R_I is around 5 KΩ and I_L is specified as 3 μA maximum. In determining the necessary source impedance R_S, the value of V_{BIAS} is not important.

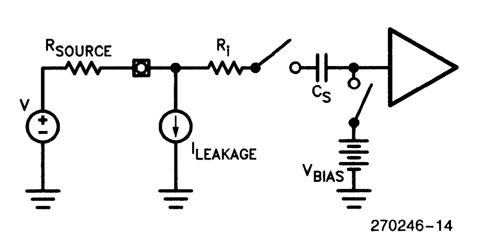


Figure 13. Idealized A/D Sampling Circuitry

External circuits with source impedances of 1 KΩ or less will be able to maintain an input voltage within a tolerance of about ±0.61 LSB (1.0 KΩ × 3.0 μA = 3.0 mV) given the D.C. leakage. Source impedances above 2 KΩ can result in an external error of at least one LSB due to the voltage drop caused by the 1 μA leakage. In addition, source impedances above 25 KΩ may degrade converter accuracy as a result of the internal sample capacitor not being fully charged during the 1 μs (12 MHz clock) sample window.

If large source impedances degrade converter accuracy because the sample capacitor is not charged during the sample time, an external capacitor connected to the pin will compensate for this degradation. Since the sample capacitor is 2 pF, a 0.005 μF capacitor will charge the sample capacitor to an accurate input voltage of ±0.5 LSB (2048 × 2 pF). An external capacitor does not compensate for the voltage drop across the source resistance, but charges the sample capacitor fully during the sample time.

Placing an external capacitor on each analog input will also reduce the sensitivity to noise, as the capacitor combines with series resistance in the external circuit to form a low-pass filter. In practice, one should include a small series resistance prior to the external capacitor on the analog input pin and choose the largest capacitor value practical, given the frequency of the signal being converted. This provides a low-pass filter on the input, while the resistor will also limit input current during over-voltage conditions.

Figure 14 shows a simple analog interface circuit based upon the discussion above. The circuit in the figure also provides limited protection against over-voltage conditions on the analog input. Should the input voltage inappropriately drop significantly below ground, diode D2 will forward bias at about 0.8 DCV. The equivalent circuit to Figure 14 essentially sets up an RC charge circuit with one time constant equal to 1.35 μs (270Ω × 0.005 μF). The charge on this circuit represents the amount of voltage that will appear at the input pin of the MC-96 converter. Momentary transitions below ground are not steady state conditions. The instantaneous charge will be 0V followed in time by increasingly negative voltage. Given the time constant discussed earlier, and assuming a transition below ground of −0.8V, the MCS-96 specification of −0.3V will not be exceeded for a duration of approximately 700 ns. This will effectively limit the voltage, and therefore the current through the pin, to a safe level for brief periods of time. *However, before any circuit is used in an actual application, it should be thoroughly analyzed for applicability to the specific problem at hand.*

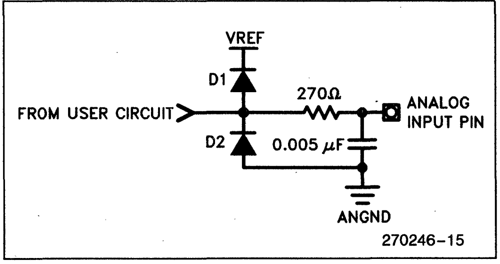


Figure 14. Suggested A/D Input Circuit

3.3 Analog References

Reference supply levels strongly influence the absolute accuracy of the conversion. For this reason, it is recommended that the ANGND pin be tied to the two V_{SS} pins as close to the chip as possible with minimum trace length. Bypass capacitors should also be used between V_{REF} and ANGND. ANGND should be within about a tenth of a volt V_{SS}. V_{REF} should be well regulated and used only for the A/D converter. The V_{REF} supply can be between 4.5V and 5.5V and needs to be able to source around 5 mA. Figure 6 shows all of these connections.

Note that if only ratiometric information is desired, V_{REF} can be connected to V. In addition, V_{REF} and ANGND must be connected even if the A/D converter is not being used. Remember that Port 0 receives its power from the V_{REF} and ANGND pins even when it is used as digital I/O.

3.4 The A/D Transfer Function

The conversion result is a 10-bit ratiometric representation of the input voltage, so the numerical value obtained from the conversion will be:

$$\text{INT } [1023 \times (V_{IN} - \text{ANGND})/(V_{REF} - \text{ANGND})].$$

This produces a stair-stepped transfer function when the output code is plotted versus input voltage (see Figure 15). The resulting digital codes can be taken as simple ratiometric information, or they can be used to provide information about absolute voltages or relative voltage changes on the inputs. The more demanding the application is on the A/D converter, the more important it is to fully understand the converter's operation. For simple applications, knowing the absolute error of the converter is sufficient. However, closing a servo-loop with analog inputs necessitates a detailed understanding of an A/D converter's operation and errors.

The errors inherent in an analog-to-digital conversion process are many: quantizing error; zero offset; full-scale error; differential non-linearity; and non-linearity. These are "transfer function" errors related to the A/D converter. In addition, converter temperature drift, V_{CC} rejection, sample-hold feedthrough, multiplexer off-isolation, channel-to-channel matching and random noise should be considered. Fortunately, one "Absolute Error" specification is available which describes the sum total of all deviations between the actual conversion process and an ideal converter. However, the various sub-components of error are important in many applications. These error components are described in Section 3.5 and in the text below where ideal and actual converters are compared.

An unavoidable error simply results from the conversion of a continuous voltage to an integer digital representation. This error is called quantizing error, and is always ±0.5 LSB. Quantizing error is the only error seen in a perfect A/D converter, and is obviously present in actual converters. Figure 15 shows the transfer function for an ideal 3-bit A/D converter (i.e. the Ideal Characteristic).

Note that in Figure 15 the Ideal Characteristic possesses unique qualities: it's first code transition occurs when the input voltage is 0.5 LSB; it's full-scale code transition occurs when the input voltage equals the full-scale reference minus 1.5 LSB; and it's code widths are all exactly one LSB. These qualities result in a digitization without offset, full-scale or linearity errors. In other words, a perfect conversion.

Figure 16 shows an Actual Characteristic of a hypothetical 3-bit converter, which is not perfect. When the Ideal Characteristic is overlaid with the imperfect characteristic, the actual converter is seen to exhibit errors in the location of the first and final code transitions and code widths. The deviation of the first code transition from ideal is called "zero offset", and the deviation of the final code transition from ideal is "full-scale error". The deviation of the code widths from ideal causes two types of errors. Differential Non-Linearity and Non-Linearity. Differential Non-Linearity is a local linearity error measurement, whereas Non-Linearity is an overall linearity error measure.

Differential Non-Linearity is the degree to which actual code widths differ from the ideal one LSB width. Differential Non-Linearity gives the user a measure of how much the input voltage may have changed in order to produce a one count change in the conversion result. Non-Linearity is the worst case deviation of code transitions from the corresponding code transitions of the Ideal Characteristic. Non-Linearity describes how much Differential Non-Linearities could add up to produce an overall maximum departure from a linear characteristic. If the Differential Non-Linearity errors are too large, it is possible for an A/D converter to miss codes or exhibit non-monotonicity. Neither behavior is desireable in a closed-loop system. A converter has no missed codes if there exists for each output code a unique input voltage range that produces that code only. A converter is monotonic if every subsequent code change represents an input voltage change in the same direction.

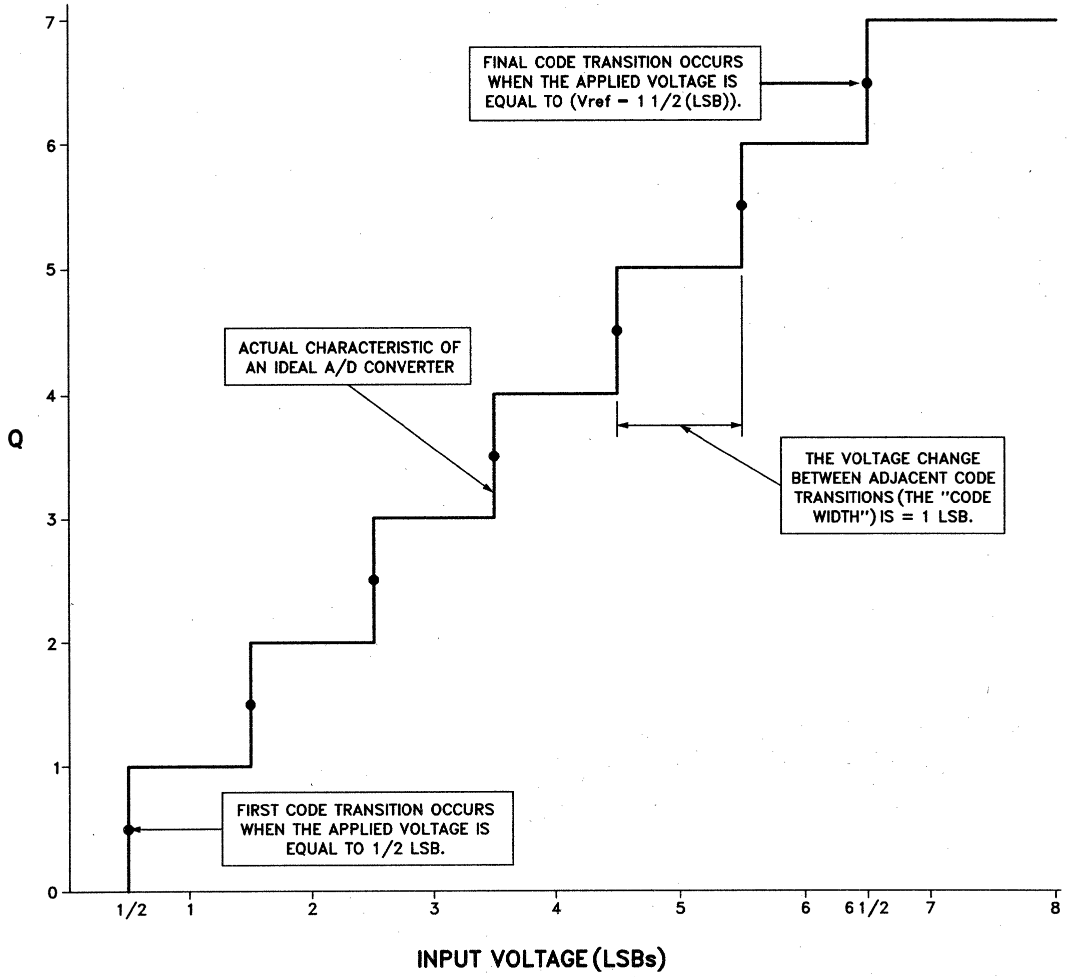
Differential Non-Linearity and Non-Linearity are quantified by measuring the Terminal Based Linearity Errors. A Terminal Based Characteristic results when an Actual Characteristic is shifted and rotated to eliminate zero offset and full-scale error (see Figure 17). The Terminal Based Characteristic is similar to the Actual Characteristic that would be seen if zero offset and full-scale error were externally trimmed away. In practice, this is done by using input circuits which include gain and offset trimming. In addition, V_{REF} on the 8096BH could also be closely regulated and trimmed within the specified range to affect full-scale error.

Other factors that affect a real A/D Converter system include sensitivity to temperature, failure to completely reject all unwanted signals, multiplexer channel dissimilarities and random noise. Fortunately these effects are small.

Temperature sensitivities are described by the rate at which typical specifications change with a change in temperature.

Undesired signals come from three main sources. First, noise on V_{CC}—V_{CC} Rejection. Second, input signal changes on the channel being converted after the sample window has closed—Feedthrough. Third, signals applied to channels not selected by the multiplexer—Off-Isolation.

Finally, multiplexer on-channel resistances differ slightly from one channel to the next causing Channel-to-Channel Matching errors, and random noise in general results in Repeatability errors.



FINAL CODE TRANSITION OCCURS WHEN THE APPLIED VOLTAGE IS EQUAL TO (Vref − 1 1/2 (LSB)).

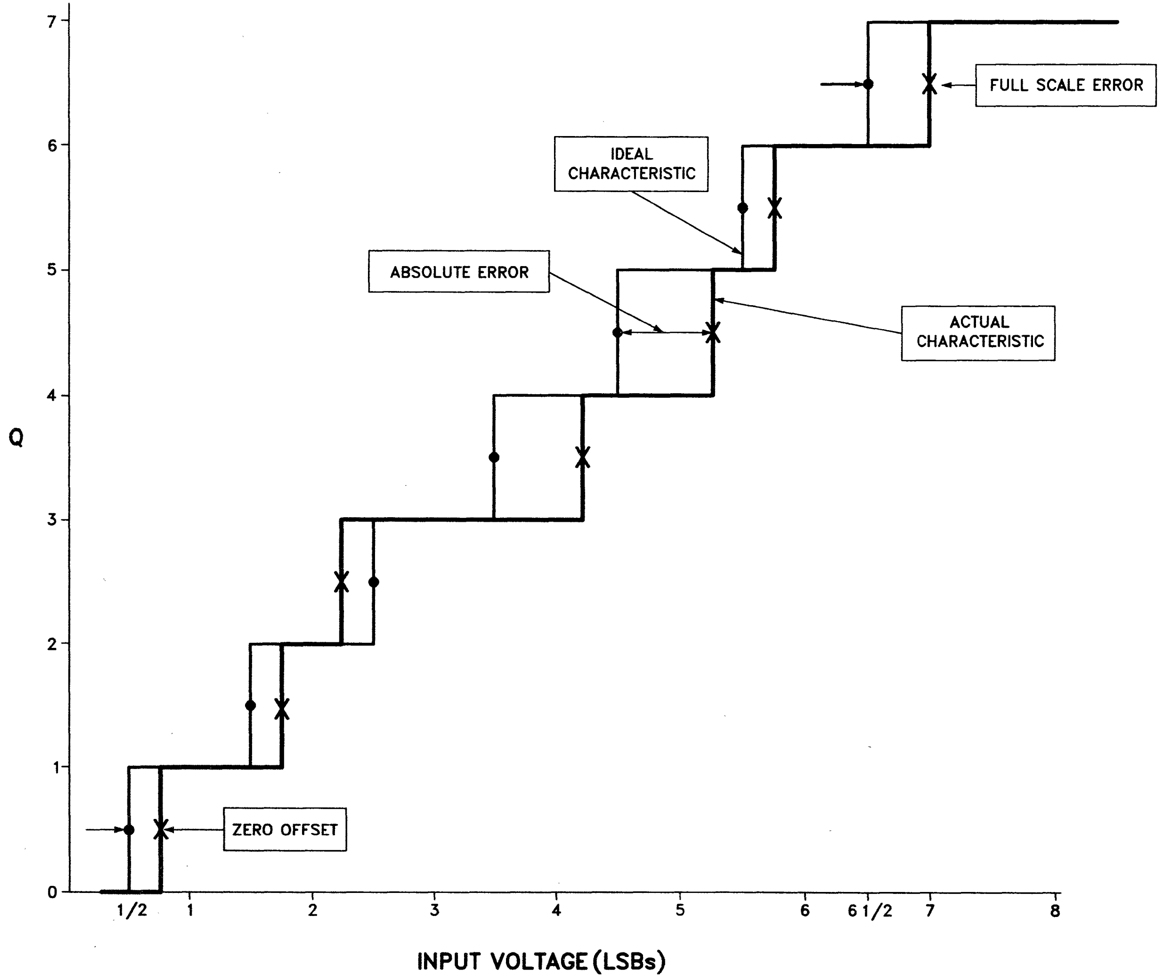
ACTUAL CHARACTERISTIC OF AN IDEAL A/D CONVERTER

THE VOLTAGE CHANGE BETWEEN ADJACENT CODE TRANSITIONS (THE "CODE WIDTH") IS = 1 LSB.

FIRST CODE TRANSITION OCCURS WHEN THE APPLIED VOLTAGE IS EQUAL TO 1/2 LSB.

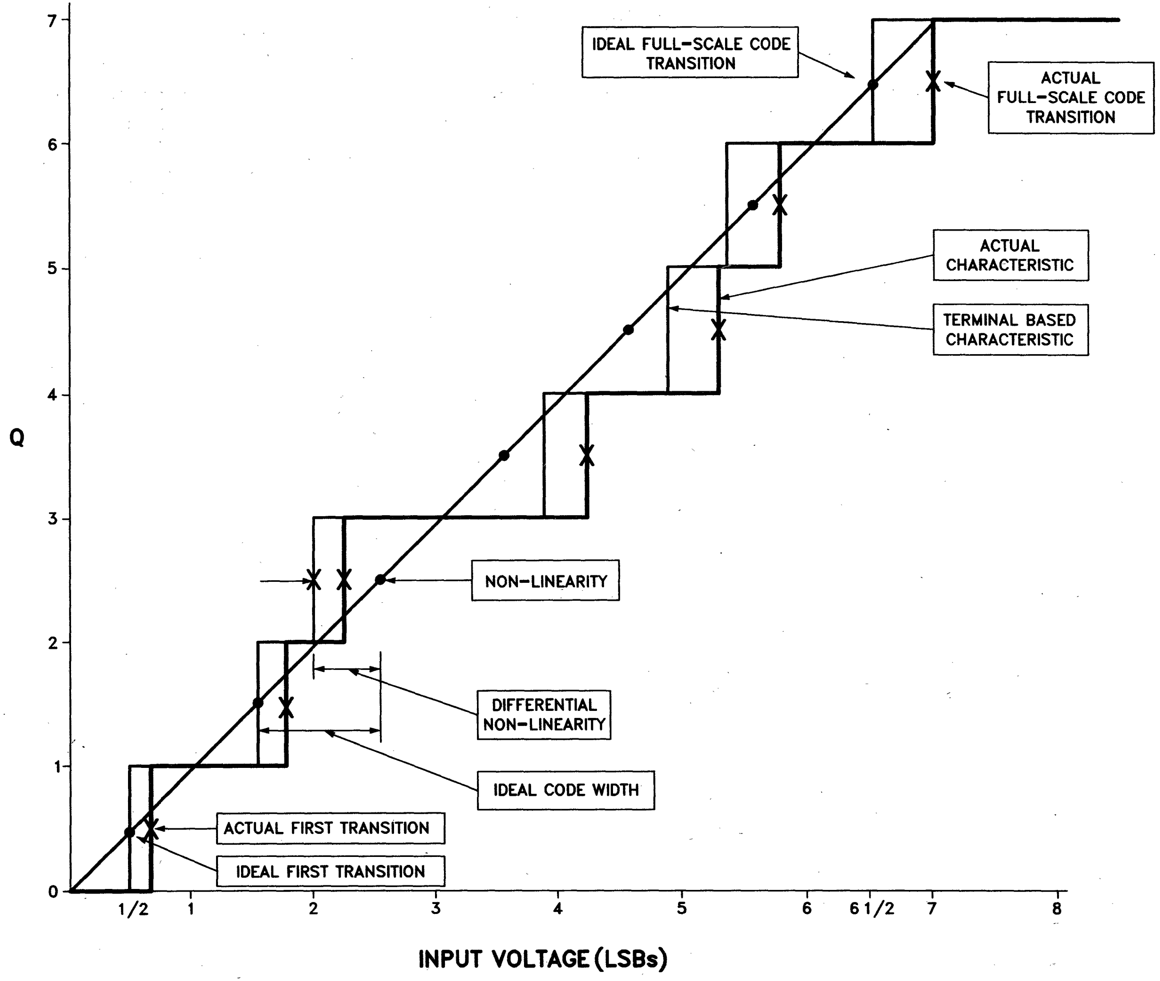
270246–16

Figure 15. Ideal A/D Characteristic



270246-17

Figure 16. Actual and Ideal Characteristics



270246-18

Figure 17. Terminal Based Characteristic

3.5 A/D Glossary of Terms

Figures 15, 16 and 17 display many of these terms.

ABSOLUTE ERROR—The maximum difference between corresponding actual and ideal code transitions. Absolute Error accounts for all deviations of an actual converter from an ideal converter.

ACTUAL CHARACTERISTIC—The characteristic of an actual converter. The characteristic of a given converter may vary over temperature, supply voltage, and frequency conditions. An Actual Characteristic rarely has ideal first and last transition locations or ideal code widths. It may even vary over multiple conversion under the same conditions.

BREAK-BEFORE-MAKE—The property of a multiplexer which guarantees that a previously selected channel will be deselected before a new channel is selected. (e.g. the converter will not short inputs together.)

CHANNEL-TO-CHANNEL MATCHING—The difference between corresponding code transitions of actual characteristics taken from different channels under the same temperature, voltage and frequency conditions.

CHARACTERISTIC—A graph of input voltage versus the resultant output code for an A/D converter. It describes the transfer function of the A/D converter.

CODE—The digital value output by the converter.

CODE CENTER—The voltage corresponding to the midpoint between two adjacent code transitions.

CODE TRANSITION—The point at which the converter changes from an output code of Q, to a code of Q+1. The input voltage corresponding to a code transition is defined to be that voltage which is equally likely to produce either of two adjacent codes.

CODE WIDTH—The voltage corresponding to the difference between two adjacent code transitions.

CROSSTALK—See "Off-Isolation".

D.C. INPUT LEAKAGE—Leakage current to ground from an analog input pin.

DIFFERENTIAL NON-LINEARITY—The difference between the ideal and actual code widths of the terminal based characteristic of a converter.

FEEDTHROUGH—Attenuation of a voltage applied on the selected channel of the A/D converter after the sample window closes.

FULL SCALE ERROR—The difference between the expected and actual input voltage corresponding to the full scale code transition.

IDEAL CHARACTERISTIC—A characteristic with its first code transition at $V_{IN} = 0.5$ LSB, its last code transition at $V_{IN} = (V_{REF} - 1.5$ LSB) and all code widths equal to one LSB.

INPUT RESISTANCE—The effective series resistance from the analog input pin to the sample capacitor.

LSB—LEAST SIGNIFICANT BIT: The voltage value corresponding to the full scale voltage divided by 2^n, where n is the number of bits of resolution of the converter. For a 10-bit converter with a reference voltage of 5.12 volts, one LSB is 5.0 mV. Note that this is different than digital LSBs, since an uncertainty of two LSB, when referring to an A/D converter, equals 10 mV. (This has been confused with an uncertainty of two digital bits, which would mean four counts, or 20 mV.)

MONOTONIC—The property of successive approximation converters which guarantees that increasing input voltages produce adjacent codes of increasing value, and that decreasing input voltages produce adjacent codes of decreasing value.

NO MISSED CODES—For each and every output code, there exists a unique input voltage range which produces that code only.

NON-LINEARITY—The maximum deviation of code transitions of the terminal based characteristic from the corresponding code transitions of the ideal characteristics.

OFF-ISOLATION—Attenuation of a voltage applied on a deselected channel of the A/D converter. (Also referred to as Crosstalk.)

REPEATABILITY—The difference between corresponding code transitions from different actual characteristics taken from the same converter on the same channel at the same temperature, voltage and frequency conditions.

RESOLUTION—The number of input voltage levels that the converter can unambiguously distinguish between. Also defines the number of useful bits of information which the converter can return.

SAMPLE DELAY—The delay from receiving the start conversion signal to when the sample window opens.

SAMPLE DELAY UNCERTAINTY—The variation in the Sample Delay.

SAMPLE TIME—The time that the sample window is open.

SAMPLE TIME UNCERTAINTY—The variation in the sample time.

SAMPLE WINDOW—Begins when the sample capacitor is attached to a selected channel and ends when the sample capacitor is disconnected from the selected channel.

SUCCESSIVE APPROXIMATION—An A/D conversion method which uses a binary search to arrive at the best digital representation of an analog input.

TEMPERATURE COEFFICIENTS—Change in the stated variable per degree centigrade temperature change. Temperature coefficients are added to the typical values of a specification to see the effect of temperature drift.

TERMINAL BASED CHARACTERISTIC—An Actual Characteristic which as been rotated and translated to remove zero offset and full-scale error.

VCC REJECTION—Attenuation of noise on the V_{CC} line to the A/D converter.

ZERO OFFSET—The difference between the expected and actual input voltage corresponding to the first code transition.

4.0 ANALOG OUTPUTS

Analog outputs can be generated by two methods, either by using the PWM output or the HSO. Either device will generate a rectangular pulse train that varies in duty cycle and (for the HSO only) period. If a smooth analog signal is desired as an output, the rectangular waveform must be filtered.

In most cases this filtering is best done after the signal is buffered to make it swing from 0 to 5 volts since both of the outputs are guaranteed only to TTL levels. A block diagram of the type of circuit needed is shown in Figure 18. By proper selection of components, accounting for temperature and power supply drift, a highly accurate 8-bit D to A converter can be made using either the HSO or the PWM output. Figure 19 shows two typical circuits. If the HSO is used the accuracy could be theoretically extended to 16-bits, however the temperature and noise related problems would be extremely hard to handle.

When driving some circuits it may be desirable to use unfiltered Pulse Width Modulation. This is particularly true for motor drive circuits. The PWM output can be used to generate these waveforms if a fixed period on the order of 64 μs is acceptable. If this is not the case then the HSO unit can be used. The HSO can generate a variable waveform with a duty cycle variable in up to 65536 steps and a period of up to 131 milliseconds. Both of these outputs produce TTL levels.

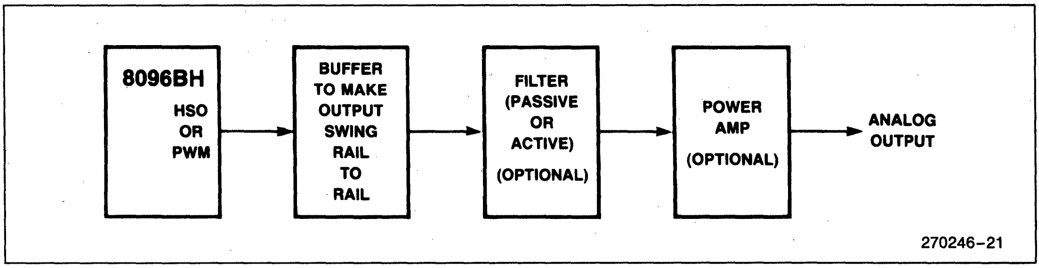
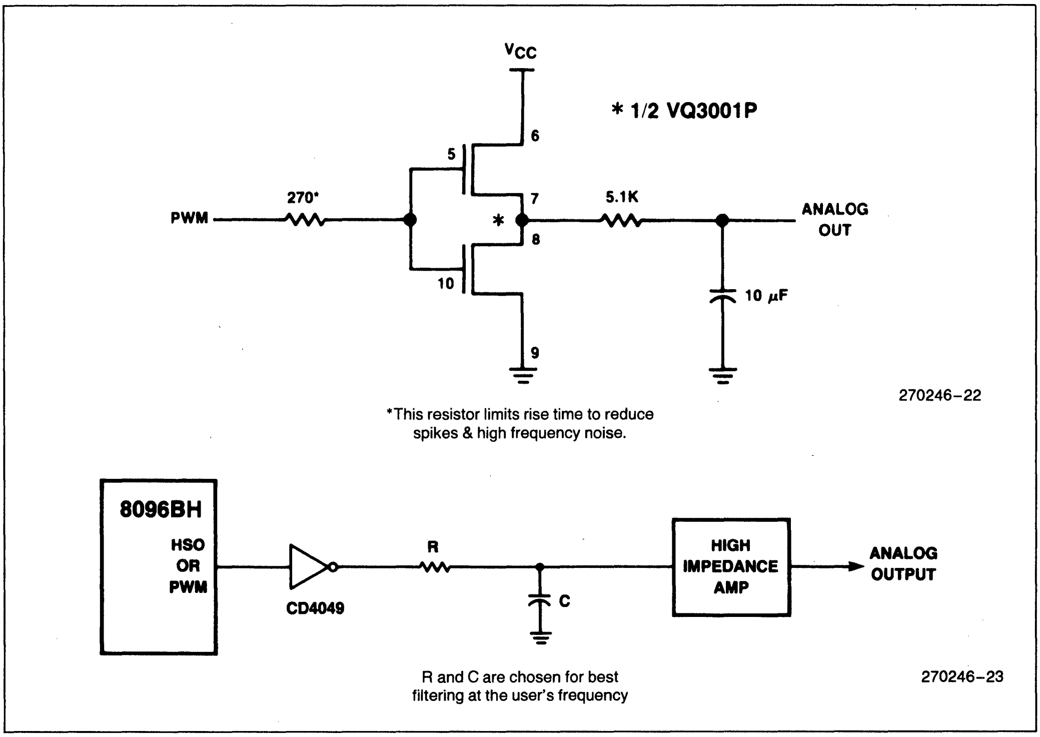


Figure 18. D/A Buffer Block Diagram



*This resistor limits rise time to reduce spikes & high frequency noise.

R and C are chosen for best filtering at the user's frequency

Figure 19. Buffer Circuits for D/A

5.0 I/O TIMINGS

The I/O pins on the 8096BH are sampled and changed at specific times within an instruction cycle. The changes occur relative to the internal phases shown in Figure 4. Note that the delay from XTAL1 to the internal clocks range from about 30 ns to 100 ns over process and temperature. Signals generated by internal phases are further delayed by 5 ns to 15 ns. The timings shown in this section are idealized; no propagation delay factors have been taken into account. Designing a system that depends on an I/O pin to change within a window of less than 50 ns using the information in this section is not recommended.

5.1 HSO Outputs

Changes in the HSO lines are synchronized to Timer 1. All of the external HSO lines due to change at a certain value of a timer will change just pior to the incrementing of Timer 1. This corresponds to an internal change during Phase B every eight state times. From an external perspective the HSO pin should change just prior to the rising edge of CLKOUT and be stable by its falling edge. Information from the HSO can be latched on the CLKOUT falling edge. Internal events can occur anytime during the 8 state time window.

Timer 2 is synchronized to increment no faster than Timer 1, so there will always be at least one incrementing of Timer 1 while Timer 2 is at a specific value.

5.2 HSI Input Sampling

The HSI pins are sampled internally once each state time. Any value on these pins must remain stable for at least 1 full state time to guarantee that it is recognized. The actual sample occurs at the end of Phase A, which, due to propagation delay, is just after the rising edge of CLKOUT. Therefore, if information is to be synchronized to the HSI it should be latched-in on CLKOUT falling. The time restriction applies even if the divide by eight mode is being used. If two events occur on the same pin within the same 8 state time window, only one of the events will be recorded. If the events occur on different pins they will always be recorded, regardless of the time difference. The 8 state time window, (i.e. the amount of time during which Timer 1 remains constant), is stable to within about 20 ns. The window starts roughly around the rising edge of CLKOUT, however this timing is very approximate due to the amount of internal circuitry involved.

5.3 Standard I/O Port Pins

Port 0 is different from the other digital ports in that it is actually part of the A/D converter. The port is sampled once every state time, however, sampling is not synchronized to Timer 1. If this port is used, the input signal on the pin must be stable one state time before the reading of the SFR.

Port 1 and Port 2 have quasi-bidirectional I/O pins. When used as inputs the data on these pins must be stable one state time prior to reading the SFR. This timing is also valid for the input-only pins of Port 2 and is similar to the HSI in that the sample occurs just after the rising edge of CLKOUT. When used as outputs, the quasi-bidirectional pins will change state shortly after CLKOUT falls. If the change was from '0' to a '1' the low impedance pullup will remain on for one state time after the change.

Ports 3 and 4 are addressed as off-chip memory-mapped I/O. The port pins will change state shortly after the rising edge of CLKOUT. When these pins are used as Ports 3 and 4 they are open drains, their structure is different when they are used as part of the bus. See Section 10.4 of the 8096BH Architecture chapter. Additional information on port reconstruction is available in Section 7.7 of this chapter.

6.0 SERIAL PORT TIMINGS

The serial port on the 8096BH was designed to be compatible with the 8051 serial port. Since the 8051 uses a divide by 2 clock and the 8096BH uses a divide by 3, the serial port on the 8096BH had to be provided with its own clock circuit to maximize its compatibility with the 8051 at high baud rates. This means that the serial port itself does not know about state times. There is circuitry which is synchronized to the serial port and to the rest of the 8096BH so that information can be passed back and forth.

The baud rate generator is clocked by either XTAL1 or T2CLK. Because T2CLK needs to be synchronized to the XTAL1 signal its speed must be limited to $\frac{1}{16}$ that of XTAL1. The serial port will not function during the time between the consecutive writes to the baud rate register. Section 11.4 of the 8096BH Architecture chapter discusses programming the baud rate generator.

6.1 Mode 0

Mode 0 is the shift register mode. The TXD pin sends out a clock train, while the RXD pin transmits or receives the data. Figure 20 shows the waveforms and timing. Note that the port starts functioning when a '0' is written to the REN (Receiver Enable) bit in the serial port control register. If REN is already high, clearing the RI flag will start a reception.

In this mode the serial port can be used to expand the I/O capability of the 8096BH by simply adding shift registers. A schematic of a typical circuit is shown in Figure 21. This circuit inverts the data coming in, so it must be reinverted in software. The enable and latch connections to the shift registers can be driven by decoders, rather than directly from the low speed I/O ports, if the software and hardware are properly designed.

6.2 Mode 1 Timings

Mode 1 operation of the serial port makes use of 10-bit data packages, a start bit, 8 data bits and a stop bit. The transmit and receive functions are controlled by separate shift clocks. The transmit shift clock starts when the baud rate generator is initialized, the receive shift clock is reset when a '1 to 0' transition (start bit) is received. The transmit clock may therefore not be in sync with the receive clock, although they will both be at the same frequency.

The TI (Transmit Interrupt) and RI (Receive Interrupt) flags are set to indicate when operations are complete. TI is set when the last data bit of the message has been sent, not when the stop bit is sent. If an attempt to send another byte is made before the stop bit is sent the port will hold off transmission until the stop bit is complete. RI is set when 8 data bits are received, not when the stop bit is received. Note that when the serial port status register is read both TI and RI are cleared.

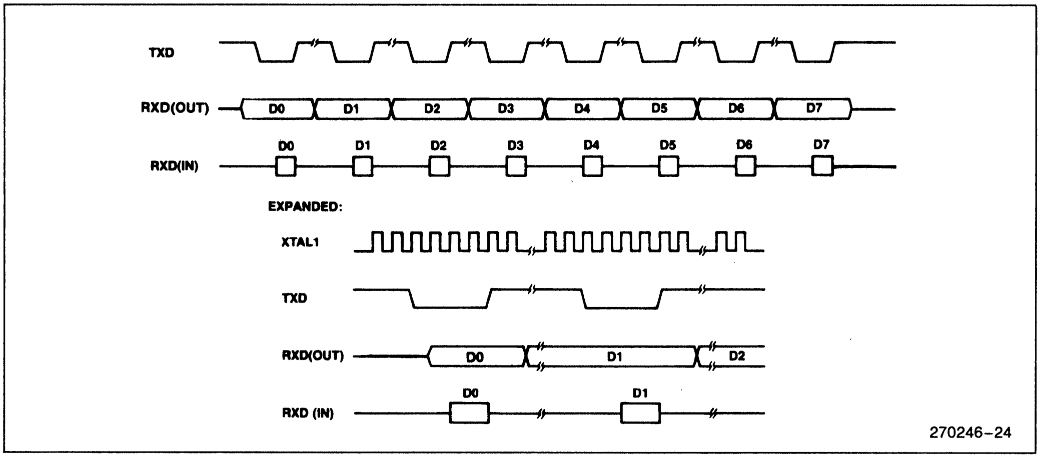


Figure 20. Serial Port Timings in Mode 0

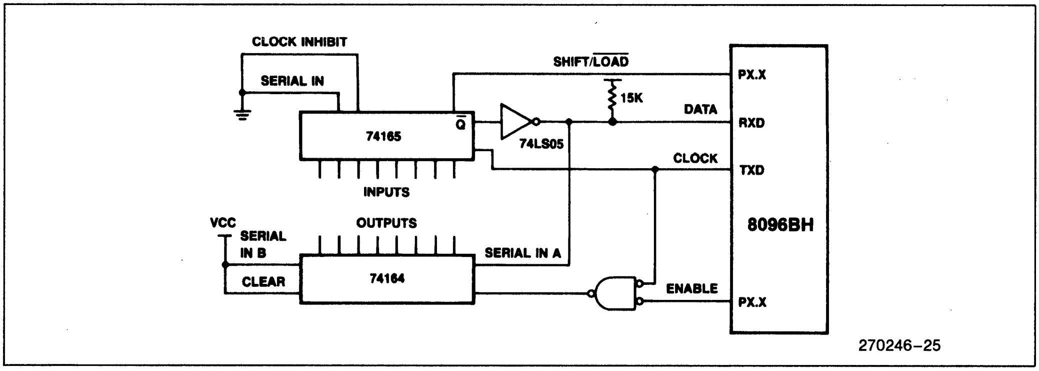


Figure 21. Mode 0 Serial Port Example

Caution should be used when using the serial port to connect more than two devices in half-duplex, (i.e. one wire for transmit *and* receive). If the receiving processor does not wait for one bit time after RI is set before starting to transmit, the stop bit on the link could be squashed. This could cause a problem for other devices listening on the link.

6.3 Mode 2 and 3 Timings

Modes 2 and 3 operate in a manner similar to that of Mode 1. The only difference is that the data is now made up of 9 bits, so 11-bit packages are transmitted and received. This means that TI and RI will be set on the 9th data bit rather than the 8th. The 9th bit can be used for parity or multiple processor communications (see Section 11 of the 8096BH Architecture chapter).

7.0 BUS TIMING AND MEMORY INTERFACE

7.1 Bus Functionality

The 8096BH has a multiplexed (address/data) bus which can be dynamically configured to have an 8-bit or 16-bit data width. There are control lines to demultiplex the bus (ALE or $\overline{\text{ADV}}$), indicate reads ($\overline{\text{RD}}$), indicate writes ($\overline{\text{WRL}}$ and $\overline{\text{WRH}}$, or $\overline{\text{WR}}$ with $\overline{\text{BHE}}$ and AD0), and a signal to indicate accesses that are for an instruction fetch (INST). Section 3.5 of the 8096BH Architecture chapter contains an overview of the bus operation.

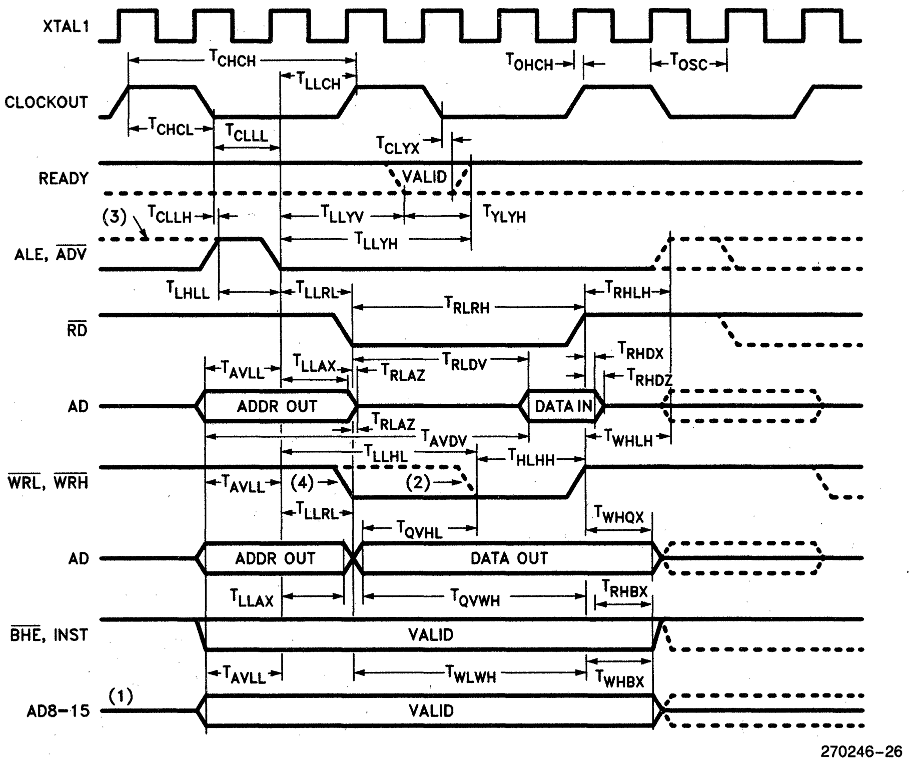
7.2 Timing Specifications

Figure 22 shows the timing of the bus signals and data lines. Please refer to the latest data sheet for the exact device you are using to ensure that your system is designed to the proper specifications. The major timing specifications are described in Figure 23.

7.3 READY Line Usage

When the processor has to address a memory location that cannot respond within the standard specifications, it is necessary to use the READY line to generate wait states. When the READY line is held low, the processor waits in a loop for the line to come high or until the number of inserted wait states is equal to the limit set in the Chip Configuration Register (see Section 2 of the MCS-96 Architecture chapter). There is a maximum time that the READY line can be held low without risking a processor malfunction due to dynamic nodes that have not been refreshed during the wait states. This time is shown as TYLYH in the data sheet.

In most cases the READY line is brought low after the address is decoded and it is determined that a wait state is needed. It is very likely that some addresses, such as those addressing memory mapped peripherals, would need wait states, and others would not. The READY line must be stable within the TLLYV specification after ALE falls or the processor could lock-up. There is



NOTES:
1. 8-bit bus only.
2. 8-bit or 16-bit bus and write strobe mode selected.
3. When ADV selected.
4. 8-bit or 16-bit bus and no write strobe mode selected.

Figure 22. Bus Signal Timings

no requirement as to when READY may go high, as long as the maximum READY low time (TYLYH) is not violated. To ensure that only one wait state is inserted it is necessary to provide external circuitry which brings READY high TLLYH after the falling edge of ALE/$\overline{\text{ADV}}$, or program the Chip Configuration Register to select a Ready Control limit of one.

Internally, the chip latches READY on the first falling edge of Phase A after ALE/$\overline{\text{ADV}}$ falls. Phase A is buffered and brought out externally as CLOCKOUT, so CLOCKOUT is a delayed Phase A. If a 1 is seen, the bus cycle proceeds uninterrupted with no wait state insertions. If a 0 is seen, one wait state (3 Tosc) is inserted.

If a wait state is inserted, READY is internally latched on the next rising edge of Phase A. If a 1 is found the bus cycle resumes with the net impact being the insertion of one wait state. If a 0 is seen, a second wait state is inserted.

The READY pin is again latched on the next rising edge of CLOCKOUT if two wait states were inserted. If the chip sees a 1, the bus cycle is resumed with the result being an insertion of two wait states. If another 0 is seen, a third wait state is inserted in the bus cycle and the READY pin is again latched on the following rising edge of CLOCKOUT. If internal Ready Control is not used, the READY line must at this point be a 1 to ensure proper operation.

Tosc—Oscillator Period, one cycle time on XTAL1.

Timings the Memory System Must Meet

TLLYH—ALE/$\overline{\text{ADV}}$ low to READY high: Maximum time after ALE/$\overline{\text{ADV}}$ falls until READY is brought high to ensure no more wait states. If this time is exceeded unexpected wait states may result. Nominally 1 Tosc + 3 Tosc × number of wait states desired.

TLLYV—ALE/$\overline{\text{ADV}}$ low to READY low: Maximum time after ALE/$\overline{\text{ADV}}$ falls until READY must be valid. If this time is exceeded the device could malfunction necessitating a chip reset. Nominally 2 Tosc periods.

TCLYX—READY hold after CLOCKOUT low: Minimum time that the value on the READY pin must be valid after CLOCKOUT falls. The minimum hold time is always zero nanoseconds.

TYLYH—READY low to READY high: Maximum time the part can be in the not-ready state. If it is exceeded, the 8096BH dynamic nodes which hold the current instruction may 'forget' how to finish the instruction.

TAVDV—ADDRESS valid to DATA valid: Maximum time that the memory has to output valid data after the 8096BH outputs a valid address. Nominally, a maximum of 5 Tosc periods.

TAVGV—ADDRESS valid to BUSWIDTH valid: Maximum time after ADDRESS becomes valid until BUSWIDTH must be valid. Nominally less than 2 Tosc periods.

TLLGV—ALE/$\overline{\text{ADV}}$ low to BUSWIDTH valid: Maximum time after ALE/$\overline{\text{ADV}}$ is low until BUSWIDTH must be valid. If this time is exceeded the part could malfunction necessitating a chip reset. Nominally less than 1 Tosc.

TLLGX—BUSWIDTH hold after ALE/$\overline{\text{ADV}}$ low: Minimum time that BUSWIDTH must be valid after ALE/$\overline{\text{ADV}}$ is low Nominally 1 Tosc.

TRLDV—$\overline{\text{READ}}$ low to DATA valid: Maximum time that the memory has to output data after $\overline{\text{READ}}$ goes low. Nominally, a maximum of 3 Tosc periods.

TRHDZ—$\overline{\text{READ}}$ high to DATA float: Time after $\overline{\text{READ}}$ is high until the memory must float the bus. The memory signal can be removed as soon as $\overline{\text{READ}}$ is not low, and must be removed within the specified maximum time from when $\overline{\text{READ}}$ is high. Nominally a maximum of 1 Tosc period.

TRHDX—DATA hold after $\overline{\text{READ}}$ goes high: Minimum time that memory must hold input DATA valid after $\overline{\text{RD}}$ is high. The hold time minimum is always zero nanoseconds.

TRLAZ—READ low to ADDRESS float: This is the bus control specifying the time from an active low READ signal until the ADDRESS drivers for the cycle must be off the bus. This is specified in order for data to be returned from the memory system without bus contention. Typically this is 0 ns for no bus contention. However, up to 10 ns is acceptable in systems.

Figure 23. Timing Specification Explanations

Timings the 8096 Will Provide

TOHCH—XTAL1 high to CLOCKOUT high: Delay from the rising edge of XTAL1 to the resultant rising edge on CLOCKOUT. Needed in systems where the signal driving XTAL1 is also used as a clock for external devices. Typically 50 to 100 nanoseconds.

TCHCH—CLKOUT high to CLKOUT high: The period of CLKOUT and the duration of one state time. Always 3 Tosc average, but individual periods could vary by a few nanoseconds.

TCHCL—CLKOUT high to CLKOUT low: Nominally 1 Tosc period.

TCLLH—CLKOUT low to ALE high: A help in deriving other timings. Typically plus or minus 5 ns to 10 ns.

TCLVL—CLOCKOUT low to ALE/$\overline{\text{ADV}}$ low: A help in deriving other timings. Nominally 1 Tosc.

TLLCH—ALE/$\overline{\text{ADV}}$ low to CLKOUT high: Used to derive other timings, nominally 1 Tosc period.

TLHLL—ALE/$\overline{\text{ADV}}$ high to ALE/$\overline{\text{ADV}}$ low: ALE/$\overline{\text{ADV}}$ high time. Useful in determining ALE/$\overline{\text{ADV}}$ rising edge to ADDRESS valid time. Nominally 1 Tosc period for ALE and 1 Tosc for $\overline{\text{ADV}}$ with back-to-back bus cycles.

TAVLL—ADDRESS valid to ALE/$\overline{\text{ADV}}$ low: Length of time ADDRESS is valid before ALE/$\overline{\text{ADV}}$ falls. Important timing for address latch circuitry. Nominally 1 Tosc period.

TLLAX—ALE/$\overline{\text{ADV}}$ low to ADDRESS invalid: Length of time ADDRESS is valid after ALE/$\overline{\text{ADV}}$ falls. Important timing for address latch circuitry. Nominally 1 Tosc period.

TLLRL—ALE/$\overline{\text{ADV}}$ low to $\overline{\text{READ}}$ or $\overline{\text{WRITE}}$ low: Length of time after ALE/$\overline{\text{ADV}}$ falls before $\overline{\text{RD}}$ or $\overline{\text{WR}}$ fall. Could be needed to ensure that proper memory decoding takes place before it is output enabled. Nominally 1 Tosc period.

TLLHL—ALE/$\overline{\text{ADV}}$ low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ low: Minimum time after ALE/$\overline{\text{ADV}}$ is low that the write strobe signals will go low. Could be needed to ensure that proper memory decoding takes place before it is output enabled. Nominally 2 Tosc periods.

TRLRH—$\overline{\text{READ}}$ low to $\overline{\text{READ}}$ high: $\overline{\text{RD}}$ pulse width, nominally 1 Tosc period.

TRHLH—$\overline{\text{READ}}$ high to ALE/$\overline{\text{ADV}}$ high: Time between $\overline{\text{RD}}$ going inactive and next ALE/$\overline{\text{ADV}}$, also used to calculate time between $\overline{\text{RD}}$ inactive and next ADDRESS valid. Nominally 1 Tosc period.

TRHBX—$\overline{\text{READ}}$ high to INST, $\overline{\text{BHE}}$, AD8–15 Inactive: Minimum time that the INST and BHE lines will be valid after $\overline{\text{RD}}$ goes high. Also the minimum time that the upper eight address lines (8-bit bus mode) will remain valid after $\overline{\text{RD}}$ goes high. Nominally 1 Tosc.

TWHBX—$\overline{\text{WRITE}}$ high to INST, $\overline{\text{BHE}}$, AD8–15 Inactive: Minimum time that the INST and $\overline{\text{BHE}}$ lines will be valid after $\overline{\text{WR}}$ goes high. Also the minimum time that the upper eight address lines (8-bit bus mode) will remain valid after $\overline{\text{WR}}$ goes high. Nominally 1 Tosc.

TWLWH—$\overline{\text{WRITE}}$ low to $\overline{\text{WRITE}}$ high: Write pulse width, nominally 3 Tosc periods.

THLHH—$\overline{\text{WRL}}$, $\overline{\text{WRH}}$ low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ high: Write strobe signal pulse width. Nominally 2 Tosc periods.

TQVHL—OUTPUT valid to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ low: Minimum time that OUTPUT data is valid prior to write strobes becoming active. Needed for interfacing to memories that read data on the falling edge of write. Nominally 1 Tosc.

TQVWH—OUTPUT valid to $\overline{\text{WRITE}}$ high: Time that the OUTPUT data is valid before $\overline{\text{WR}}$ is high. Nominally 3 Tosc periods.

TWHQX—$\overline{\text{WRITE}}$ high to OUTPUT not valid: Time that the OUTPUT data is valid after $\overline{\text{WR}}$ is high. Nominally 1 Tosc period.

TWHLH—$\overline{\text{WRITE}}$ high to ALE/$\overline{\text{ADV}}$ high: Time between write high and next ALE/$\overline{\text{ADV}}$, also used to calculate the time between $\overline{\text{WR}}$ high and next ADDRESS valid. Nominally 2 Tosc periods.

Figure 23. Timing Specification Explanations (Continued)

7.4 INST Line Usage

The INST (Instruction) line is high during bus cycles that are for an instruction fetch and low for any other bus cycle. The INST signal (not present on 48-pin versions) can be used with a logic analyzer to debug a system. In this way it is possible to determine if a fetch was for instructions or data, making the task of tracing the program much easier.

7.5 BUSWIDTH Pin Usage

The BUSWIDTH pin is a control input which determines the width of the bus access in progress. BUSWIDTH is sampled after the rising edge of the first CLOCKOUT after ALE/ADV goes low. If a one is seen, the bus access progresses as a 16-bit cycle. If a zero is seen, the bus access progresses as an 8-bit cycle. The BUSWIDTH setup and hold timing requirements appear in the data sheet.

The BUSWIDTH pin can be overridden by causing the BUS WIDTH SELECT bit in the Chip Configuration Register (CCR) to be zero. This will permanently select an 8-bit bus width. However, if the BUS WIDTH SELECT bit in the CCR is a one, the BUSWIDTH pin determines the bus width. See Section 3.5 of the 8096BH Architecture chapter. Since the BUSWIDTH pin is not available on 48-pin devices, the BUS WIDTH SELECT bit in the CCR determines bus width.

7.6 Address Decoding

The multiplexed bus of the 8096BH must be demultiplexed before it can be used. This can be done with two 74LS373 transparent latches for an 8096BH in 16-bit bus mode, or one 74LS373 for an 8096BH in 8-bit bus mode. As explained in Section 3.5 of the 8096BH Architecture chapter, the latched address signals will be referred to as MA0 through MA15 (Memory Address), and the data lines will be called MD0 through MD15 (Memory Data).

Since the 8096BH can make accesses to memory for either bytes or words, it is necessary to have a way of determining the type of access desired when the bus is 16-bits wide. For write cycles, the signals Write Low (WRL) and Write High (WRH) are provided. WRL will go low during all word writes and during all byte writes to an even location. Similarly, WRH will go low during all word writes and during all byte writes to an odd location. During read cycles, an 8096BH in 16-bit bus mode will always do a word read of an even location. If only one byte of the word is needed, the chip discards the byte it does not need.

Since 8096BH memory accesses over an 8-bit wide bus are always bytes, only one write strobe is needed for write cycles. For this purpose the WRL signal was made to go low for all write cycles during 8-bit bus accesses. When a word operation is requested, the bus controller performs two byte-wide bus cycles.

In many cases it may be desirable to have a write signal with a longer pulse width than WRL/WRH. The Write (WR) line of the 8096BH is an alternate control signal that shares a pin with WRL and is only available in 16-bit bus mode. WR is nominally one Tosc longer than the WRL/WRH signals, but goes low for any write cycle. Therefore it is necessary to decode for the type of write (byte or word) desired.

The Byte High Enable (BHE) signal and MA0 can be used for this purpose. BHE is an alternate control

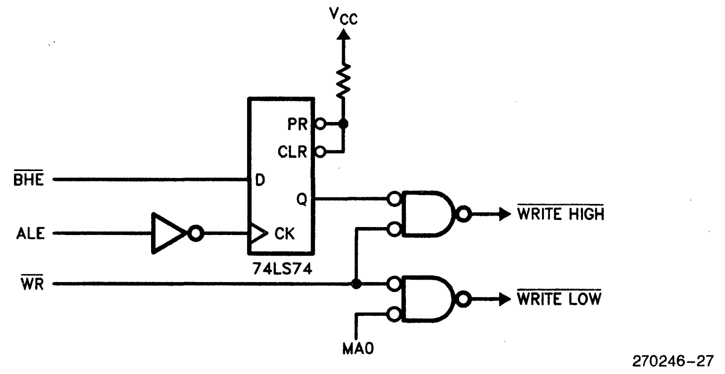


Figure 24. Decoding WR and BHE to Generate WriteLow and WriteHigh

signal that shares a pin with $\overline{WRH}$. When $\overline{BHE}$ is low, the high byte of the 16-bit bus is enabled. When MA0 is low, the lower byte is enabled. When MA0 is low and $\overline{BHE}$ is low, both bytes are enabled. Figure 24 shows how to use $\overline{WR}$, $\overline{BHE}$ and $\overline{MA0}$ to decode bus accesses. It's important to note that this decoding inserts a delay in the write signal which must be considered in a system timing analysis.

External memory systems for the 8096BH can be set up in many ways. Figures 25 through 28 show block diagrams of memory systems using an 8-bit bus with a single EPROM, using an 8-bit bus with RAM and EPROM, using a 16-bit bus with two external EPROMs and using a 16-bit bus in a RAM and ROM system.

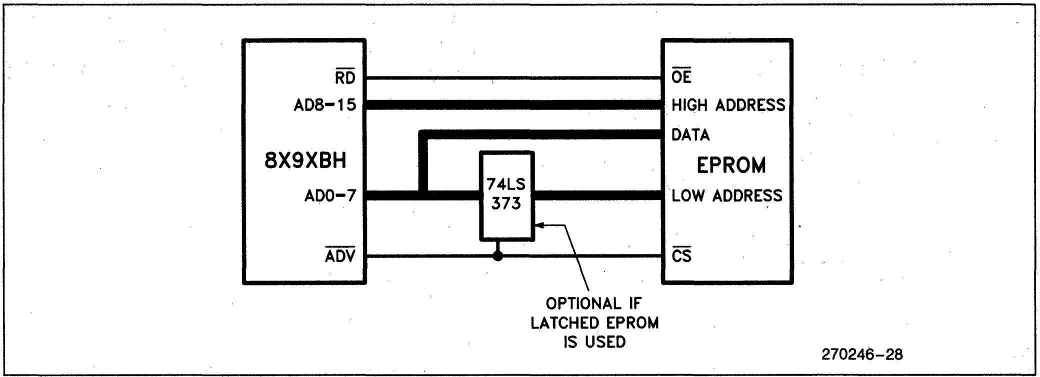


Figure 25. An 8-Bit Bus with EPROM Only

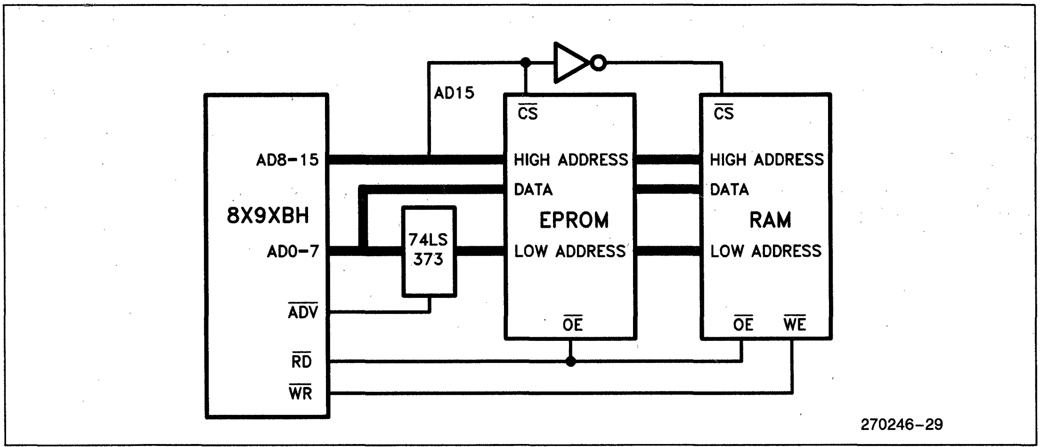


Figure 26. An 8-Bit Bus with EPROM and RAM

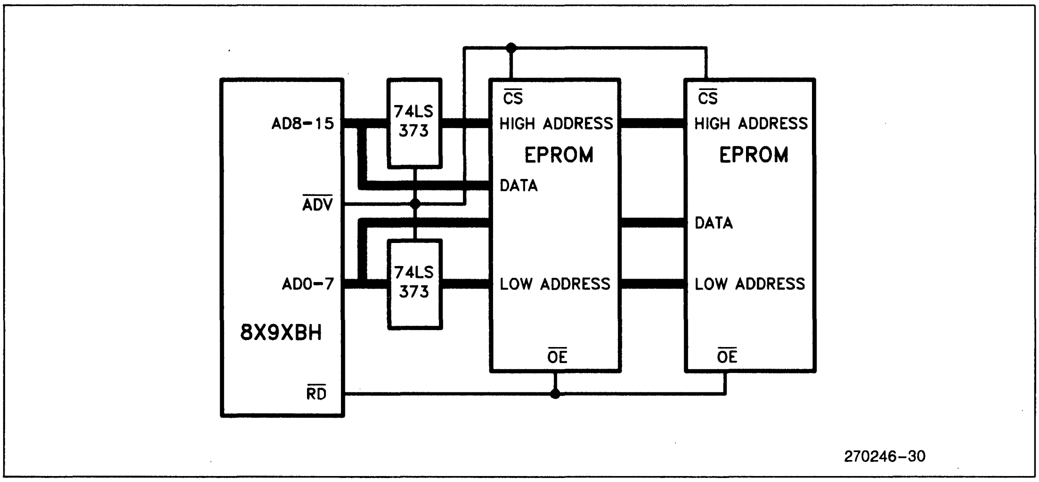


Figure 27. A 16-Bit Bus with EPROM Only

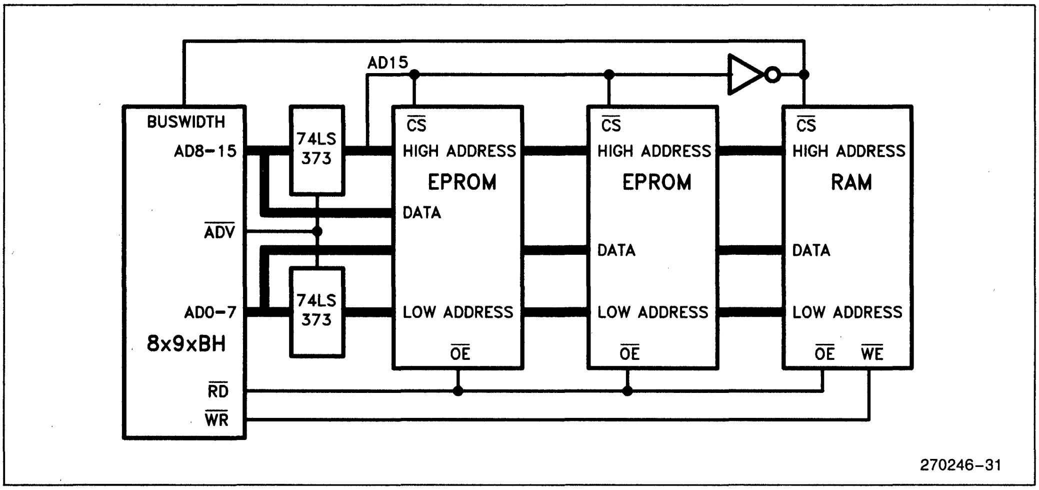


Figure 28. Memory System with Dynamic Bus Width

7.7 I/O Port Reconstruction

When a single-chip system is being designed using a multiple chip system as a prototype, it may be necessary to reconstruct I/O Ports 3 and 4 using a memory-mapped I/O technique. The circuit shown in Figure 30 provides this function. It can be attached to a 8096BH system which has the required address decoding and bus demultiplexing.

The output circuitry is basically just a latch that operates when 1FFEH or 1FFFH are placed on the MA lines. The inverters surrounding the latch create an open-collector output to emulate the open-drain output found on the 8096BH. The 'reset' line is used to set the ports to all 1's when the 8096BH is reset. It should be noted that the voltage and current characteristics of the port will differ from those of the 8096BH, but the basic functionality will be the same.

The input circuitry is just a bus transceiver that is addressed at 1FFEH or 1FFFH. If the ports are going to be used for either input or output, but not both, some of the circuitry can be eliminated.

8.0 NOISE PROTECTION TIPS

Designing controllers differs from designing other computer equipment in the area of noise protection. A microcontroller circuit under the hood of a car, in a photocopier, CRT terminal, or a high speed printer is subject to many types of electrical noise. Noise can get to the processor directly through the power supply, or it can be induced onto the board by electromagnetic fields. It is also possible for the PC board to find itself in the path of electrostatic discharges. Glitches and noise on the PC board can cause the processor to act unpredictably, usually by changing either the memory locations or the program counter.

There are both hardware and software solutions to noise problems, but the best solution is good design practice and a few ounces of prevention. The 8096BH has a Watchdog Timer which will reset the device if it fails to execute the software properly. The software should be set up to take advantage of this feature.

It is also recommended that unused areas of code be filled with NOPs and periodic jumps to an error routine or RST (reset chip) instructions. This is particularly important in the code around lookup tables, since if lookup tables are executed all sorts of bad things can happen. Wherever space allows, each table should be surrounded by 7 NOPs (the longest 8096BH instruction has 7 bytes) and a RST or jump to error routine instruction. This will help to ensure a speedy recovery should the processor have a glitch in the program flow.

Many hardware solutions exist for keeping PC board noise to a minimum. Ground planes, gridded ground and V_{CC} structures, bypass capacitors, transient absorbers and power busses with built-in capacitors can all be of great help. It is much easier to design a board with these features than to try to retrofit them later. Proper PC board layout is probably the single most important and, unfortunately, least understood aspect of project design. Minimizing loop areas and inductance, as well as providing clean grounds are very important. More information on protecting against noise can be found in the Application Note AP-125, "Designing Microcontroller Systems for Noisy Environments".

9.0 PACKAGING

The MCS-96 family of products is offered in many versions. They are available in 48-pin or 68-pin packages, with or without on-chip ROM/EPROM and with or without an A/D converter. A summary of the available options is shown in Figure 31.

The 48-pin versions are available in ceramic and plastic 48-pin Dual-In-Line package (DIP). The ceramic versions have order numbers with the prefix "C". The plastic versions have the prefix "P".

The 68-pin versions are available in a ceramic pin grid array (PGA), a plastic leaded chip carrier (PLCC) and a Type B leadless chip carrier (LCC). PGA devices have part numbers with the prefix "C". PLCC devices have the prefix "N". LCC devices have the prefix "R".

SHRINK-DIP is offered in 64-pin packages with a package designator of "U".

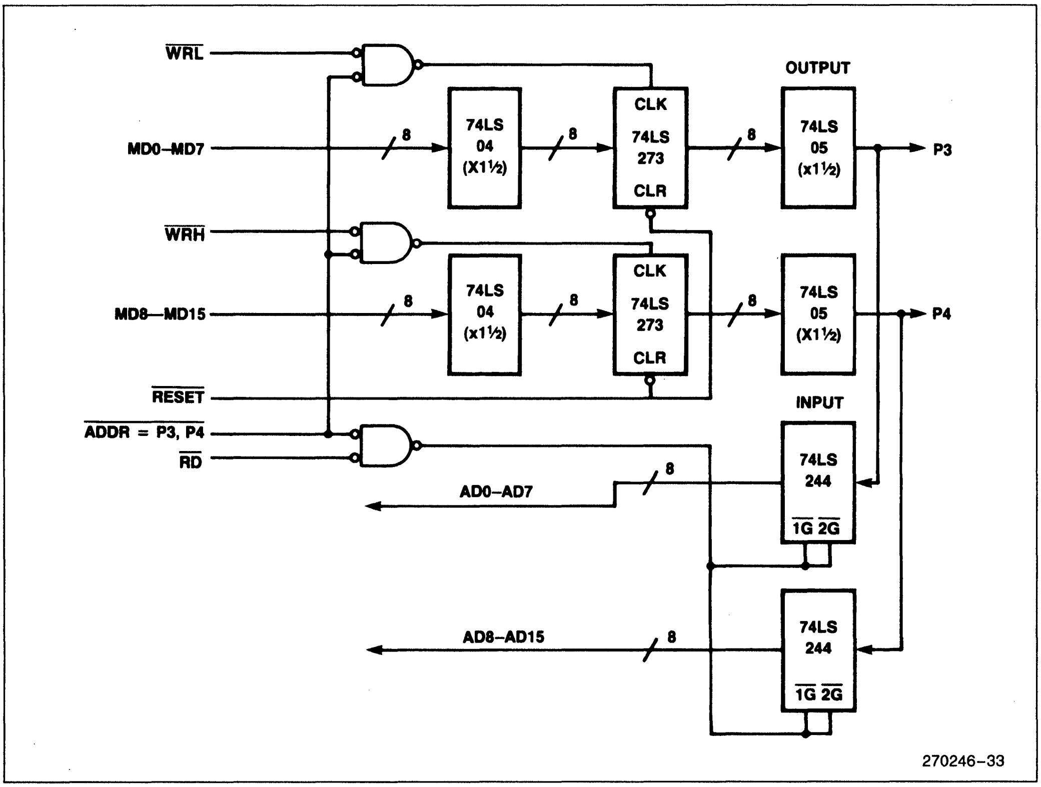


Figure 30. I/O Port Reconstruction

	Factory Masked ROM			CPU			User Programmable					
							EPROM			OTP		
	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin
ANALOG	8397BH 8397F	8397JF 8397BH	8395BH 8398	8097BH 8097JF	8097BH 8097JF	8095BH 8098	8797BH		8795BH 8798	8797BH 8797JF	8797JF 8797BH	8798
NO ANALOG	8396BH			8096BH								

Figure 31. HMOS MCS®96 Packaging

•48 Pin devices have four Analog Input pins.
•For ROM/OTP/EPROM devices, 8X9XBH and 8X98 = 8 KBytes, 8X9XJF = 16 KBytes
•64 Pin devices have all 48 Pin device features plus the following:
  Four additional Analog Input channels
  One additional Quasi-Bidirectional 8 Bit Parallel Port
  Four additional Port 2 pins with multiplexed features
    Timer 2 Clock Source Pin
    Timer 2 Reset pin
    Two additional quasi-bidirectional port pins

•68 Pin devices have all 48 and 64 pin features plus the following:
  Dynamic Buswidth sizing (8 or 16 bit bus)
  Dedicated System Clock Output (CLKOUT)
  INST pin for memory expansion
  Non-Maskable Interrupt for debugging
•Package Designators:
  N = PLCC
  C = Ceramic DIP
  A = Ceramic Pin Grid Array
  P = Plastic DIP
  R = Ceramic LCC
  U = Shrink DIP

10.0 EPROM PROGRAMMING

The 879XBH contains 8K bytes of ultraviolet Erasable and Electrically Programmable Read Only Memory (EPROM) for internal storage. This memory can be programmed in a variety of ways—including at run-time under software control.

The EPROM is mapped into memory locations 2000H through 3FFFH if $\overline{EA}$ is a TTL high. However, applying +12.75V to $\overline{EA}$ when the chip is reset will place the 879XBH in EPROM Programming Mode. The Programming Mode has been implemented to support EPROM programming and verification.

When an 879XBH is in Programming Mode, special hardware functions are available to the user. These functions include algorithms for slave, gang and auto EPROM programming.

10.1 Programming the 879XBH

Three flexible EPROM programming modes are available on the 879XBH—auto, slave and run-time. These modes can be used to program 879XBHs in a gang, stand alone or run-time environment.

The Auto Programming Mode enables an 879XBH to program itself, and up to 15 other 879XBHs, with the 8K bytes of code beginning at address 4000H on its external bus. The Slave Mode provides a standard interface that enables any number of 879XBHs to be programmed by a master device such as an EPROM programmer. The Run-Time Mode allows individual EPROM locations to be programmed at run-time under complete software control.

In the Programming Mode, some I/O pins have been renamed. These new pin functions are used to determine the programming function that is performed, provide programming ALEs, provide slave ID numbers and pass error information. Figure 33 shows how the pins are renamed. Figure 34 describes each new pin function.

While in Programming Mode, PMODE selects the programming function that is performed (see Figure 32). When not in the Programming Mode, Run-Time programming can be done at any time.

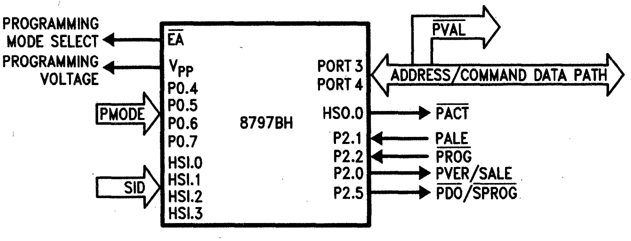
PMODE	Programming Mode
0–4	Reserved
5	Slave Programming
6–0BH	Reserved
0CH	Auto Programming Mode
0DH	Program Configuration Byte
0EH–0FH	Reserved

Figure 32. Programming Function PMODE Values

To guarantee proper execution, the pins of PMODE and SID must be in their desired state before the $\overline{RESET}$ pin is allowed to rise and reset the part. Once the device is reset, it is in the selected mode and should not be switched to another mode without a new reset sequence.

When $\overline{EA}$ selects the Programming Mode, the chip reset sequence loads the CCR from the Programming Chip Configuration Byte (PCCB). This is a separate EPROM location that is not mapped under normal operation. PCCB is only important when programming in the Auto Programming Mode. In this mode, the 879XBH that is being programmed gets the data to be programmed from external memory over the system bus. Therefore, PCCB must correctly correspond to the memory system in the programming setup, which is not necessarily the memory organization of the application.

The following sections describe 879XBH programming in each programming mode.



270246–34

Figure 33. Programming Mode Pin Function

Name	Function
PMODE	**PROGRAMMING MODE SELECT:** Determines the EPROM programming algorithm that is performed. PMODE is sampled after a chip reset and should be static while the device is operating.
SID	**SLAVE ID NUMBER:** Used to assign each slave pin of Port 3 or 4 to use for passing programming verification acknowledgement. For example, if gang programming in the Slave Programming Mode, the slave with SID = 0001 will use Port 3.1 to signal correct or incorrect program verification.
PALE	**PROGRAMMING ALE INPUT**: Accepted by an 879XBH that is in the Slave Programming Mode. Used to indicate that Ports 3 and 4 contain a command/ address.
$\overline{\text{PROG}}$	**PROGRAMMING PULSE:** Accepted by 879XBH that is in the Slave Programming Mode. Used to indicate that Ports 3 and 4 contain the data to be programmed. A falling edge on $\overline{\text{PROG}}$ signifies data valid and starts the programming cycle. A rising edge on $\overline{\text{PROG}}$ will halt programming in the slaves.
$\overline{\text{PACT}}$	**PROGRAMMING ACTIVE:** Used in the Auto-Programming Mode to indicate when programming activity is complete.
PVER	**PROGRAM VERIFIED:** A signal output after a programming operation by devices in the Slave Programming Mode and after programming in the Auto Configuration Byte Programming Mode. This signal is on Port 2.0 and is asserted as a logic 1 if the bytes program correctly.
$\overline{\text{PVAL}}$	**PROGRAM VALID:** These signals indicate the success or failure of programming in the Auto Programming Mode and when using this mode for gang programming. For the Auto Programming Mode this signal is asserted at Port 3.0. When using this mode for gang programming, all bits of Port 3 and Port 4 are asserted to indicate programming validity of the various slaves. A zero indicates successful programming on PVAL.0. A zero on PVAL.1 through PVAL.15 indicates a fail.
$\overline{\text{PDO}}$	**PROGRAMMING DURATION OVERFLOWED:** A signal output by devices in the Slave Programming Mode. Used to signify that the $\overline{\text{PROG}}$ pulse applied for a programming operation was longer than allowed.
SALE	**SLAVE ALE:** Output signal from an 879XBH in the Auto Programming Mode. A falling edge on SALE indicates that Ports 3 and 4 contain valid address/command information for slave 879XBHs that may be attached to the master.
$\overline{\text{SPROG}}$	**SLAVE PROGRAMMING PULSE:** Output from an 879XBH in the Auto Programming Mode. A falling edge on $\overline{\text{SPROG}}$ indictates that Ports 3 and 4 contain valid data for programming into slave 879XBHs that may be attached to the master.
PORTS 3 and 4	**ADDRESS/COMMAND/DATA BUS:** Used to pass commands, addresses and data to and from slave mode 879XBHs. Used by chips in the Auto Programming Mode to pass command, addresses and data to slaves. Also used in the Auto Programming Mode as a regular system bus to access external memory. Each line should be pulled up to VCC through a resistor.

Figure 34. Programming Mode Pin Definitions

10.2 Auto Programming Mode

The Auto Programming Mode provides the ability to program the internal 879XBH EPROM without having to use a special EPROM programmer. In this mode, the 879XBH simply programs itself with the data found at external locations 4000H through 5FFFH. All that is required is that some sort of external memory reside at these locations, that $\overline{EA}$ selects the programming mode and that V_{PP} is applied. Figure 35 shows a minimum configuration for using an 8K x 8 EPROM to program one 879XBH in the Auto Programming Mode.

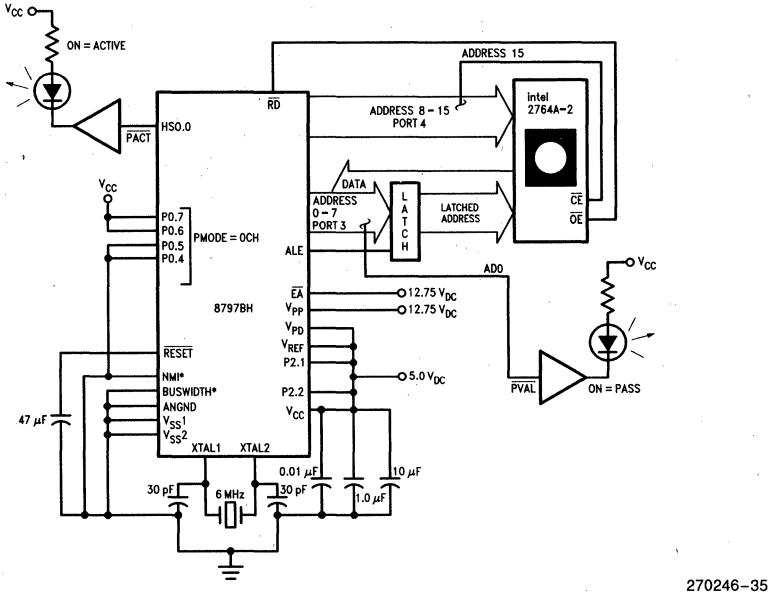
The 879XBH first reads a word from external memory, then the Modified Quick-Pulse Programming™ Algorithm (described later) is used to program the appropriate EPROM location. Since the erased state of a byte is 0FFH, the Auto Programming Mode will skip locations where the data to be programmed is 0FFH. When all 8K has been programmed, $\overline{PACT}$ goes high and the devices outputs a 0 on Port 3.0 ($\overline{PVAL}$) if it programmed correctly and a 1 if it failed.

10.2.1 GANG PROGRAMMING WITH THE AUTO PROGRAMMING MODE

An 879XBH in the Auto Programming Mode can also be used as a programmer for up to 15 other 879XBHs that are configured in the Slave Programming Mode. To accomplish this, the 879XBH acting as the master outputs the slave command/data pairs on Ports 3 and 4 necessary to program slave devices with the same data it is programming itself with. Slave ALE (SALE) and Slave $\overline{PROG}$ ($\overline{SPROG}$) signals are provided by the master to the slaves to demultiplex the commands from the data. Figure 36 is a block diagram of a gang programming system using one 879XBH in the Auto Programming Mode. The Slave Programming Mode is described in the next section.

The master 879XBH first reads a word from the external memory controlled by ALE, $\overline{RD}$ and $\overline{WR}$. It then drives Ports 3 and 4 with a Data Program command using the appropriate address and alerts the slaves with a falling edge on SALE. Next, the data to be programmed is driven onto Ports 3 and 4 and slave programming begins with a falling edge on $\overline{SPROG}$. At the same time, the master begins to program its own EPROM location with the data read in. Intel's Modified Quick-Pulse Programming™ Algorithm is used, with Data Verify commands being given to the slaves after each programming pulse.

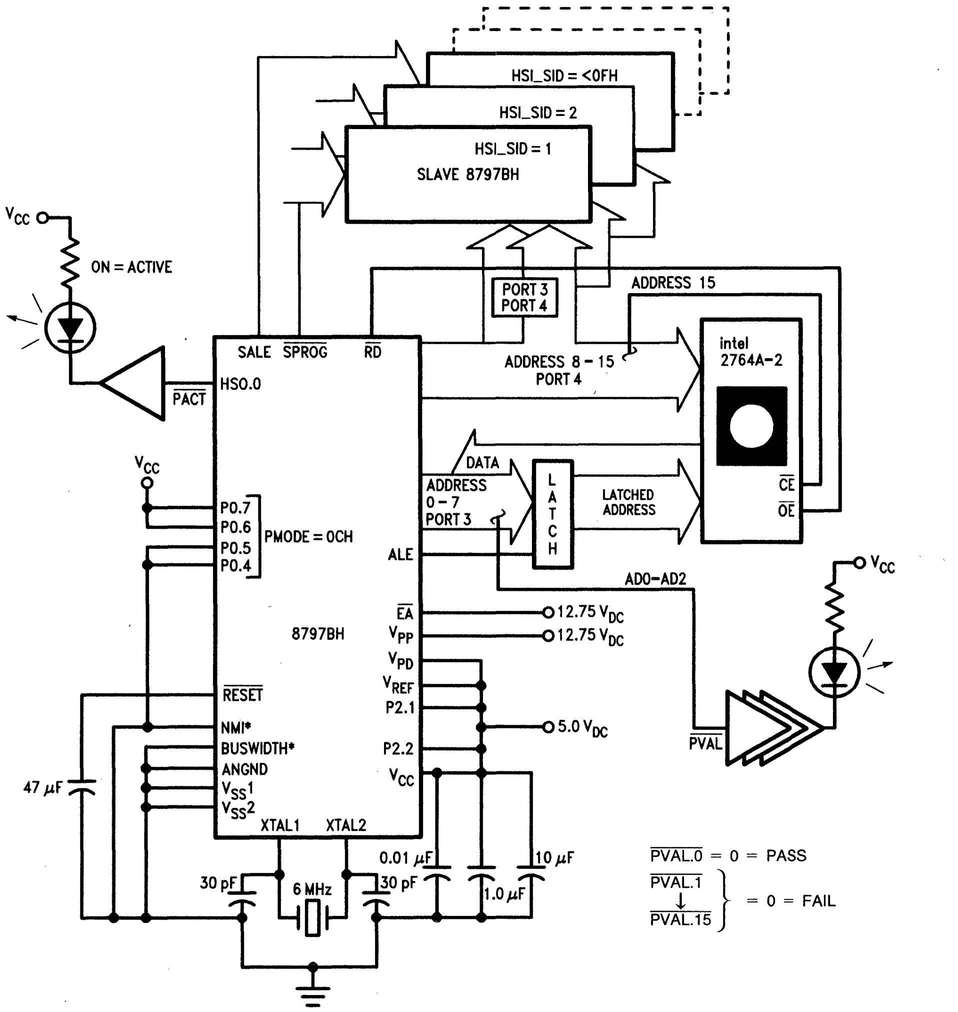
When programming is complete $\overline{PACT}$ goes high and Ports 3 and 4 are driven with all 1s if all devices programmed correctly. Individual bits of Port 3 and 4 will be driven to 0 if the slave with that bit number as an SID did not program correctly. The 879XBH used as the master assigns itself an SID of 0.



270246-35

NOTE:
*Ports 3 and 4 should have pullups to VCC.

Figure 35. The Auto Programming Mode



NOTE:
*EA and VPP on slaves must be at +12.75 Vdc. Each slave's PMODE must equal 05H. Ports 3 and 4 should have pullups to VCC. Minimum configuration connections must also be made for slaves. A 10 MHz clock is recommended for the slaves.

Figure 36. Gang Programming with the Auto Programming Mode

10.3 Slave Programming Mode

Any number of 879XBHs can be programmed by a master programmer through the Slave Programming Mode.

The programming device uses Ports 3 and 4 of the devices being programmed as a command/data path. The slaves accept signals on PALE (Program ALE) and $\overline{\text{PROG}}$ (Program Enable) to demultiplex the commands and data. The slaves also use PVER, $\overline{\text{PDO}}$ and Ports 3 and 4 to pass error information to the programmer. Support for gang programming of up to 16 879XBHs is provided. If each device art is given a unique SID (Slave ID Number) an 879XBH in the Auto Programming Mode can be used as a master to program itself and up to 15 other slave 879XBHs. There is, however, no 879XBH dependent limit to the number of devices that can be gang programmed in the slave mode.

It is important to note that the interface to an 879XBH in the slave mode is similar to a multiplexed bus. Attempting to issue consecutive PALE pulses without a corresponding $\overline{\text{PROG}}$ pulse will produce unexpected results. Similarly, issuing consecutive $\overline{\text{PROG}}$ pulses without the corresponding PALE pulses immediately preceding is equally unpredictable.

10.3.1 SLAVE PROGRAMMING COMMANDS

The commands sent to the slaves are 16-bits wide and contain two fields. Bits 14 and 15 specify the action that the slaves are to perform. Bits 0 through 13 specify the address upon which the action is to take place. Commands are sent via Ports 3 and 4 and are available to cause the slaves to program a word, verify a word, or dump a word (Table 1). The address part of the command sent to the slaves ranges from 2000H to 3FFFH and refers to the internal EPROM memory space. The following sections describe each slave programming mode command.

Table 1. Slave Programming Mode Commands

P4.7	P4.6	Action
0	0	Word Dump
0	1	Data Verify
1	0	Data Program
1	1	Reserved

DATA PROGRAM COMMAND—After a Data Program Command has been sent to the slaves, $\overline{\text{PROG}}$ must be pulled low to cause the data on Ports 3 and 4 to be programmed into the location specified during the command. The falling edge of $\overline{\text{PROG}}$ is not only used to indicate data valid, but also triggers the hardware programming of the word specified. The slaves will begin programming 48 states after $\overline{\text{PROG}}$ falls, and will continue to program the location until $\overline{\text{PROG}}$ rises.

After the rising edge of $\overline{\text{PROG}}$, the slaves automatically perform a verification of the address just programmed. The result of this verification is then output on PVER (Program Verify) and $\overline{\text{PDO}}$ (Program Duration Overflowed). Therefore, verification information is available following the Data Program Command for programming systems that cannot use the Data Verify command.

If PVER and $\overline{\text{PDO}}$ of all slaves are 1s after $\overline{\text{PROG}}$ rises then the data program was successful everywhere. If PVER is a 0 in any slave, then the data programmed did not verify correctly in that device. If $\overline{\text{PDO}}$ is a 0 in any slave, then the programming pulse in those devices was terminated by an internal safety feature rather than the rising edge of $\overline{\text{PROG}}$. The safety feature prevents over-programming in the slave mode. Figure 37 shows the relationship of PALE, $\overline{\text{PROG}}$, PVER and $\overline{\text{PDO}}$ to the Command/Data Path on Ports 3 and 4 for the Data Program Command.

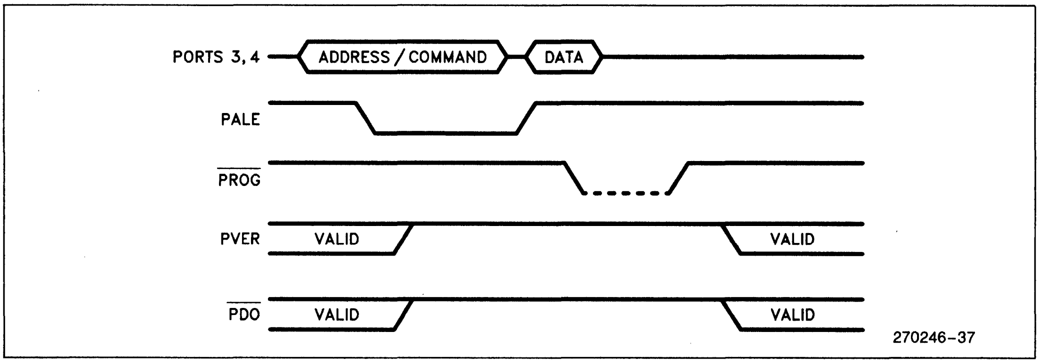


Figure 37. Data Program Signals in Slave Programming Mode

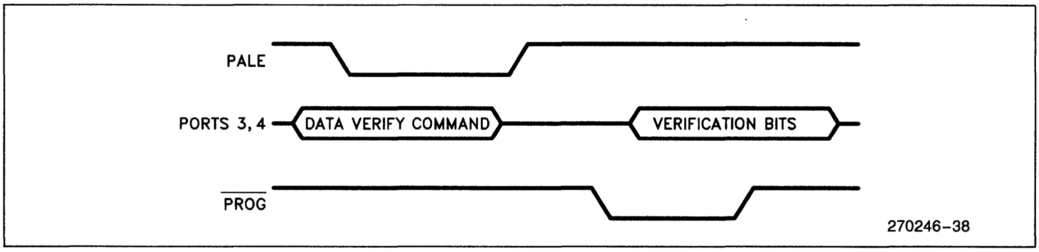


Figure 38. Data Verify Command Signals

DATA VERIFY COMMAND—When the Data Verify Command is sent, the slaves respond by driving one bit of Port 3 and 4 to indicate correct or incorrect verification of the previous Data Program. A 1 indicates correct verification, while a 0 indicates incorrect verification. The SID (Slave ID Number) of each slave determines which bit of the command/data path is driven. $\overline{\text{PROG}}$ from the programmer governs when the slaves drive the bus. Figure 38 shows the relationship of Ports 3 and 4 to PALE and $\overline{\text{PROG}}$.

This command is always preceded by a Data Program Command in a programming system with as many as 16 slaves. However, a Data Verify Command does not have to follow every Data Program Command.

WORD DUMP COMMAND—When the Word Dump Command is issued, the 879XBH being programmed adds 2000H to the address field of the command and places the value found at the new address on Ports 3 and 4. For example, sending the command #0100H to a slave will result in the slave placing the word found at location 2100H on Ports 3 and 4. $\overline{\text{PROG}}$ from the programmer governs when the slave drives the bus. The signals are the same as shown in Figure 22.

Note that this command will work only when just one slave is attached to the bus, and that there is no restriction on commands that precede or follow a Word Dump Command.

10.3.2 GANG PROGRAMMING WITH THE SLAVE PROGRAMMING MODE

Gang programming of 879XBHs can be done using the Slave Programming Mode. There is no 879XBH based limit on the number of chips that may be hooked to the same Port 3/Port 4 data path for gang programming.

If more than 16 chips are being gang programmed, the PVER and $\overline{\text{PDO}}$ outputs of each chip could be used for verification. The master programmer could issue a data program command then either watch every chip's error signals, or AND all the signals together to get a system PVER and $\overline{\text{PDO}}$.

If 16 or fewer 879XBHs are to be gang programmed at once, a more flexible form of verification is available. By giving each chip being programmed a unique SID, the master programmer could then issue a data verify command after the data program command. When a verify command is seen by the slaves, each will drive one pin of Port 3 or 4 with a 1 if the programming verified correctly or a 0 if programming failed. The SID is used by each slave to determine which Port 3, 4 bit it is assigned. An 879XBH in the Auto Programming Mode could be the master programmer if 15 or fewer slaves need to be programmed (see Gang Programming with the Auto Programming Mode).

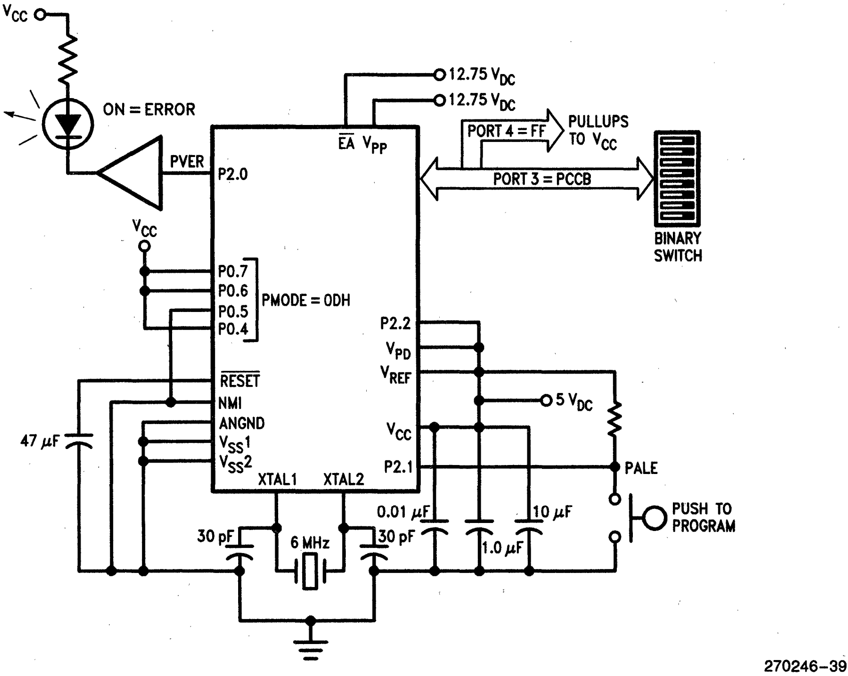
10.4 Auto Configuration Byte Programming Mode

The CCB (location 2018H) can be treated just like any other EPROM location, and programmed using any programming mode. But to provide for simple programming of the CCB when no other locations need to be programmed, the Auto Configuration Byte Programming Mode is provided. Programming in this mode also programs PCCB. Figure 39 shows a block diagram for using the Auto Configuration Byte Programming Mode.

With PMODE = 0DH and 0FFH on Port 4, CCB and PCCB will be programmed to the value on Port 3 when a logic 0 is placed on PALE. After programming is complete, PVER will be driven to a 1 if the bytes programmed correctly, and a 0 if the programming failed.

This method of programming is the only way to program PCCB. PCCB is a non-memory mapped EPROM location that gets loaded into CCR during the reset sequence when the voltage on $\overline{EA}$ puts the 879XBH in Programming Mode. If PCCB is not programmed using the Auto Configuration Byte Programming Mode, every time the 879XBH is put into Programming Mode the CCR will be loaded with 0FFH (the value of the erased PCCB location).

However, if programming the CCB and PCCB is done using this Programming Mode, the PCCB will take on the value programmed into CCB. This means that until the device is erased, programming activities that use the system bus will employ the bus width and controls selected by the user's CCB.



NOTES:
1. Tie Port 3 to the value desired to be programmed into CCB, and PCCB.
2. Make all necessary minimum connections for power, ground and clock.

Figure 39. The Auto CCR Programming Mode

10.5 Run-Time Programming

Run-Time Programming of the 879XBH is provided to allow the user complete flexibility in the ways in which the internal EPROM is programmed. That flexibility includes the ability to program just one byte or one word instead of the whole EPROM, and extends to the hardware necessary to program. The only additional requirement of a system is that a programming voltage is applied to VPP. Run-Time Programming is done with $\overline{EA}$ at TTL-high (normal operation—internal/external access).

To Run-Time program, the user writes a byte or word to the location to be programmed. Once this is done, the 879XBH will continue to program that location until another data read from or data write to the EPROM occurs. The user can therefore control the duration of the programming pulse to within a few mircoseconds. An intelligent algorithm should be implemented in software. It is recommended that the Modified Quick-Pulse Programming Algorithm be implemented.

After the programming of a location has started, care must be taken to ensure that no program fetches (or pre-fetches) occur from internal memory. This is of no concern if the program is executing from external memory. However, if the program is executing from internal memory when the write occurs, it will be necessary to use the built in "Jump to Self" located at 201AH.

"Jump to Self" is a two byte instruction in the Intel test ROM which can be CALLed after the user has started programming a location by writing to it. A software timer interrupt could then be used to escape from the "Jump to Self" when the proper programming pulse duration has elapsed. Figure 40 is an example of how to program an EPROM location while execution is entirely internal.

Upon entering the PROGRAM routine, the address and data are retrieved from the STACK and a Software Timer is set to expire one programming pulse later. The data is then written to the EPROM location and a CALL to location 201AH is made. Location 201AH is in Intel reserved test ROM, and contains the two byte opcode for a "Jump to Self". The minimum interrupt service routine would remove the 201AH return address from the STACK and return.

```
PROGRAM:

    POP  temp                                         ;take parameters from the
                                                       STACK
    POP  address_temp
    POP  data_temp
    PUSH temp

    PUSHF                                             ;save current status
    LDB  int_mask , #enable_swt_only                  ;enable only swt interrups
    LDB  HSO_COMMAND , #SWTO_ovf                      ;load swt command to interrupt
    ADD  HSO_TIME,TIMER1, #program_pulse              ;when program pulse time
                                                      ;has elapsed
    EI
    ST   data-temp, [address_temp]
    CALL 201AH
    POPF
    RET

SWT_ISR:
    • • •

swt0_expired:
    POP  0
    RET
    • • •
```

Figure 40. Programming the EPROM from Internal Memory Execution

10.6 ROM/EPROM Program Lock

Protection mechanisms have been provided on the ROM and EPROM versions of the 8096BH to inhibit unauthorized accesses of internal program memory. However, there must always be a way to allow authorized program memory dumps for testing purposes. The following describes 839XBH, 879XBH program lock features and the mode provided for authorized memory dumps.

10.6.1 LOCK FEATURES

Write protection is provided for EPROM devices while READ protection is provided for both ROM and EPROM devices.

Write protection is enabled by causing the LOC0 bit in the CCR to take the value 0. When WRITE protection is selected, the bus controller will cycle through the write sequence, but will not actually drive data to the EPROM and will not enable V_{PP} to the EPROM. This protects the entire EPROM 2000H–3FFFH from inadvertant or unauthorized programming, and also prevents writes to the EPROM from upsetting program execution. If write protection is not enabled, a data write to an internal EPROM location will begin programming that location, and continue programming the location until a data read of the internal EPROM is executed. While programming, instruction fetches from internal EPROM will not be successful.

READ protection is selected by causing the LOC1 bit in the CCR to take the value 0. When READ protection is enabled, the bus controller will only perform a data read from the address range 2020H–3FFFH if the slave program counter is in the range 2000H–3FFFH. Note that since the slave PC can be many bytes ahead of the CPU program counter, an instruction that is located after address 3FFAH may not be allowed to access protected memory, even though the instruction is itself protected.

If the bus controller receives a request to perform a READ of protected memory, the READ sequence occurs with indeterminant data being returned to the CPU.

Other enhancements were also made to the 8096BH for program protection. For example, the value of $\overline{\text{EA}}$ is latched on reset so that the device cannot be switched from external to internal execution mode at run-time. In addition, if READ protection is selected, an NMI event will cause the device to switch to external only execution mode. Internal execution can only resume by resetting the chip.

10.6.2 AUTHORIZED ACCESS OF PROTECTED MEMORY

To provide a method of dumping the internal ROM/EPROM for testing purposes a "Security Key" mechanism and ROM dump mode have been implemented.

The security key is a 128 bit number, located in internal memory, that must be matched before a ROM dump will occur. The application code contains the security key starting at location 2020H.

The ROM dump mode is entered just like any programming mode ($\overline{\text{EA}}$ = 12.75V), except that a special PMODE strapping is used. The PMODE for ROM dump is 6H (0110B).

The ROM dump sequence begins with a security key verification. Users must place at external locations 4020H–402FH the same 16 byte key that resides inside the chip at locations 2020H–202FH. Before doing a ROM dump, the chip checks that the keys match.

After a successful key verification, the chip dumps data to external locations 1000H–11FFH and 4000H–5FFFH. Unspecified data appears at the low addresses.

Internal EPROM/ROM is dumped to 4000H–5FFFH, beginning with internal address 2000H.

If a security key verification is not successful, the chip will put itself into an endless loop of internal execution.

NOTE:

Substantial effort has been expended to provide an excellent program protection scheme. However, Intel cannot, and does not guarantee that the protection methods that we have devised will prevent unauthorized access.

10.7 Modified Quick-Pulse Programming™ Algorithm

The Modified Quick-Pulse Programming Algorithm calls for each EPROM location to receive 25 separate 100 μs (±5 μs) program cycles. Verification of correct programming is done after the 25 pulses. If the location verifies correctly, the next location is programmed. If the location fails to verify, the location has failed.

Once all locations are programmed and verified, the entire EPROM is again verified.

Programming of 879XBH devices is done with V_{PP} = 12.75V ±0.25V and V_{CC} = 5.0V ±0.5V.

10.8 Signature Word

The 8X9XBH contains a signature word at location 2070H. The word can be accessed in the slave mode by executing a word dump command.

Table 2. 8X9XBH Signature Words

Device	Signature Word
879XBH	896FH
839XBH	896EH
809XBH	Undefined

10.9 Erasing the 879XBH EPROM

Initially, and after each erasure, all bits of the 879XBH are in the "1" state. Data is introduced by selectively programming "0s" into the desired bit locations. Although only "0s" will be programmed, both "1s" and "0s" can be present in the data word. The only way to change a "0" to a "1" is by ultraviolet light erasure.

The erasure characteristics of the 879XBH are such that erasure begins to occur upon exposure to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000–4000 Å range. Constant exposure to room level fluorescent lighting could erase the typical 879XBH in approximately 3 years, while it would take approximately 1 week to cause erasure when exposed to direct sunlight. If the 879XBH is to be exposed to light for extended periods of time, opaque labels must be placed over the EPROM's window to prevent unintentional erasure.

The recommended erasure procedure for the 879XBH is exposure to shortwave ultraviolet light which has a wavelength of 2537Å. The integrated dose (i.e., UV intensity × exposure time) for erasure should be a minimum of 15 Wsec/cm^2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with a 12000 μW/cm^2 power rating. The 879XBH should be placed within 1 inch of the lamp tubes during erasure. The maximum integrated dose an 879XBH can be exposed to without damage is 7258 Wsec/cm^2 (1 week @ 12000 μW/cm^2). Exposure of the 879XBH to high intensity UV light for long periods may cause permanent damage.

11.0 QUICK REFERENCE

11.1 Pin Description

On the 48-pin devices the following pins are not bonded out: Port1, Port0 (Analog In) bits 0–3, T2CLK (P2.3), T2RST (P2.4), P2.6, P2.7, CLKOUT, INST, NMI, BUSWIDTH. S-DIP packages do not have INST, CLKOUT, BUSWIDTH or NMI.

PIN DESCRIPTIONS

Symbol	Name and Function
V_{CC}	Main supply voltage (5V).
V_{SS}	Digital circuit ground (0V). Two pins.
V_{PD}	RAM standby supply voltage (5V). This voltage must be present during normal operation. In a Power Down condition (i.e. V_{CC} drops to zero), if $\overline{\text{RESET}}$ is activated before V_{CC} drops below spec and V_{PD} continues to be held within spec., the top 16 bytes in the Register File will retain their contents. $\overline{\text{RESET}}$ must be held low during the Power Down and should not be brought high until V_{CC} is within spec and the oscillator has stabilized. See Section 2.3.
V_{REF}	Reference voltage for the A/D converter (5V). V_{REF} is also the supply voltage to the analog portion of the A/D converter and the logic used to read Port 0. See Section 8.
ANGND	Reference ground for the A/D converter. Should be held at nominally the same potential as V_{SS}. See Section 8.
V_{PP}	Programming voltage for the EPROM devices. It should be +12.75V when programming and will float to 5V otherwise. It should not be above 5.5V on other than EPROM devices. This pin must float in the application circuit.
XTAL1	Input of the oscillator inverter and of the internal clock generator. See Section 1.5.
XTAL2	Output of the oscillator inverter. See Section 1.5.
CLKOUT	Output of the internal clock generator. The frequency of CLKOUT is ⅓ the oscillator frequency. It has a 33% duty cycle. See Section 1.5
$\overline{\text{RESET}}$	Reset input to the chip. Input low for at least 10XTAL1 cycles to reset the chip. The subsequent low-to-high transition re-synchronizes CLKOUT and commences a 10-state-time sequence in which the PSW is cleared, a byte read from 2018H loads CCR, and a jump to location 2080H is executed. Input high for normal operation. $\overline{\text{RESET}}$ has an internal pullup. See Section 13.
BUSWIDTH	Input for buswidth selection. If CCR bit 1 is a one, this pin selects the bus width for the bus cycle in progress. If BUSWIDTH is a 1, a 16-bit bus cycle occurs. If BUSWIDTH is a 0 an 8-bit cycle occurs. If CCR bit 1 is a 0, the bus is always an 8-bit bus. If this pin is left unconnected, it will rise to V_{CC}. See Section 2.7.
NMI	A positive transition causes a vector to external memory location 0000H. External memory from 00H through 0FFH is reserved for Intel development systems.
INST	Output high during an external memory read indicates the read is an instruction fetch. INST is valid throughout the bus cycle.
$\overline{\text{EA}}$	Input for memory select (External Access). $\overline{\text{EA}}$ equal to a TTL-high causes memory accesses to locations 2000H through 3FFFH to be directed to on-chip ROM/EPROM. $\overline{\text{EA}}$ equal to a TTL-low causes accesses to these locations to be directed to off-chip memory. $\overline{\text{EA}}$ = +12.5V causes execution to begin in the Programming mode on EPROM devices. $\overline{\text{EA}}$ has an internal pulldown, so it goes to 0 unless driven otherwise.
ALE/$\overline{\text{ADV}}$	Address Latch Enable or Address Valid output, as selected by CCR. Both pin options provide a latch to demultiplex the address from the address/data bus. When the pin is $\overline{\text{ADV}}$, it goes inactive high at the end of the bus cycle. $\overline{\text{ADV}}$ can be used as a chip select for external memory. ALE/$\overline{\text{ADV}}$ is activated only during external memory accesses. See Section 2.7.
$\overline{\text{RD}}$	Read signal output to external memory. $\overline{\text{RD}}$ is activated only during external memory reads.

PIN DESCRIPTIONS (Continued)

Symbol	Name and Function
$\overline{\text{WR}}/\overline{\text{WRL}}$	Write and Write Low output to external memory, as selected by the CCR. $\overline{\text{WR}}$ will go low for every external write, while $\overline{\text{WRL}}$ will go low only for external writes where an even byte is being written. $\overline{\text{WR}}/\overline{\text{WRL}}$ is activated only during external memory writes. See Section 2.7.
$\overline{\text{BHE}}/\overline{\text{WRH}}$	Bus High Enable or Write High output to external memory, as selected by the CCR. $\overline{\text{BHE}}$ = 0 selects the bank of memory that is connected to the high byte of the data bus. A0 = 0 selects the bank of memory that is connected to the low byte of the data bus. Thus accesses to a 16-bit wide memory can be to the low byte only (A0 = 0, $\overline{\text{BHE}}$ = 1), to the high byte only (A0 = 1, BHE # = 0), or both bytes (A0 = 0, $\overline{\text{BHE}}$ = 0). If the $\overline{\text{WRH}}$ function is selected, the pin will go low if the bus cycle is writing to an odd memory location. See Section 2.7.
READY	Ready input to lengthen external memory cycles, for interfacing to slow or dynamic memory, or for bus sharing. If the pin is high, CPU operation continues in a normal manner. If the pin is low prior to the falling edge of CLKOUT, the Memory Controller goes into a wait mode until the next positive transition in CLKOUT occurs with READY high. The bus cycle can be lengthened by up to 1 μs. When the external memory is not being used, READY has no effect. Internal control of the number of wait states inserted into a bus cycle held not ready is available through configuration of CCR. READY has a weak internal pullup, so it goes to 1 unless externally pulled low. See Section 2.7.
HSI	Inputs to High Speed Input Unit. Four HSI pins are available: HSI.0, HSI.1, HSI.2, and HSI.3. Two of them (HSI.2 and HSI.3) are shared with the HSO Unit. The HSI pins are also used as inputs by EPROM devices in Programming mode. See Section 6.
HSO	Outputs from High Speed Output Unit. Six HSO pins are available: HSO.0, HSO.1, HSO.2, HSO.3, HSO.4, and HSO.5. Two of them (HSO.4 and HSO.5) are shared with the HSI Unit. See Section 7.
Port 0	8-bit high impedance input-only port. These pins can be used as digital inputs and/or as analog inputs to the on-chip A/D converter. These pins are also a mode input to EPROM devices in the Programming mode. See Section 10.
Port 1	8-bit quasi-bidirectional I/O port. See Section 10.
Port 2	8-bit multi-functional port. Six of its pins are shared with other functions in the 8096BH, the remaining 2 are quasi-bidirectional. These pins are also used to input and output control signals on EPROM devices in Programming Mode. See Section 10.
Ports 3 and 4	8-bit bi-directional I/O ports with open drain outputs. These pins are shared with the multiplexed address/data bus which has strong internal pullups. Ports 3 and 4 are also used as a command, address and data path by EPROM devices operating in the programming mode. See Sections 2.7 and 10.

11.2 Pin List

The following is a list of pins in alphabetical order. Where a pin has two names it has been listed under both names, except for the system bus pins, AD0–AD15, which are listed under Port 3 and Port 4.

Name	68-Pin PLCC	68-Pin PGA	48-Pin DIP	64-Pin SDIP
ACH0/P0.0	6	4	—	4
ACH1/P0.1	5	5	—	3
ACH2/P0.2	7	3	—	5
ACH3/P0.3	4	6	—	2
ACH4/P0.4/MOD.0	11	67	43	9
ACH5/P0.5/MOD.1	10	68	42	8
ACH6/P0.6/MOD.2	8	2	40	6
ACH7/P0.7/MOD.3	9	1	41	7
ALE/ADV	62	16	34	60
ANGND	12	66	44	10
BHE/WRH	41	37	15	39
BUSWIDTH (TEST)	64	14	—	—
CLKOUT	65	13	—	—
EA	2	8	39	1
EXTINT/P2.2/PROG	15	63	47	13
HSI.0	24	54	3	22
HSI.1	25	53	4	23
HSI.2/HSO.4	26	52	5	24
HSI.3/HSO.5	27	51	6	25
HSO.0	28	50	7	26
HSO.1	29	49	8	27
HSO.2	34	44	9	32
HSO.3	35	43	10	33
HSO.4/HSI.2	26	52	5	24
HSO.5/HSI.3	27	51	6	25
INST	63	15	—	—
NMI	3	7	—	—
PWM/P2.5/PDO	39	39	13	37
PALE/P2.1/RXD	17	61	1	15
PROG/P2.2/EXTNT	15	63	47	13
PVER/P2.0/TXD	18	60	2	16
P0.0/ACH0	6	4	—	4
P0.1/ACH1	5	5	—	3
P0.2/ACH2	7	3	—	5
P0.3/ACH3	4	6	—	2
P0.4/ACH4/MOD.0	11	67	43	9
PO.5/ACH5/MOD.1	10	68	42	8
PO.6/ACH6/MOD.2	8	2	40	6
PO.7/ACH7/MOD.3	9	1	41	7
P1.0	19	59	—	17
P1.1	20	58	—	18
P1.2	21	57	—	19
P1.3	22	56	—	20
P1.4	23	55	—	21
P1.5	30	48	—	28
P1.6	31	47	—	29
P1.7	32	46	—	30
P2.0/TXD/PVER	18	60	2	16
P2.1/RXD/PALE	17	61	1	15
P2.2/EXTINT	15	63	47	13
P2.3/T2CLK	44	34	—	42
P2.4/T2RST	42	36	—	40
P2.5/PWM/PDO	39	39	13	37
P2.6	33	45	—	31
P2.7	38	40	—	36
P3.0/AD0 PVAL	60	18	32	58
P3.1/AD1 PVAL	59	19	31	57
P3.2/AD2 PVAL	58	20	30	56
P3.3/AD3 PVAL	57	21	29	55
P3.4/AD4 PVAL	56	22	28	54
P3.5/AD5 PVAL	55	23	27	53
P3.6/AD6 PVAL	54	24	26	52
P3.7/AD7 PVAL	53	25	25	51
P4.0/AD8 PVAL	52	26	24	50
P4.1/AD9 PVAL	51	27	23	49
P4.2/AD10 PVAL	50	28	22	48
P4.3/AD11 PVAL	49	29	21	47
P4.4/AD12 PVAL	48	30	20	46
P4.5/AD13 PVAL	47	31	19	45
P4.6/AD14 PVAL	46	32	18	44
P4.7/AD15 PVAL	45	33	17	43
RD	61	17	33	59
READY	43	35	16	41
RESET	16	62	48	14
RXD/P2.1	17	61	1	15
SALE/PVER/P2.0	18	60	2	16
SPROG/PDO/P2.5	39	39	13	37
TXD/P2.0/SALE	18	60	2	16
T2CLK/P2.3	44	34	—	42
T2RST/P2.4	42	36	—	40
V_{PP}	37	41	12	35
V_{CC}	1	9	38	64
V_{PD}	14	64	46	12
V_{REF}	13	65	45	11
V_{SS}	68	10	11	34
V_{SS}	36	42	37	63
WR/WRL	40	38	14	38
WRH/BHE	41	37	15	39
XTAL1	67	11	36	61
XTAL2	66	12	35	62

The Following pins are not bonded out in the 48-pin package:

P1.0 through P1.7, P0.0 through P0.3, P2.3, P2.4, P2.6, P2.7 CLKOUT, INST, NMI, TEST, T2CLK (P2.3), T2RST (P2.4).

11.3 Packaging

The MCS-96 products are available in 48-pin, 64-pin and 68-pin packages, with and without A/D, and with and without on-chip ROM or EPROM. The MCS-96 numbeiing system shown below this section shows the pinouts for the 48- and 68-pin packages. The 48-pin version is offered in a Dual-In-Line package while the 68-pin versions come in a Plastic Leaded Chip Carrier (PLCC), a Pin Grid Array (PGA) or a Type "B" Leadless Chip Carrier.

The MCS®-96 Family Nomenclature

	Factory Masked ROM			CPU			User Programmable					
							EPROM			OTP		
	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin
ANALOG	8397BH 8397JF	8397JF 8397BH	8395BH 8398	8097BH 8097JF	8097BH 8097JF	8095BH 8098	8797BH		8795BH 8798	8797BH 8797JF	8797JF 8797BH	8798
NO ANALOG	8396BH			8096BH								

Transistor Count

Device Type	#MOS Gates
839XBH/879XBH	120,000
809XBH	50,000

MTBF Calculations*

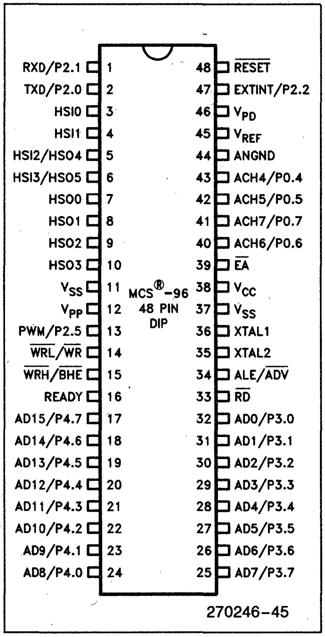
3.8 × 10^7 Device Hours @ 55°C
1.7 × 10^7 Device Hours @ 70°C

*MTBF data was obtained through calculations based upon the actual average junction temperatures under stress at 55°C and 70°C ambient.

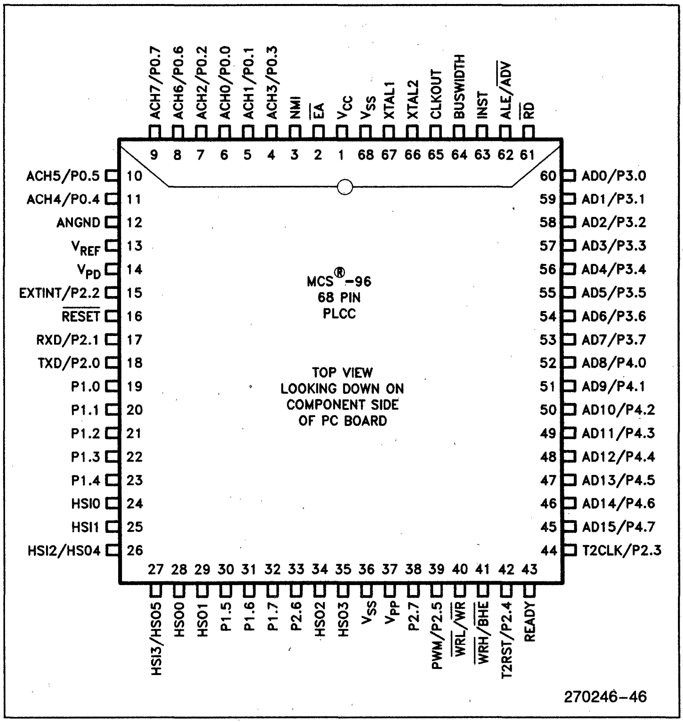
Thermal Characteristics

T_{CASE}		Package Type	θJa	θJc
COMM'L	EXPRESS			
85°C	100°C	PGA	35°C/W	10°C/W
85°C	100°C	PLCC	37°C/W	13°C/W
		LCC	28°C/W	—
		Plastic DIP	38°C/W	19°C/W
79.75°C	94.75°C	Ceramic DIP	26°C/W	6.5°C/W

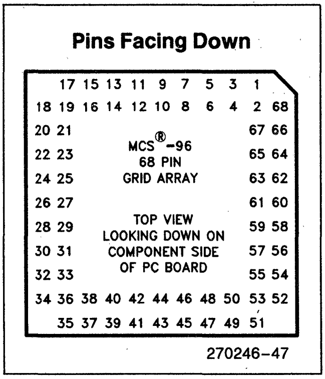
11.4 Package Diagrams



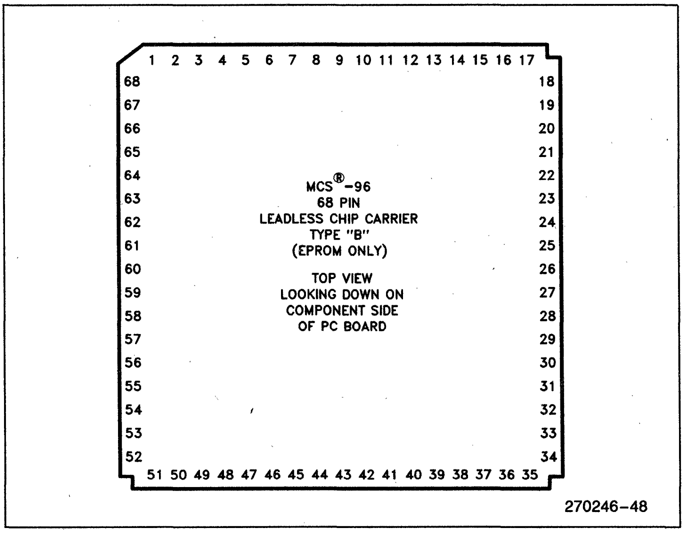
48-Pin Package



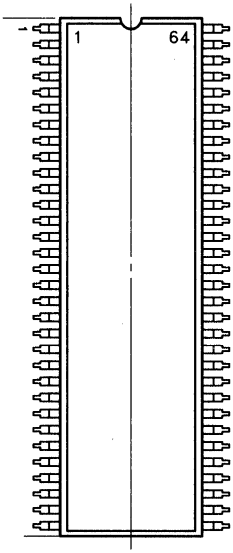
68-Pin Package (PLCC - Top View)



68-Pin Package (Pin Grid Array - Top View)



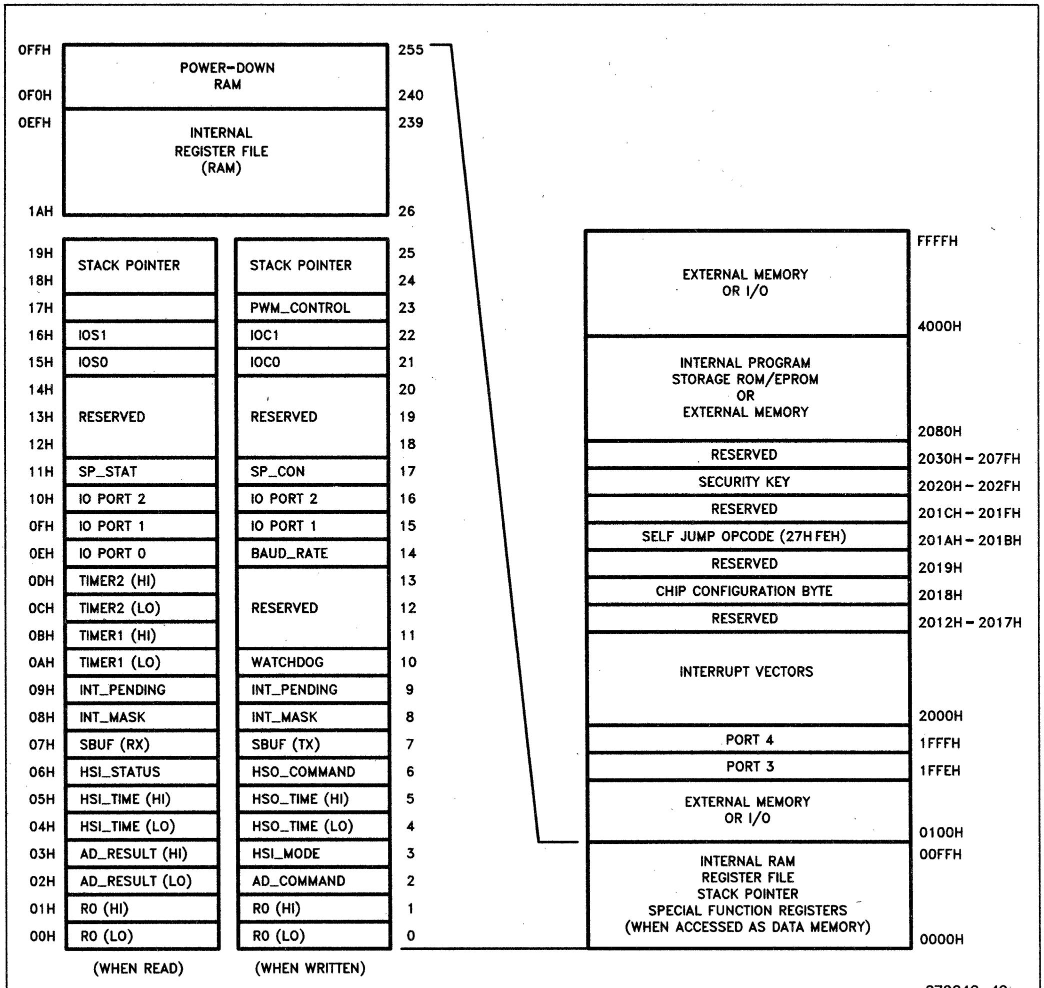
68-Pin Package (LCC - Top View)



270246–65

11.5 Shrink-DIP Package

11.6 Memory Map



0FFH – 0F0H	POWER-DOWN RAM	255 – 240
0EFH – 1AH	INTERNAL REGISTER FILE (RAM)	239 – 26

	(WHEN READ)	(WHEN WRITTEN)	
19H	STACK POINTER	STACK POINTER	25
18H			24
17H		PWM_CONTROL	23
16H	IOS1	IOC1	22
15H	IOS0	IOC0	21
14H	RESERVED	RESERVED	20
13H			19
12H			18
11H	SP_STAT	SP_CON	17
10H	IO PORT 2	IO PORT 2	16
0FH	IO PORT 1	IO PORT 1	15
0EH	IO PORT 0	BAUD_RATE	14
0DH	TIMER2 (HI)	RESERVED	13
0CH	TIMER2 (LO)		12
0BH	TIMER1 (HI)		11
0AH	TIMER1 (LO)	WATCHDOG	10
09H	INT_PENDING	INT_PENDING	9
08H	INT_MASK	INT_MASK	8
07H	SBUF (RX)	SBUF (TX)	7
06H	HSI_STATUS	HSO_COMMAND	6
05H	HSI_TIME (HI)	HSO_TIME (HI)	5
04H	HSI_TIME (LO)	HSO_TIME (LO)	4
03H	AD_RESULT (HI)	HSI_MODE	3
02H	AD_RESULT (LO)	AD_COMMAND	2
01H	R0 (HI)	R0 (HI)	1
00H	R0 (LO)	R0 (LO)	0

EXTERNAL MEMORY OR I/O	FFFFH – 4000H
INTERNAL PROGRAM STORAGE ROM/EPROM OR EXTERNAL MEMORY	2080H
RESERVED	2030H – 207FH
SECURITY KEY	2020H – 202FH
RESERVED	201CH – 201FH
SELF JUMP OPCODE (27H FEH)	201AH – 201BH
RESERVED	2019H
CHIP CONFIGURATION BYTE	2018H
RESERVED	2012H – 2017H
INTERRUPT VECTORS	2000H
PORT 4	1FFFH
PORT 3	1FFEH
EXTERNAL MEMORY OR I/O	0100H
INTERNAL RAM REGISTER FILE STACK POINTER SPECIAL FUNCTION REGISTERS (WHEN ACCESSED AS DATA MEMORY)	00FFH – 0000H

270246-49

11.7 Instruction Summary

Mnemonic	Oper-ands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
ADD/ADDB	2	D ← D + A	✓	✓	✓	✓	↑	—	
ADD/ADDB	3	D ← B + A	✓	✓	✓	✓	↑	—	
ADDC/ADDCB	2	D ← D + A + C	↓	✓	✓	✓	↑	—	
SUB/SUBB	2	D ← D − A	✓	✓	✓	✓	↑	—	
SUB/SUBB	3	D ← B − A	✓	✓	✓	✓	↑	—	
SUBC/SUBCB	2	D ← D − A + C − 1	↓	✓	✓	✓	↑	—	
CMP/CMPB	2	D − A	✓	✓	✓	✓	↑	—	
MUL/MULU	2	D, D + 2 ← D * A	—	—	—	—	—	?	2
MUL/MULU	3	D, D + 2 ← B * A	—	—	—	—	—	?	2
MULB/MULUB	2	D, D + 1 ← D * A	—	—	—	—	—	?	3
MULB/MULUB	3	D, D + 1 ← B * A	—	—	—	—	—	?	3
DIVU	2	D ← (D, D + 2)/A, D + 2 ← remainder	—	—	—	✓	↑	—	2
DIVUB	2	D ← (D, D + 1)/A, D + 1 ← remainder	—	—	—	✓	↑	—	3
DIV	2	D ← (D, D + 2)/A, D + 2 ← remainder	—	—	—	?	↑	—	
DIVB	2	D ← (D, D + 1)/A, D + 1 ← remainder	—	—	—	?	↑	—	
AND/ANDB	2	D ← D and A	✓	✓	0	0	—	—	
AND/ANDB	3	D ← B and A	✓	✓	0	0	—	—	
OR/ORB	2	D ← D or A	✓	✓	0	0	—	—	
XOR/XORB	2	D ← D (excl. or) A	✓	✓	0	0	—	—	
LD/LDB	2	D ← A	—	—	—	—	—	—	
ST/STB	2	A ← D	—	—	—	—	—	—	
LDBSE	2	D ← A; D + 1 ← SIGN(A)	—	—	—	—	—	—	3, 4
LDBZE	2	D ← A; D + 1 ← 0	—	—	—	—	—	—	3, 4
PUSH	1	SP ← SP − 2; (SP) ← A	—	—	—	—	—	—	
POP	1	A ← (SP); SP ← SP + 2	—	—	—	—	—	—	
PUSHF	0	SP ← SP − 2; (SP) ← PSW; PSW ← 0000H I ← 0	0	0	0	0	0	0	
POPF	0	PSW ← (SP); SP ← SP + 2; I ← ✓	✓	✓	✓	✓	✓	✓	
SJMP	1	PC ← PC + 11-bit offset	—	—	—	—	—	—	5
LJMP	1	PC ← PC + 16-bit offset	—	—	—	—	—	—	5
BR [indirect]	1	PC ← (A)	—	—	—	—	—	—	
SCALL	1	SP ← SP − 2; (SP) ← PC; PC ← PC + 11-bit offset	—	—	—	—	—	—	5
LCALL	1	SP ← SP − 2; (SP) ← PC; PC ← PC + 16-bit offset	—	—	—	—	—	—	5
RET	0	PC ← (SP); SP ← SP + 2	—	—	—	—	—	—	
J (conditional)	1	PC ← PC + 8-bit offset (if taken)	—	—	—	—	—	—	5
JC	1	Jump if C = 1	—	—	—	—	—	—	5
JNC	1	Jump if C = 0	—	—	—	—	—	—	5
JE	1	Jump if Z = 1	—	—	—	—	—	—	5

NOTES:

1. If the mnemonic ends in "B", a byte operation is performed, otherwise a word operation is done. Operands D, B, and A must conform to the alignment rules for the required operand type. D and B are locations in the Register File; A can be located anywhere in memory.
2. D, D + 2 are consecutive WORDS in memory; D is DOUBLE-WORD aligned.
3. D, D + 1 are consecutive BYTES in memory; D is WORD aligned.
4. Changes a byte to a word.
5. Offset is a 2's complement number.

Mnemonic	Oper-ands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
JNE	1	Jump if Z = 0	—	—	—	—	—	—	5
JGE	1	Jump if N = 0	—	—	—	—	—	—	5
JLT	1	Jump if N = 1	—	—	—	—	—	—	5
JGT	1	Jump if N = 0 and Z = 0	—	—	—	—	—	—	5
JLE	1	Jump if N = 1 or Z = 1	—	—	—	—	—	—	5
JH	1	Jump if C = 1 and Z = 0	—	—	—	—	—	—	5
JNH	1	Jump if C = 0 or Z = 1	—	—	—	—	—	—	5
JV	1	Jump if V = 1	—	—	—	—	—	—	5
JNV	1	Jump if V = 0	—	—	—	—	—	—	5
JVT	1	Jump if VT = 1; Clear VT	—	—	—	—	0	—	5
JNVT	1	Jump if VT = 0; Clear VT	—	—	—	—	0	—	5
JST	1	Jump if ST = 1	—	—	—	—	—	—	5
JNST	1	Jump if ST = 0	—	—	—	—	—	—	5
JBS	3	Jump if Specified Bit = 1	—	—	—	—	—	—	5, 6
JBC	3	Jump if Specified Bit = 0	—	—	—	—	—	—	5, 6
DJNZ	1	D ← D − 1; if D ≠ 0 then PC ← PC + 8-bit offset	—	—	—	—	—	—	5
DEC/DECB	1	D ← D − 1	✓	✓	✓	✓	↑	—	
NEG/NEGB	1	D ← 0 − D	✓	✓	✓	✓	↑	—	
INC/INCB	1	D ← D + 1	✓	✓	✓	✓	↑	—	
EXT	1	D ← D; D + 2 ← Sign (D)	✓	✓	0	0	—	—	2
EXTB	1	D ← D; D + 1 ← Sign(D)	✓	✓	0	0	—	—	3
NOT/NOTB	1	D ← Logical Not (D)	✓	✓	0	0	—	—	
CLR/CLRB	1	D ← 0	1	0	0	0	—	—	
SHL/SHLB/SHLL	2	C ← msb — — — — — — lsb ← 0	✓	?	✓	✓	↑	—	7
SHR/SHRB/SHRL	2	0 → msb — — — — — — lsb → C	✓	?	✓	0	—	✓	7
SHRA/SHRAB/SHRAL	2	msb → msb — — — — — — lsb → C	✓	✓	✓	0	—	✓	7
SETC	0	C ← 1	—	—	1	—	—	—	
CLRC	0	C ← 0	—	—	0	—	—	—	
CLRVT	0	VT ← 0	—	—	—	—	0	—	
RST	0	PC ← 2080H	0	0	0	0	0	0	8
DI	0	Disable All Interrupts (I ← 0)	—	—	—	—	—	—	
EI	0	Enable All Interrupts (I ← 1)	—	—	—	—	—	—	
NOP	0	PC ← PC + 1	—	—	—	—	—	—	
SKIP	0	PC ← PC + 2	—	—	—	—	—	—	
NORML	2	Left shift till msb = 1; D ← shift count	✓	?	0	—	—	—	7
TRAP	0	SP ← SP − 2; (SP) ← PC PC ← (2010H)	—	—	—	—	—	—	9

NOTES:

1. If the mnemonic ends in "B", a byte operation is performed, otherwise a word operation is done. Operands D, B and A must conform to the alignment rules for the required operand type. D and B are locations in the Register File; A can be located anywhere in memory.

5. Offset is a 2's complement number.

6. Specified bit is one of the 2048 bits in the register file.

7. The "L" (Long) suffix indicates double-word operation.

8. Initiates a Reset by pulling $\overline{\text{RESET}}$ low. Software should re-initialize all the necessary registers with code starting at 2080H.

9. The assembler will not accept this mnemonic.

11.8 Opcode and State Time Listing

		DIRECT			IMMEDIATE			INDIRECT⑥					INDEXED⑥				
								NORMAL			AUTO-INC.		SHORT			LONG	
MNEMONIC	OPERANDS	OPCODE	BYTES	STATE TIMES	OPCODE	BYTES	STATE TIMES	OPCODE	BYTES	STATE① TIMES	BYTES	STATE① TIMES	OPCODE	BYTES	STATE① TIMES⑥	BYTES	STATE① TIMES⑥
ARITHMETIC INSTRUCTIONS																	
ADD	2	64	3	4	65	4	5	66	3	6/11	3	7/12	67	4	6/11	5	7/12
ADD	3	44	4	5	45	5	6	46	4	7/12	4	8/13	47	5	7/12	6	8/13
ADDB	2	74	3	4	75	3	4	76	3	6/11	3	7/12	77	4	6/11	5	7/12
ADDB	3	54	4	5	55	4	5	56	4	7/12	4	8/13	57	5	7/12	6	8/13
ADDC	2	A4	3	4	A5	4	5	A6	3	6/11	3	7/12	A7	4	6/11	5	7/12
ADDCB	2	B4	3	4	B5	3	4	B6	3	6/11	3	7/12	B7	4	6/11	5	7/12
SUB	2	68	3	4	69	4	5	6A	3	6/11	3	7/12	6B	4	6/11	5	7/12
SUB	3	48	4	5	49	5	6	4A	4	7/12	4	8/13	4B	5	7/12	6	8/13
SUBB	2	78	3	4	79	3	4	7A	3	6/11	3	7/12	7B	4	6/11	5	7/12
SUBB	3	58	4	5	59	4	5	5A	4	7/12	4	8/13	5B	5	7/12	6	8/13
SUBC	2	A8	3	4	A9	4	5	AA	3	6/11	3	7/12	AB	4	6/11	5	7/12
SUBCB	2	B8	3	4	B9	3	4	BA	3	6/11	3	7/12	BB	4	6/11	5	7/12
CMP	2	88	3	4	89	4	5	8A	3	6/11	3	7/12	8B	4	6/11	5	7/12
CMPB	2	98	3	4	99	3	4	9A	3	6/11	3	7/12	9B	4	6/11	5	7/12
MULU	2	6C	3	25	6D	4	26	6E	3	27/32	3	28/33	6F	4	27/32	5	28/33
MULU	3	4C	4	26	4D	5	27	4E	4	28/33	4	29/34	4F	5	28/33	6	29/34
MULUB	2	7C	3	17	7D	3	17	7E	3	19/24	3	20/25	7F	4	19/24	5	20/25
MULUB	3	5C	4	18	5D	4	18	5E	4	20/25	4	21/26	5F	5	20/25	6	21/26
MUL	2	②	4	29	②	5	30	②	4	31/36	4	32/37	②	5	31/36	6	32/37
MUL	3	②	5	30	②	6	31	②	5	32/37	5	33/38	②	6	32/37	7	33/38
MULB	2	②	4	21	②	4	21	②	4	23/28	4	24/29	②	5	23/28	6	24/29
MULB	3	②	5	22	②	5	22	②	5	24/29	5	25/30	②	6	24/29	7	25/30
DIVU	2	8C	3	25	8D	4	26	8E	3	28/32	3	29/33	8F	4	28/32	5	29/33
DIVUB	2	9C	3	17	9D	3	17	9E	3	20/24	3	21/25	9F	4	20/24	5	21/25
DIV	2	②	4	29	②	5	30	②	4	32/36	4	33/37	②	5	32/36	6	33/37
DIVB	2	②	4	21	②	4	21	②	4	24/28	4	25/29	②	5	24/28	6	25/29

270246-63

NOTES:

*Long indexed and Indirect + instructions have identical opcodes with Short indexed and Indirect modes, respectively. The second byte of instructions using any Indirect or indexed addressing mode specifies the exact mode used. If the second byte is even, use Indirect or Short indexed. If it is odd, use Indirect + or Long indexed. In all cases the second byte of the instruction always specifies an even (word) location for the address referenced.

① Number of state times shown for internal/external operands.

② The opcodes for signed multiply and divide are the opcodes for the unsigned functions with an "FE" appended as a prefix.

⑥ State times shown for 16-bit bus.

		DIRECT			IMMEDIATE			INDIRECT⑧					INDEXED⑧				
								NORMAL			AUTO-INC.		SHORT			LONG	
MNEMONIC	OPERANDS	OPCODE	BYTES	STATE TIMES	OPCODE	BYTES	STATE TIMES	OPCODE	BYTES	STATE① TIMES	BYTES	STATE① TIMES	OPCODE	BYTES	STATE① TIMES⑧	BYTES	STATE① TIMES⑧
LOGICAL INSTRUCTIONS																	
AND	2	60	3	4	61	4	5	62	3	6/11	3	7/12	63	4	6/11	5	7/12
AND	3	40	4	5	41	5	6	42	4	7/12	4	8/13	43	5	7/12	6	8/13
ANDB	2	70	3	4	71	3	4	72	3	6/11	3	7/12	73	4	6/11	5	7/12
ANDB	3	50	4	5	51	4	5	52	4	7/12	4	8/13	53	5	7/12	6	8/13
OR	2	80	3	4	81	4	5	82	3	6/11	3	7/12	83	4	6/11	5	7/12
ORB	2	90	3	4	91	3	4	92	3	6/11	3	7/12	93	4	6/11	5	7/12
XOR	2	84	3	4	85	4	5	86	3	6/11	3	7/12	87	4	6/11	5	7/12
XORB	2	94	3	4	95	3	4	96	3	6/11	3	7/12	97	4	6/11	5	7/12
DATA TRANSFER INSTRUCTIONS																	
LD	2	A0	3	4	A1	4	5	A2	3	6/11	3	7/12	A3	4	6/11	5	7/12
LDB	2	B0	3	4	B1	3	4	B2	3	6/11	3	7/12	B3	4	6/11	5	7/12
ST	2	C0	3	4	—	—	——	C2	3	7/11	3	8/12	C3	4	7/11	5	8/12
STB	2	C4	3	4	—	—	——	C6	3	7/11	3	8/12	C7	4	7/11	5	8/12
LDBSE	2	BC	3	4	BD	3	4	BE	3	6/11	3	7/12	BF	4	6/11	5	7/12
LDBZE	2	AC	3	4	AD	3	4	AE	3	6/11	3	7/12	AF	4	6/11	5	7/12
STACK OPERATIONS (internal stack)																	
PUSH	1	C8	2	8	C9	3	8	CA	2	11/15	2	12/16	CB	3	11/15	4	12/16
POP	1	CC	2	12	—	—	——	CE	2	14/18	2	14/18	CF	3	14/18	4	14/18
PUSHF	0	F2	1	8													
POPF	0	F3	1	9													
STACK OPERATIONS (external stack)																	
PUSH	1	C8	2	12	C9	3	12	CA	2	15/19	2	16/20	CB	3	15/19	4	16/20
POP	1	CC	2	14	—	—	——	CE	2	16/20	2	16/20	CF	3	16/20	4	16/20
PUSHF	0	F2	1	12													
POPF	0	F3	1	13													

JUMPS AND CALLS							
MNEMONIC	OPCODE	BYTES	STATES	MNEMONIC	OPCODE	BYTES	STATES
LJMP	E7	3	8	LCALL	EF	3	13/16⑤
SJMP	20-27④	2	8	SCALL	28-2F④	2	13/16⑤
BR[ ]	E3	2	8	RET	F0	1	12/16⑤
				TRAP③	F7	1	21/24

270246-64

NOTES:

① Number of state times shown for internal/external operands.

③ The assembler does not accept this mnemonic.

④ The least significant 3 bits of the opcode are concatenated with the following 8 bits to form an 11-bit, 2's complement, offset for the relative call or jump.

⑤ State times for stack located internal/external.

⑧ State times shown for 16-bit bus.

CONDITIONAL JUMPS

All conditional jumps are 2 byte instructions. They require 8 state times if the jump is taken, 4 if it is not.[8]

MNEMONIC	OPCODE	MNEMONIC	OPCODE	MNEMONIC	OPCODE	MNEMONIC	OPCODE
JC	DB	JE	DF	JGE	D6	JGT	D2
JNC	D3	JNE	D7	JLT	DE	JLE	DA
JH	D9	JV	DD	JVT	DC	JST	D8
JNH	D1	JNV	D5	JNVT	D4	JNST	D0

JUMP ON BIT CLEAR OR BIT SET

These instructions are 3-byte instructions. They require 9 state times if the jump is taken, 5 if it is not.[8]

	BIT NUMBER							
MNEMONIC	0	1	2	3	4	5	6	7
JBC	30	31	32	33	34	35	36	37
JBS	38	39	3A	3B	3C	3D	3E	3F

LOOP CONTROL

MNEMONIC	OPCODE	BYTES	STATE TIMES
DJNZ	EO	3	5/9 STATE TIME (NOT TAKEN/TAKEN)[8]

SINGLE REGISTER INSTRUCTIONS

MNEMONIC	OPCODE	BYTES	STATES[8]	MNEMONIC	OPCODE	BYTES	STATES[8]
DEC	05	2	4	EXT	06	2	4
DECB	15	2	4	EXTB	16	2	4
NEG	03	2	4	NOT	02	2	4
NEGB	13	2	4	NOTB	12	2	4
INC	07	2	4	CLR	01	2	4
INCB	17	2	4	CLRB	11	2	4

SHIFT INSTRUCTIONS

INSTR MNEMONIC	WORD		INSTR MNEMONIC	BYTE		INSTR MNEMONIC	DBL WD		STATE TIMES[8]
	OP	B		OP	B		OP	B	
SHL	09	3	SHLB	19	3	SHLL	0D	3	7 + 1 PER SHIFT[7]
SHR	08	3	SHRB	18	3	SHRL	0C	3	7 + 1 PER SHIFT[7]
SHRA	0A	3	SHRAB	1A	3	SHRAL	0E	3	7 + 1 PER SHIFT[7]

SPECIAL CONTROL INSTRUCTIONS

MNEMONIC	OPCODE	BYTES	STATES[8]	MNEMONIC	OPCODE	BYTES	STATES[8]
SETC	F9	1	4	DI	FA	1	4
CLRC	F8	1	4	EI	FB	1	4
CLRVT	FC	1	4	NOP	FD	1	4
RST[6]	FF	1	166	SKIP	00	2	4

NORMALIZE

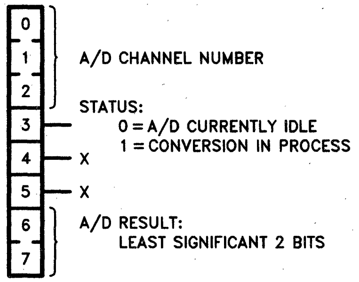
MNEMONIC	OPCODE	BYTES	STATE TIMES
NORML	0F	3	11 + 1 PER SHIFT

NOTES:

6. This instruction takes 2 states to pull $\overline{\text{RESET}}$ low, then holds it low for at least one state time to initiate a reset. The reset takes 13 states, at which time the program restarts at location 2080H.
7. Execution will take at least 8 states, even for 0 shift.
8. State times shown for 16-bit bus.

11.9 SFR Summary

A/D Result LO (02H)



A/D CHANNEL NUMBER

STATUS:
0 = A/D CURRENTLY IDLE
1 = CONVERSION IN PROCESS

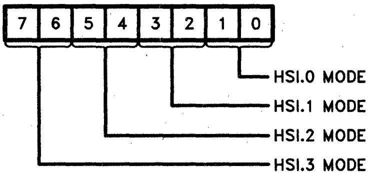
X

X

A/D RESULT:
LEAST SIGNIFICANT 2 BITS

270246–50

HSI_Mode (03H)

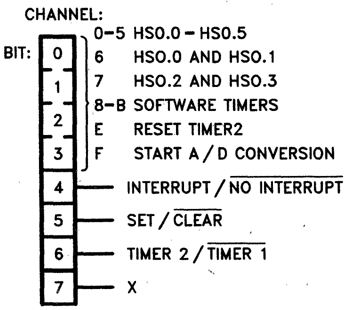


WHERE EACH 2 – BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

270246–51

HSO Command (06H)



CHANNEL:

0–5	HSO.0 – HSO.5
6	HSO.0 AND HSO.1
7	HSO.2 AND HSO.3
8–B	SOFTWARE TIMERS
E	RESET TIMER2
F	START A / D CONVERSION

INTERRUPT / $\overline{\text{NO INTERRUPT}}$

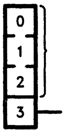
SET / $\overline{\text{CLEAR}}$

TIMER 2 / $\overline{\text{TIMER 1}}$

X

270246–52

A/D Command (02H)

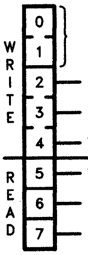


CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.

GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).

270246–53

SPCON/SPSTAT (11H)



BIT1, BIT0 SPECIFY THE MODE
00 = MODE 0 10 = MODE 2
01 = MODE 1 11 = MODE 3

PEN	ENABLE THE PARITY FUNCTION
REN	ENABLES THE RECEIVE FUNCTION
TB8	PROGRAMS THE 9TH DATA BIT
TI	IS THE TRANSMIT INTERRUPT FLAG
RI	IS THE RECEIVE INTERRUPT FLAG
RB8	IS THE 9TH DATA RECEIVED (IF NOT PARITY)
RPE	IS THE PARITY ERROR INDICATOR (IF PARITY ACTIVE)

270246–54

Baud Rate Calculations

Using XTAL1:

Mode 0: $\text{Baud Rate} = \dfrac{\text{XTAL1 frequency}}{4^*(B + 1)}; B \neq 0$

Others: $\text{Baud Rate} = \dfrac{\text{XTAL1 frequency}}{64^*(B + 1)}$

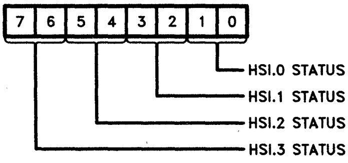
Using T2CLK:

Mode 0: $\text{Baud Rate} = \dfrac{\text{T2CLK frequency}}{B}; B \neq 0$

Others: $\text{Baud Rate} = \dfrac{\text{T2CLK frequency}}{16^*B}; B \neq 0$

Note that B cannot equal 0, except when using XTAL1 in other than Mode 0.

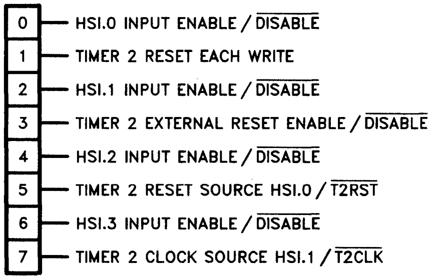
HSI_Status (06H)



WHERE FOR EACH 2 – BIT STATUS FIELD THE LOWER BIT INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT INDICATES THE CURRENT STATUS OF THE PIN.

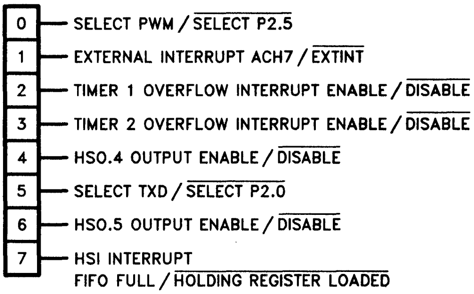
270246–55

IOC0 (15H)


0 — HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1 — TIMER 2 RESET EACH WRITE
2 — HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3 — TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4 — HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5 — TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6 — HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7 — TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$

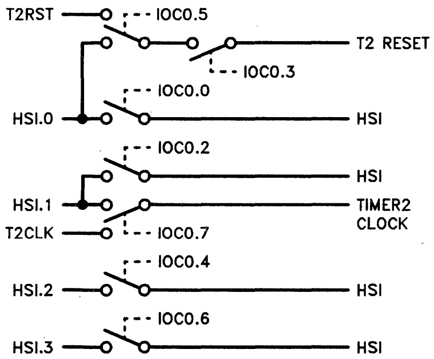
270246–56

IOC1 (16H)


0 — SELECT PWM / $\overline{\text{SELECT P2.5}}$
1 — EXTERNAL INTERRUPT ACH7 / $\overline{\text{EXTINT}}$
2 — TIMER 1 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
3 — TIMER 2 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
4 — HSO.4 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
5 — SELECT TXD / $\overline{\text{SELECT P2.0}}$
6 — HSO.5 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
7 — HSI INTERRUPT FIFO FULL / $\overline{\text{HOLDING REGISTER LOADED}}$

270246–59

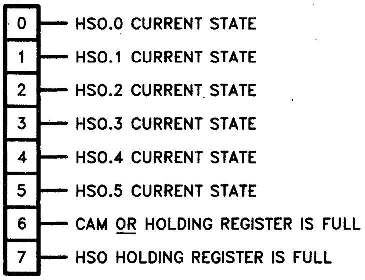
IOC0 (15H)



270246–57

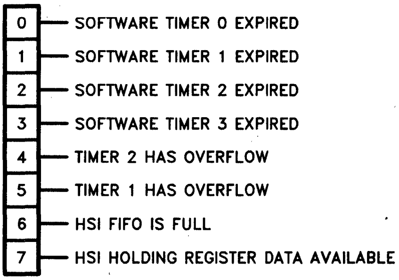
Vector	Vector Location		Priority
	(High Byte)	(Low Byte)	
Software Trap	2011H	2010H	Not Applicable
Extint	200FH	200EH	7 (Highest)
Serial Port	200DH	200CH	6
Software Timers	200BH	200AH	5
HSI.0	2009H	2008H	4
High Speed Outputs	2007H	2006H	3
HSI Data Available	2005H	2004H	2
A/D Conversion Complete	2003H	2002H	1
Timer Overflow	2001H	2000H	0 (Lowest)

IOS0 (15H)



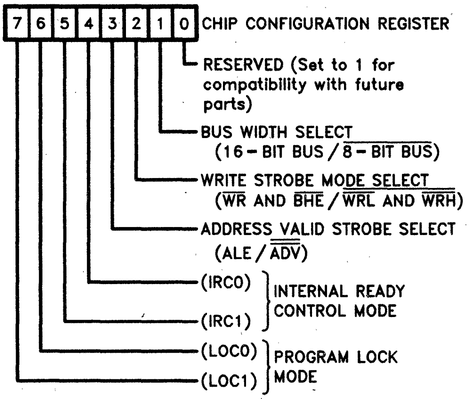
270246–58

IOS1 (16H)


0 — SOFTWARE TIMER 0 EXPIRED
1 — SOFTWARE TIMER 1 EXPIRED
2 — SOFTWARE TIMER 2 EXPIRED
3 — SOFTWARE TIMER 3 EXPIRED
4 — TIMER 2 HAS OVERFLOW
5 — TIMER 1 HAS OVERFLOW
6 — HSI FIFO IS FULL
7 — HSI HOLDING REGISTER DATA AVAILABLE

270246–60

Chip Configuration


7 6 5 4 3 2 1 0
CHIP CONFIGURATION REGISTER
RESERVED (Set to 1 for compatibility with future parts)
BUS WIDTH SELECT (16-BIT BUS / $\overline{\text{8-BIT BUS}}$)
WRITE STROBE MODE SELECT ($\overline{\text{WR}}$ AND $\overline{\text{BHE}}$ / $\overline{\text{WRL}}$ AND $\overline{\text{WRH}}$)
ADDRESS VALID STROBE SELECT (ALE / $\overline{\text{ADV}}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE

270246-61

Internal Ready Control

IRC1	IRC0	Description
0	0	Limit to 1 Wait State
0	1	Limit to 2 Wait States
1	0	Limit to 3 Wait States
1	1	Disable Internal Ready Control

Program Lock Modes

LOC1	LOC0	Protection
0	0	Read and Write Protected
0	1	Read Protected
1	0	Write Protected
1	1	No Protection

Programming Function PMODE Values

PMODE	Programming Mode
0-4	Reserved
5	Slave Programming
6-0BH	Reserved
0CH	Auto Programming Mode
0DH	Program Configuration Byte
0EH-0FH	Reserved

Slave Programming Mode Commands

P4.7	P4.6	Action
0	0	Word Dump
0	1	Data Verify
1	0	Data Program
1	1	Reserved

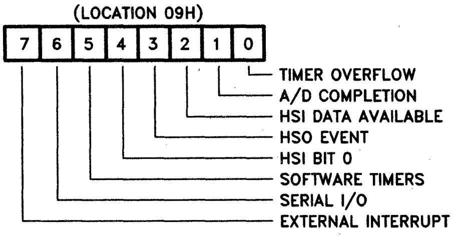
8X9XBH Signature Word

Device	Signature Word
879XBH	896FH
839XBH	896EH
809XBH	Undefined

Port 2 Pin Functions

Port	Function	Alternate Function
P2.0	Output	TXD (Serial Port Transmit)
P2.1	Input	RXD (Serial Port Receive)
P2.2	Input	EXTINT (External Interrupt)
P2.3	Input	T2CLK (Timer 2 Clock)
P2.4	Input	T2RST (Timer 2 Reset)
P2.5	Output	PWM (Pulse Width Modulation)

Interrupt Pending Register



270246-62

PSW Register

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Z	N	V	VT	C	—	I	ST	<Interrupt Mask Reg>							

PRELIMINARY

MSC®-96
809XBH, 839XBH, 879XBH
ADVANCED 16-BIT MICROCONTROLLER WITH 8- OR 16-BIT EXTERNAL BUS

- 879XBH: an 809XBH with 8K Bytes of On-Chip EPROM
- 839XBH: an 809XBH with 8K Bytes of On-Chip ROM

- **232 Byte Register File**
- **Register-to-Register Architecture**
- **10-Bit A/D Converter with S/H**
- **Five 8-Bit I/O Ports**
- **20 Interrupt Sources**
- **Pulse-Width Modulated Output**
- **ROM/EPROM Lock**
- **Run-Time Programmable EPROM**
- **High Speed I/O Subsystem**
- **Full Duplex Serial Port**
- **Dedicated Baud Rate Generator**
- **6.25 μs 16 x 16 Multiply**
- **6.25 μs 32/16 Divide**
- **16-Bit Watchdog Timer**
- **Four 16-Bit Software Timers**
- **Two 16-Bit Counter/Timers**

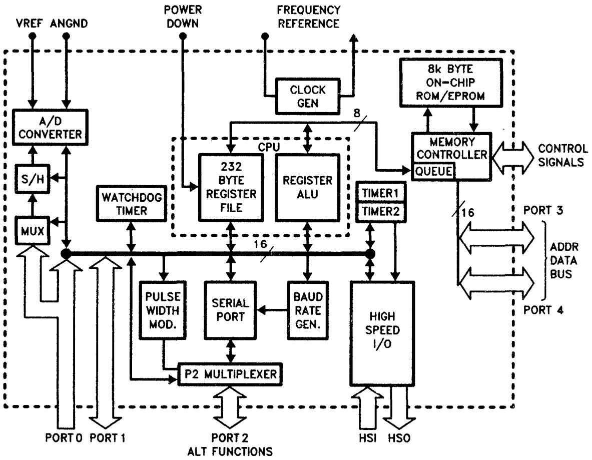
The MCS-96 family of 16-bit microcontrollers consists of many members, all of which are designed for high-speed control functions. The MCS-96 family members produced using Intel's HMOS-III process are described in this data sheet.

The CPU supports bit, byte, and word operations. Thirty-two bit double-words are supported for a subset of the instruction set. With a 12 MHz input frequency the 8096BH can do a 16-bit addition in 1.0 μs and a 16 x 16-bit multiply or 32/16 divide in 6.25 μs. Instruction execution times average 1 to 2 μs in typical applications.

Four high-speed trigger inputs are provided to record the times at which external events occur. Six high-speed pulse generator outputs are provided to trigger external events at preset times. The high-speed output unit can simultaneously perform software timer functions. Up to four 16-bit software timers can be in operation at once.

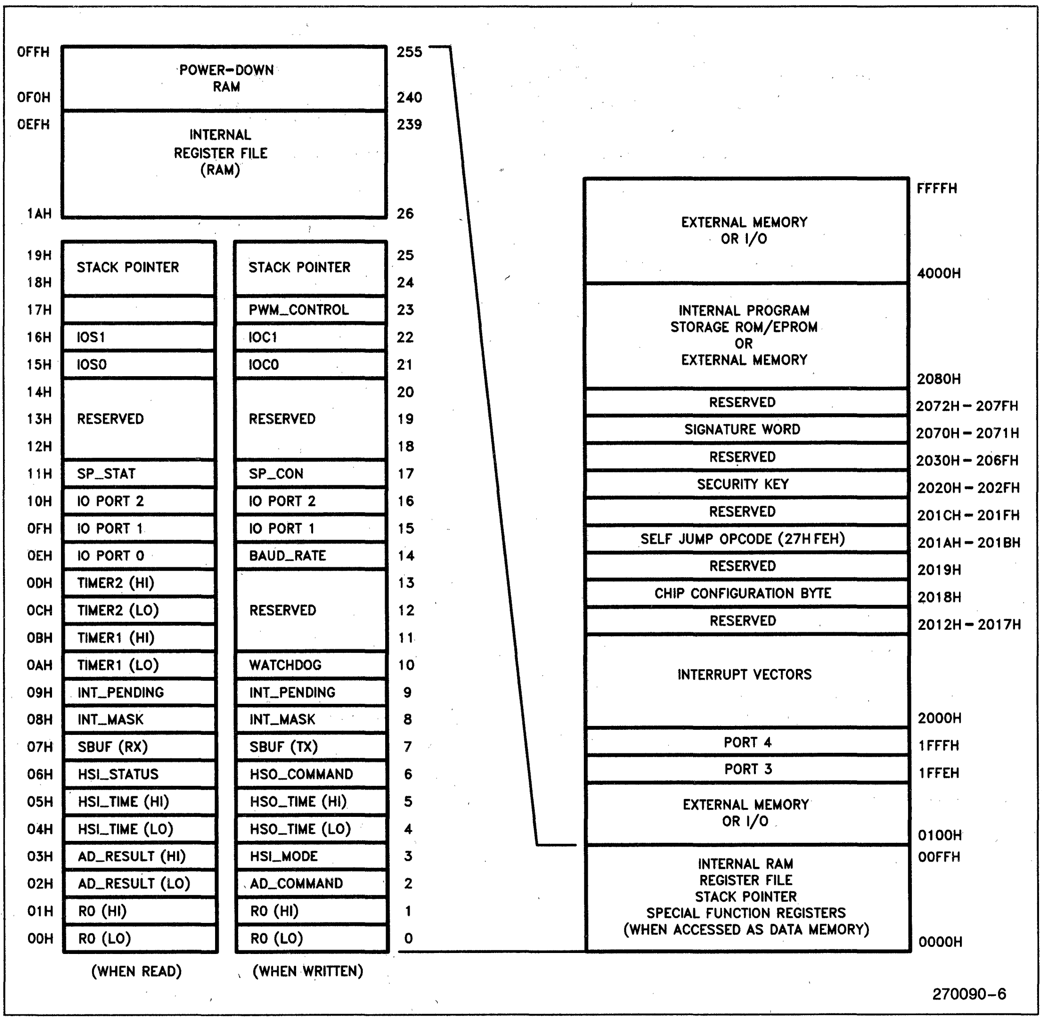
The on-chip A/D converter includes a Sample and Hold, and converts up to 8 multiplexed analog input channels to 10-bit digital values. With a 12 MHz crystal, each conversion takes 22 μs. This feature is only available on the 8X95BHs and 8X97BHs, with the 8X95BHs having 4 multiplexed analog inputs.

Also provided on-chip are a serial port, a Watchdog Timer, and a pulse-width modulated output signal.



270090–50

Figure 1. MCS®-96 Block Diagram



0FFH 0F0H	POWER-DOWN RAM		255 240
0EFH 1AH	INTERNAL REGISTER FILE (RAM)		239 26
19H 18H	STACK POINTER	STACK POINTER	25 24
17H		PWM_CONTROL	23
16H	IOS1	IOC1	22
15H	IOS0	IOC0	21
14H 13H 12H	RESERVED	RESERVED	20 19 18
11H	SP_STAT	SP_CON	17
10H	IO PORT 2	IO PORT 2	16
0FH	IO PORT 1	IO PORT 1	15
0EH	IO PORT 0	BAUD_RATE	14
0DH	TIMER2 (HI)	RESERVED	13
0CH	TIMER2 (LO)		12
0BH	TIMER1 (HI)		11
0AH	TIMER1 (LO)	WATCHDOG	10
09H	INT_PENDING	INT_PENDING	9
08H	INT_MASK	INT_MASK	8
07H	SBUF (RX)	SBUF (TX)	7
06H	HSI_STATUS	HSO_COMMAND	6
05H	HSI_TIME (HI)	HSO_TIME (HI)	5
04H	HSI_TIME (LO)	HSO_TIME (LO)	4
03H	AD_RESULT (HI)	HSI_MODE	3
02H	AD_RESULT (LO)	AD_COMMAND	2
01H	R0 (HI)	R0 (HI)	1
00H	R0 (LO)	R0 (LO)	0
	(WHEN READ)	(WHEN WRITTEN)	

EXTERNAL MEMORY OR I/O	FFFFH 4000H
INTERNAL PROGRAM STORAGE ROM/EPROM OR EXTERNAL MEMORY	2080H
RESERVED	2072H – 207FH
SIGNATURE WORD	2070H – 2071H
RESERVED	2030H – 206FH
SECURITY KEY	2020H – 202FH
RESERVED	201CH – 201FH
SELF JUMP OPCODE (27H FEH)	201AH – 201BH
RESERVED	2019H
CHIP CONFIGURATION BYTE	2018H
RESERVED	2012H – 2017H
INTERRUPT VECTORS	2000H
PORT 4	1FFFH
PORT 3	1FFEH
EXTERNAL MEMORY OR I/O	0100H
INTERNAL RAM REGISTER FILE STACK POINTER SPECIAL FUNCTION REGISTERS (WHEN ACCESSED AS DATA MEMORY)	00FFH 0000H

270090-6

Figure 2. Memory Map

PACKAGING

The 8096BH is available in 48-pin, 64-pin, and 68-pin packages, with and without A/D, and with and without on-chip ROM or EPROM. The 8096BH numbering system is shown in Figure 3. Figures 4–9 show the pinouts for the 48-, 64- and 68-pin packages. The 48-pin version is offered in a Dual-In-Line package while the 68-pin versions come in a Plastic Leaded Chip Carrier (PLCC), a Pin Grid Array (PGA) or a Type "B" Leadless Chip Carrier.

	Factory Masked ROM			CPU			User Programmable					
							EPROM			OTP		
	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin
ANALOG	8397BH	8397BH	8395BH	8097BH	8097BH	8095BH	8797BH		8795BH	8797BH	8797BH	
NO ANALOG	8396BH			8096BH								

Figure 3. MCS®96 Packaging—8X9XBH

NOTES:

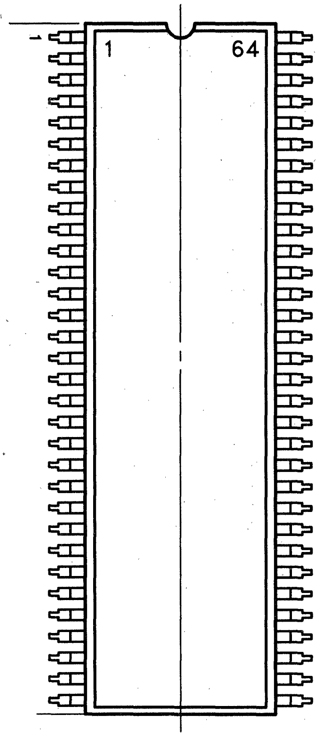
1. 48-pin devices have four Analog Input pins.
2. 64-pin devices have all 48-pin device features plus the following:
   Four additional Analog Input channels
   One additional Quasi-Bidirectional 8-bit Parallel Port
   Four additional Port 2 pins with multiplexed features
      Timer 2 Clock Source pin
      Timer 2 Reset pin
      Two additional quasi-bidirectional port pins
3. 68-pin devices have 48 and 64-pin features plus the following:
   Dynamic Buswidth sizing (8 or 16-bit bus)
   Dedicated System Clock Output (CLKOUT)
   INST pin for memory expansion
   Non-Maskable Interrupt for debugging
4. Package Designators:
   N = PLCC
   C = Ceramic DIP
   A = Ceramic Pin Grid Array
   P = Plastic DIP
   R = Ceramic LCC
   U = Shrink DIP

PGA/ LCC	PLCC	Description	PGA/ LCC	PLCC	Description	PGA/ LCC	PLCC	Description
1	9	ACH7/P0.7/PMOD.3	24	54	AD6/P3.6	47	31	P1.6
2	8	ACH6/P0.6/PMOD.2	25	53	AD7/P3.7	48	30	P1.5
3	7	ACH2/P0.2	26	52	AD8/P4.0	49	29	HSO.1
4	6	ACH0/P0.0	27	51	AD9/P4.1	50	28	HSO.0
5	5	ACH1/P0.1	28	50	AD10/P4.2	51	27	HSO.5/HSI.3
6	4	ACH3/P0.3	29	49	AD11/P4.3	52	26	HSO.4/HSI.2
7	3	NMI	30	48	AD12/P4.4	53	25	HSI.1
8	2	$\overline{\text{EA}}$	31	47	AD13/P4.5	54	24	HSI.0
9	1	VCC	32	46	AD14/P4.6	55	23	P1.4
10	68	VSS	33	45	AD15/P4.7	56	22	P1.3
11	67	XTAL1	34	44	T2CLK/P2.3	57	21	P1.2
12	66	XTAL2	35	43	READY	58	20	P1.1
13	65	CLKOUT	36	42	T2RST/P2.4	59	19	P1.0
14	64	BUSWIDTH	37	41	$\overline{\text{BHE}}/\overline{\text{WRH}}$	60	18	TXD/P2.0/PVER/SALE
15	63	INST	38	40	$\overline{\text{WR}}/\overline{\text{WRL}}$	61	17	RXD/P2.1/PALE
16	62	ALE/$\overline{\text{ADV}}$	39	39	PWM/P2.5/$\overline{\text{PDO}}/\overline{\text{SPROG}}$	62	16	$\overline{\text{RESET}}$
17	61	$\overline{\text{RD}}$	40	38	P2.7	63	15	EXTINT/P2.2/$\overline{\text{PROG}}$
18	60	AD0/P3.0	41	37	VPP	64	14	VPD
19	59	AD1/P3.1	42	36	VSS	65	13	VREF
20	58	AD2/P3.2	43	35	HSO.3	66	12	ANGND
21	57	AD3/P3.3	44	34	HSO.2	67	11	ACH4/P0.4/PMOD.0
22	56	AD4/P3.4	45	33	P2.6	68	10	ACH5/P0.5/PMOD.1
23	55	AD5/P3.5	46	32	P1.7			

Figure 4a. PGA, PLCC and LCC Function Pinouts

	Description		Description
1	$\overline{\text{EA}}$	33	HS0.3
2	ACH3/P0.3	34	V_{SS}
3	ACH1/P0.1	35	V_{PP}
4	ACH0/P0.0	36	P2.7
5	ACH2/P0.2	37	PWM/P2.5/ $\overline{\text{PD0}}$/$\overline{\text{SPROG}}$
6	ACH6/P0.6/PMOD.2		
7	ACH7/P0.7/PMOD.3	38	$\overline{\text{WR}}$/$\overline{\text{WRL}}$
8	ACH5/P0.5/PMOD.1	39	$\overline{\text{BHE}}$/$\overline{\text{WRH}}$
9	ACH4/P0.4/PMOD.0	40	T2RST/P2.4
10	ANGND	41	READY
11	V_{REF}	42	T2CLK/P2.3
12	V_{PD}	43	AD15/P4.7
13	EXINT/P2.2/$\overline{\text{PROG}}$	44	AD14/P4.6
14	$\overline{\text{RESET}}$	45	AD13/P4.5
15	RXD/P2.1/PALE	46	AD12/P4.4
16	TXD/P2.0/ PVER/SALE	47	AD11/P4.3
		48	AD10/P4.2
17	P1.0	49	AD9/P4.1
18	P1.1	50	AD8/P4.0
19	P1.2	51	AD7/P3.7
20	P1.3	52	AD6/P3.6
21	P1.4	53	AD5/P3.5
22	HSI.0	54	AD4/P3.4
23	HSI.1	55	AD3/P3.3
24	HS0.4/HSI.2	56	AD2/P3.2
25	HS0.5/HSI.3	57	AD1/P3.1
26	HS0.0	58	AD0/P3.0
27	HS0.1	59	$\overline{\text{RD}}$
28	P1.5	60	ALE/$\overline{\text{ADV}}$
29	P1.6	61	XTAL2
30	P1.7	62	XTAL1
31	P2.6	63	V_{SS}
32	HS0.2	64	V_{CC}

Figure 4b. Shrink-DIP Function Pinouts



270090-56

Figure 5. Shrink-DIP Package

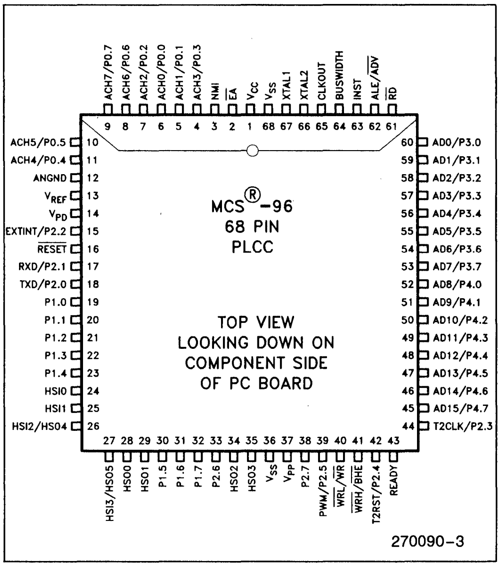


Figure 6. 68-Pin Package (PLCC - Top View)

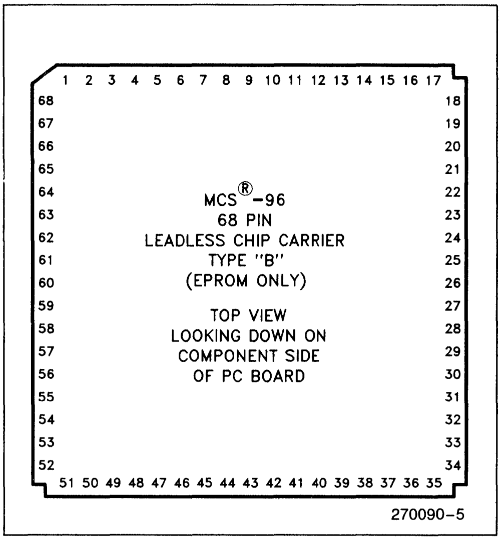


Figure 8. 68-Pin Package (LCC - Top View)

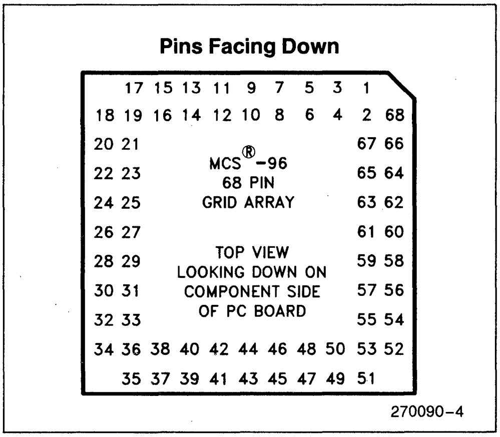


Figure 7. 68-Pin Package (Pin Grid Array - Top View)

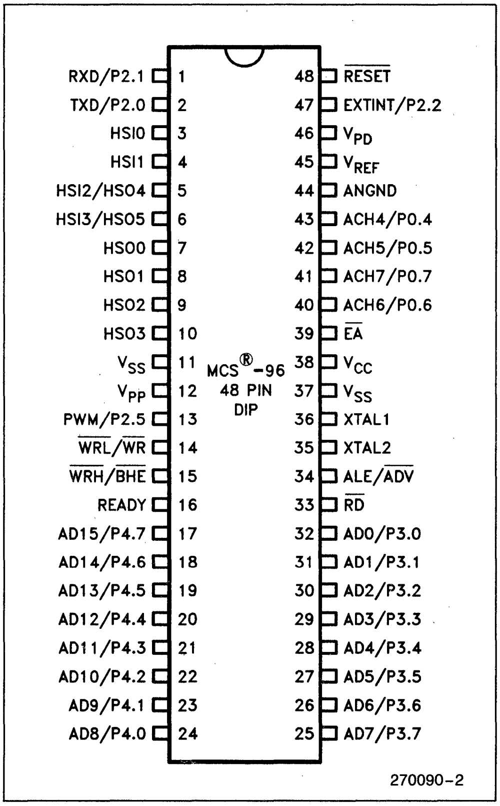


Figure 9. 48-Pin Package

PIN DESCRIPTIONS

Symbol	Name and Function
V_{CC}	Main supply voltage (5V).
V_{SS}	Digital circuit ground (0V). There are two V_{SS} pins, both of which must be connected.
V_{PD}	RAM standby supply voltage (5V). This voltage must be present during normal operation. In a Power Down condition (i.e. V_{CC} drops to zero), if $\overline{\text{RESET}}$ is activated before V_{CC} drops below spec and V_{PD} continues to be held within spec., the top 16 bytes in the Register File will retain their contents. $\overline{\text{RESET}}$ must be held low during the Power Down and should not be brought high until V_{CC} is within spec and the oscillator has stabilized.
V_{REF}	Reference voltage for the A/D converter (5V). V_{REF} is also the supply voltage to the analog portion of the A/D converter and the logic used to read Port 0.
ANGND	Reference ground for the A/D converter. Must be held at nominally the same potential as V_{SS}.
V_{PP}	Programming voltage for the EPROM devices. It should be +12.75V for programming. This pin must be left floating in the application circuit.
XTAL1	Input of the oscillator inverter and of the internal clock generator.
XTAL2	Output of the oscillator inverter.
CLKOUT*†	Output of the internal clock generator. The frequency of CLKOUT is ⅓ the oscillator frequency. It has a 33% duty cycle.
$\overline{\text{RESET}}$	Reset input to the chip. Input low for a minimum 10 XTAL1 cycles to reset the chip. The subsequent low-to-high transition re-synchronizes CLKOUT and commences a 10-state-time sequence in which the PSW is cleared, a byte read from 2018H loads CCR, and a jump to location 2080H is executed. Input high for normal operation. $\overline{\text{RESET}}$ has an internal pullup.
BUSWIDTH*†	Input for bus width selection. If CCR bit 1 is a one, this pin selects the bus width for the bus cycle in progress. If BUSWIDTH is a 1, a 16-bit bus cycle occurs. If BUSWIDTH is a 0 an 8-bit cycle occurs. If CCR bit 1 is a 0, the bus is always an 8-bit bus. If this pin is left unconnected, it will rise to V_{CC}.
NMI*†	A positive transition causes a vector to external memory location 0000H. External memory from 00H through 0FFH is reserved for Intel development systems.
INST*†	Output high during an external memory read indicates the read is an instruction fetch. INST is valid throughout the bus cycle.
$\overline{\text{EA}}$	Input for memory select (External Access). $\overline{\text{EA}}$ equal to a TTL-high causes memory accesses to locations 2000H through 3FFFH to be directed to on-chip ROM/EPROM. $\overline{\text{EA}}$ equal to a TTL-low causes accesses to these locations to be directed to off-chip memory. $\overline{\text{EA}}$ = +12.5V causes execution to begin in the Programming Mode. $\overline{\text{EA}}$ has an internal pulldown, so it goes to 0 unless driven otherwise.
ALE/$\overline{\text{ADV}}$	Address Latch Enable or Address Valid output, as selected by CCR. Both pin options provide a latch to demultiplex the address from the address/data bus. When the pin is $\overline{\text{ADV}}$, it goes inactive high at the end of the bus cycle. $\overline{\text{ADV}}$ can be used as a chip select for a single external RAM memory. ALE/$\overline{\text{ADV}}$ is activated only during external memory accesses.
$\overline{\text{RD}}$	Read signal output to external memory. $\overline{\text{RD}}$ is activated only during external memory reads.
$\overline{\text{WR}}$/$\overline{\text{WRL}}$	Write and Write Low output to external memory, as selected by the CCR. $\overline{\text{WR}}$ will go low for every external write, while $\overline{\text{WRL}}$ will go low only for external writes where an even byte is being written. $\overline{\text{WR}}$/$\overline{\text{WRL}}$ is activated only during external memory writes.
$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	Bus High Enable or Write High output to external memory, as selected by the CCR. $\overline{\text{BHE}}$ = 0 selects the bank of memory that is connected to the high byte of the data bus. A0 = 0 selects the bank of memory that is connected to the low byte of the data bus. Thus accesses to a 16-bit wide memory can be to the low byte only (A0 = 0, $\overline{\text{BHE}}$ = 1), to the high byte only (A0 = 1, $\overline{\text{BHE}}$ = 0), or both bytes (A0 = 0, $\overline{\text{BHE}}$ = 0). If the $\overline{\text{WRH}}$ function is selected, the pin will go low if the bus cycle is writing to an odd memory location.

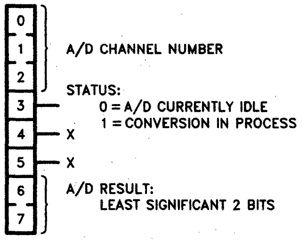
*Not available on Shrink-DIP package
†Not available on 48-pin device

PIN DESCRIPTIONS (Continued)

Symbol	Name and Function
READY	Ready input to lengthen external memory cycles, for interfacing to slow or dynamic memory, or for bus sharing. If the pin is high, CPU operation continues in a normal manner. If the pin is low prior to the falling edge of CLKOUT, the memory controller goes into a wait mode until the next positive transition in CLKOUT occurs with READY high. The bus cycle can be lengthened by up to 1 μs. When the external memory is not being used, READY has no effect. Internal control of the number of wait states inserted into a bus cycle held not ready is available through configuration of CCR. READY has a weak internal pullup, so it goes to 1 unless externally pulled low.
HSI	Inputs to High Speed Input Unit. Four HSI pins are available: HSI.0, HSI.1, HSI.2, and HSI.3. Two of them (HSI.2 and HSI.3) are shared with the HSO Unit. The HSI pins are also used as inputs by EPROM devices in Programming Mode.
HSO	Outputs from High Speed Output Unit. Six HSO pins are available: HSO.0, HSO.1, HSO.2, HSO.3, HSO.4, and HSO.5. Two of them (HSO.4 and HSO.5) are shared with the HSI Unit.
Port 0‡	8-bit high impedance input-only port. These pins can be used as digital inputs and/or as analog inputs to the on-chip A/D converter. These pins are also a mode input to EPROM devices in the Programming Mode.
Port 1†	8-bit quasi-bidirectional I/O port.
Port 2†	8-bit multi-functional port. Six of its pins are shared with other functions in the 8096BH, the remaining 2 are quasi-bidirectional. These pins are also used to input and output control signals on EPROM devices in Programming Mode.
Ports 3 and 4	8-bit bi-directional I/O ports with open drain outputs. These pins are shared with the multiplexed address/data bus which has strong internal pullups. Ports 3 and 4 are also used as a command, address and data path by EPROM devices operating in the Programming Mode. When used as ports, pullups to V_{CC} may be needed.

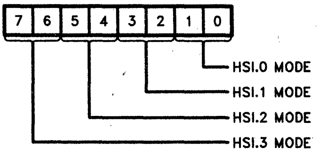
†Not available on 48-pin device
‡Port 0.0.1.2.3 not available on 48-pin device

A/D Result LO (02H)



270090-21

HSI_Mode (03H)

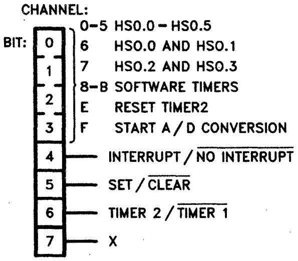


WHERE EACH 2-BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

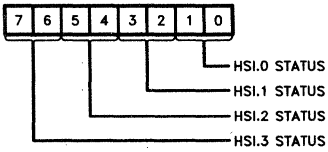
270090-22

HSO Command (06H)



270090-23

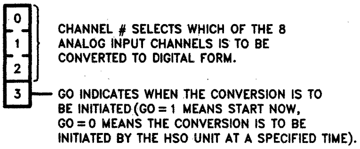
HSI_Status (06H)



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT INDICATES THE CURRENT STATUS OF THE PIN.

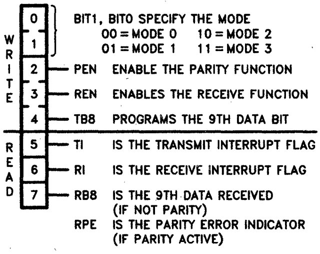
270090-25

A/D Command (02H)


0
1
2
3
CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.
GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).

270090-24

SPCON/SPSTAT (11H)


WRITE
READ
0
1
2
3
4
5
6
7
BIT1, BIT0 SPECIFY THE MODE
00 = MODE 0 10 = MODE 2
01 = MODE 1 11 = MODE 3
PEN ENABLE THE PARITY FUNCTION
REN ENABLES THE RECEIVE FUNCTION
TB8 PROGRAMS THE 9TH DATA BIT
TI IS THE TRANSMIT INTERRUPT FLAG
RI IS THE RECEIVE INTERRUPT FLAG
RB8 IS THE 9TH DATA RECEIVED (IF NOT PARITY)
RPE IS THE PARITY ERROR INDICATOR (IF PARITY ACTIVE)

270090-26

Baud Rate Calculations

Using XTAL1:

Mode 0: $\text{Baud Rate} = \frac{\text{XTAL1 frequency}}{4^*(B + 1)}; B \neq 0$

Others: $\text{Baud Rate} = \frac{\text{XTAL1 frequency}}{64^*(B + 1)}$

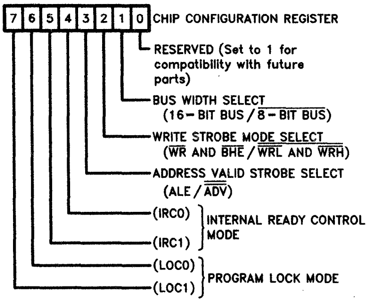
Using T2CLK:

Mode 0: $\text{Baud Rate} = \frac{\text{T2CLK frequency}}{B}; B \neq 0$

Others: $\text{Baud Rate} = \frac{\text{T2CLK frequency}}{16^*B}; B \neq 0$

Note that B cannot equal 0, except when using XTAL1 in other than Mode 0.

Chip Configuration


7 6 5 4 3 2 1 0
CHIP CONFIGURATION REGISTER
RESERVED (Set to 1 for compatibility with future parts)
BUS WIDTH SELECT (16-BIT BUS / $\overline{\text{8-BIT BUS}}$)
WRITE STROBE MODE SELECT ($\overline{\text{WR}}$ AND $\overline{\text{BHE}}$ / $\overline{\text{WRL}}$ AND $\overline{\text{WRH}}$)
ADDRESS VALID STROBE SELECT (ALE / $\overline{\text{ADV}}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE

270090-32

Interrupt Pending Register

(LOCATION 09H)

0	TIMER OVERFLOW
1	A/D COMPLETION
2	HSI DATA AVAILABLE
3	HSO EVENT
4	HSI BIT 0
5	SOFTWARE TIMERS
6	SERIAL I/O
7	EXTERNAL INTERRUPT

270090–55

PSW Register

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Z	N	V	VT	C	—	I	ST	<Interrupt Mask Reg>							

IOC0 (15H)*

0	HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1	TIMER 2 RESET EACH WRITE
2	HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3	TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4	HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5	TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6	HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7	TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$

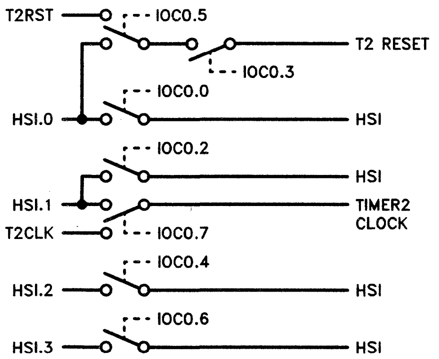
270090–30

IOS0 (15H)

0	HSO.0 CURRENT STATE
1	HSO.1 CURRENT STATE
2	HSO.2 CURRENT STATE
3	HSO.3 CURRENT STATE
4	HSO.4 CURRENT STATE
5	HSO.5 CURRENT STATE
6	CAM OR HOLDING REGISTER IS FULL
7	HSO HOLDING REGISTER IS FULL

270090–27

IOC0 (15H)*



270090–29

IOC1 (16H)

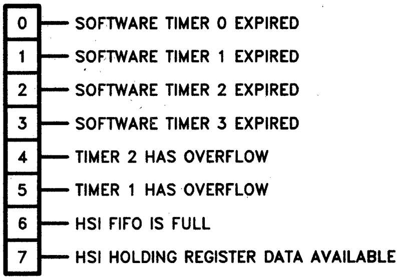
0	SELECT PWM / $\overline{\text{SELECT P2.5}}$
1	EXTERNAL INTERRUPT ACH7 / $\overline{\text{EXTINT}}$
2	TIMER 1 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
3	TIMER 2 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
4	HSO.4 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
5	SELECT TXD / $\overline{\text{SELECT P2.0}}$
6	HSO.5 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
7	HSI INTERRUPT FIFO FULL / $\overline{\text{HOLDING REGISTER LOADED}}$

270090–31

*See Errata section

Vector	Vector Location		Priority
	(High Byte)	(Low Byte)	
Software Trap	2011H	2010H	Not Applicable
Extint	200FH	200EH	7 (Highest)
Serial Port	200DH	200CH	6
Software Timers	200BH	200AH	5
HSI.0	2009H	2008H	4
High Speed Outputs	2007H	2006H	3
HSI Data Available	2005H	2004H	2
A/D Conversion Complete	2003H	2002H	1
Timer Overflow	2001H	2000H	0 (Lowest)

IOS1 (16H)


0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE

270090-28

ELECTRICAL CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias ....0°C to +70°C

Storage Temperature ....−40°C to +150°C

Voltage from $\overline{EA}$ or V_{PP}
to V_{SS} or ANGND ....−0.3V to +13.0V

Voltage from Any Other Pin to
V_{SS} or ANGND ....−0.3V to +7.0V*

Average Output Current from Any Pin ....10 mA

Power Dissipation..........................1.5W

*This includes V_{PP} on ROM and CPU only devices.

**Notice: Stresses 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 specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

NOTICE: Specifications contained within the following tables are subject to change.

OPERATING CONDITIONS

Symbol	Parameter	Min	Max	Units
T_A	Ambient Temperature Under Bias	0	+70	°C
V_{CC}	Digital Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	6.0	12	MHz
V_{PD}	Power-Down Supply Voltage	4.50	5.50	V

NOTE:
ANGND and V_{SS} should be nominally at the same potential.

D.C. CHARACTERISTICS (Test Conditions: V_{CC}, V_{REF}, V_{PD}, V_{PP}, V_{EA} = 5.0V ±0.5V; F_{OSC} = 6.0 MHz; T_A = 0°C to 70°C; V_{SS}, ANGND = 0V)

Symbol	Parameter	Min	Max	Units	Test Conditions
I_{CC}	V_{CC} Supply Current (0°C ≤ T_A ≤ 70°C)		240	mA	All Outputs Disconnected.
I_{CC1}	V_{CC} Supply Current (T_A = 70°C)		185	mA	
I_{PD}	V_{PD} Supply Current		1	mA	Normal operation and Power-Down.
I_{REF}	V_{REF} Supply Current		8	mA	
V_{IL}	Input Low Voltage	−0.3	+0.8	V	
V_{IH}	Input High Voltage (Except $\overline{RESET}$, NMI, XTAL1)	2.0	V_{CC} +0.5	V	
V_{IH1}	Input High Voltage, $\overline{RESET}$ Rising	2.4	V_{CC} +0.5	V	
V_{IH2}	Input High Voltage, $\overline{RESET}$ Falling Hysteresis	2.1	V_{CC} +0.5	V	
V_{IH3}	Input High Voltage, NMI, XTAL1	2.2	V_{CC} +0.5	V	
I_{LI}	Input Leakage Current to each pin of HSI, P3, P4, and to P2.1.		±10	μA	V_{in} = 0 to V_{CC}
I_{LI1}	D.C. Input Leakage Current to each pin of P0		+3	μA	V_{in} = 0 to V_{CC}
I_{IH}	Input High Current to $\overline{EA}$		100	μA	V_{IH} = 2.4V
I_{IL}	Input Low Current to each pin of P1, and to P2.6, P2.7.		−125	μA	V_{IL} = 0.45V
I_{IL1}	Input Low Current to $\overline{RESET}$	−0.25	−2	mA	V_{IL} = 0.45V
I_{IL2}	Input Low Current P2.2, P2.3, P2.4, READY, BUSWIDTH		−50	μA	V_{IL} = 0.45V
V_{OL}	Output Low Voltage on Quasi-Bidirectional port pins and P3, P4 when used as ports		0.45	V	I_{OL} = 0.8 mA (Note 1)
V_{OL1}	Output Low Voltage on Quasi-Bidirectional port pins and P3, P4 when used as ports		0.75	V	I_{OL} = 2.0 mA (Notes 1, 2, 3)
V_{OL2}	Output Low Voltage on Standard Output pins, $\overline{RESET}$ and Bus/Control Pins		0.45	V	I_{OL} = 2.0 mA (Notes 1, 2, 3)

D.C. CHARACTERISTICS (Test Conditions: V_{CC}, V_{REF}, V_{PD}, V_{PP}, V_{EA} = 5.0V ±0.5V; F_{OSC} = 6.0 MHz; T_A = 0°C to 70°C; V_{SS}, ANGND = 0V) (Continued)

Symbol	Parameter	Min	Max	Units	Test Conditions
V_{OH}	Output High Voltage on Quasi-Bidirectional pins	2.4		V	$I_{OH} = -20\ \mu A$ (Note 1)
V_{OH1}	Output High Voltage on Standard Output pins and Bus/Control pins	2.4		V	$I_{OH} = -200\ \mu A$ (Note 1)
I_{OH3}	Output High Current on $\overline{RESET}$	−50		μA	V_{OH} = 2.4V
C_S	Pin Capacitance (Any Pin to V_{SS})		10	pF	fTEST = 1.0 MHz

NOTES:

1. Quasi-bidirectional pins include those on P1, for P2.6 and P2.7. Standard Output Pins include TXD, RXD (Mode 0 only), PWM, and HSO pins. Bus/Control pins include CLKOUT, ALE, $\overline{BHE}$, $\overline{RD}$, $\overline{WR}$, INST and AD0–15.
2. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V.
   I_{OL} on quasi-bidirectional pins and Ports 3 and 4 when used as ports: 4.0 mA
   I_{OL} on standard output pins and $\overline{RESET}$: 8.0 mA
   I_{OL} on Bus/Control pins: 2.0 mA
3. During normal (non-transient) operation the following limits apply:
   Total I_{OL} on Port 1 must not exceed 8.0 mA.
   Total I_{OL} on P2.0, P2.6, $\overline{RESET}$ and all HSO pins must not exceed 15 mA.
   Total I_{OL} on Port 3 must not exceed 10 mA.
   Total I_{OL} on P2.5, P2.7, and Port 4 must not exceed 20 mA.

A.C. CHARACTERISTICS V_{CC}, V_{PD} = 4.5 to 5.5V; T_A = 0°C to 70°C; f_{OSC} = 6.0 to 12.0 MHz

Test Conditions: Load Capacitance on Output Pins = 80 pF
Oscillator Frequency = 10 MHz

TIMING REQUIREMENTS (Other system components must meet these specs.)

Symbol	Parameter	Min	Max	Units
T_{CLYX}(2, 3)	READY Hold after CLKOUT Edge	0(1)		ns
T_{LLYV}	End of ALE/$\overline{ADV}$ to READY Valid		2Tosc−70	ns
T_{LLYH}	End of ALE/$\overline{ADV}$ to READY High	2Tosc+40	4Tosc−80	ns
T_{YLYH}	Non-Ready Time		1000	ns
T_{AVDV}(4)	Address Valid to Input Data Valid		5Tosc−120	ns
T_{RLDV}	$\overline{RD}$ Active to Input Data Valid		3Tosc−100	ns
T_{RHDX}	Data Hold after $\overline{RD}$ Inactive	0		ns
T_{RHDZ}	$\overline{RD}$ Inactive to Input Data Float	0	Tosc−25	ns
T_{AVGV}(2, 4)	Address Valid to BUSWIDTH Valid		2 Tosc −125	ns
T_{LLGX}(2, 3)	BUSWIDTH Hold after ALE/$\overline{ADV}$ Low	Tosc +40		ns
T_{LLGV}(2, 3)	ALE/$\overline{ADV}$ Low to BUSWIDTH Valid		Tosc −75	ns
T_{RLPV}	Reset Low to Ports Valid		10Tosc	ns

NOTES:

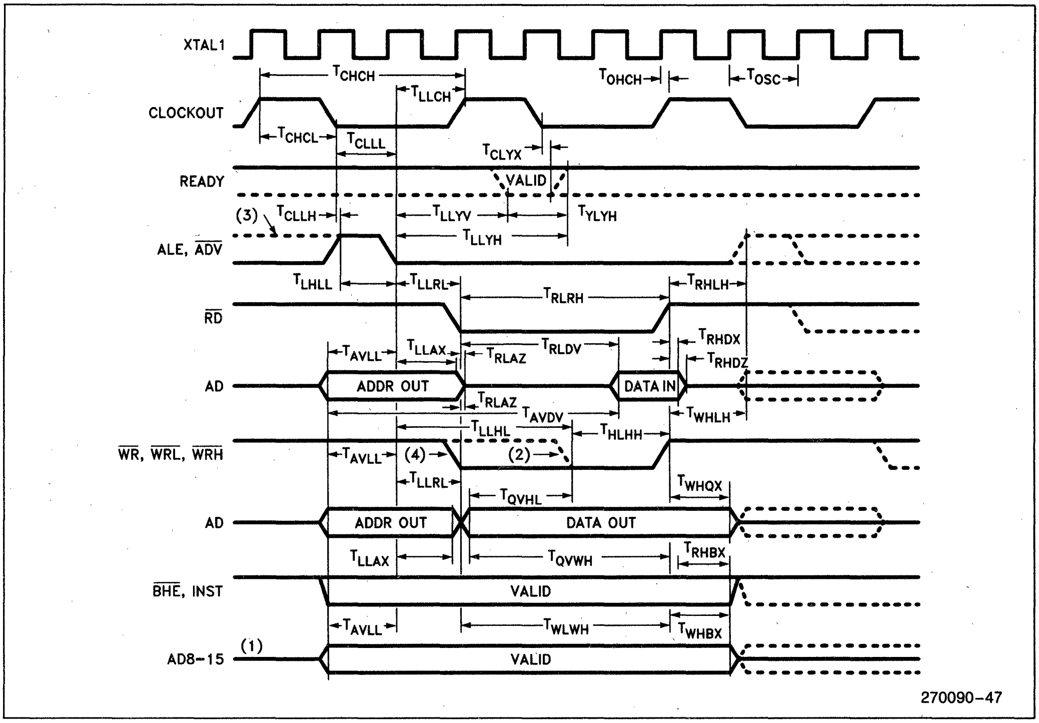
1. If the 48-pin or 64-pin device is being used then this timing can be generated by assuming that the CLKOUT falling edge has occurred at 2Tosc+55 (TLLCH(max) + TCHCL(max)) after the falling edge of ALE.
2. Pins not bonded out on 64-pin devices
3. Pins not bonded out on 48-pin devices.
4. The term "Address Valid" applies to AD0–15, $\overline{BHE}$ and INST.

TIMING RESPONSES (MCS-96 devices meet these specs.)

Symbol	Parameter	Min	Max	Units
F_{XTAL}	Oscillator Frequency	6.0	12.0	MHz
T_{OSC}	Oscillator Period	83	166	ns
T_{OHCH}	XTAL1 Rising Edge to Clockout Rising Edge	0(4)	120(4)	ns
T_{CHCH}(1, 4)	CLKOUT Period(3)	3Tosc(3)	3Tosc(3)	ns
T_{CHCL}(1, 4)	CLKOUT High Time	Tosc − 35	Tosc + 10	ns
T_{CLLH}(1, 4)	CLKOUT Low to ALE High	−20	+25	ns
T_{LLCH}(4)	ALE/$\overline{\text{ADV}}$ Low to CLKOUT High(1)	Tosc − 25	Tosc + 45	ns
T_{LHLL}	ALE/$\overline{\text{ADV}}$ High Time	Tosc − 30	Tosc + 35(5)	ns
T_{AVLL}(6)	Address Setup to End of ALE/$\overline{\text{ADV}}$	Tosc − 50		ns
T_{RLAZ}(7)	$\overline{\text{RD}}$ or $\overline{\text{WR}}$ Low to Address Float	Typ. = 0	10	ns
T_{LLRL}	End of ALE/$\overline{\text{ADV}}$ to $\overline{\text{RD}}$ or $\overline{\text{WR}}$ Active	Tosc − 40		ns
T_{LLAX}(7)	Address Hold after End of ALE/$\overline{\text{ADV}}$	Tosc − 40		ns
T_{WLWH}	$\overline{\text{WR}}$ Pulse Width	3Tosc − 35		ns
T_{QVWH}	Output Data Valid to End of $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$	3Tosc − 60		ns
T_{WHQX}	Output Data Hold after $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$	Tosc − 50		ns
T_{WHLH}	End of $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$ to ALE/$\overline{\text{ADV}}$ High	Tosc − 75		ns
T_{RLRH}	$\overline{\text{RD}}$ Pulse Width	3Tosc − 30		ns
T_{RHLH}	End of $\overline{\text{RD}}$ to ALE/$\overline{\text{ADV}}$ High	Tosc − 45		ns
T_{CLLL}(4)	CLOCKOUT Low(1) to ALE/$\overline{\text{ADV}}$ Low	Tosc − 40	Tosc + 35	ns
T_{RHBX}(4)	$\overline{\text{RD}}$ High to INST(1), $\overline{\text{BHE}}$, AD8-15 Inactive	Tosc − 25	Tosc + 30	ns
T_{WHBX}(4)	$\overline{\text{WR}}$ High to INST(1), $\overline{\text{BHE}}$, AD8-15 Inactive	Tosc − 50	Tosc + 100	ns
T_{HLHH}	$\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ High	2Tosc − 35	2Tosc + 40	ns
T_{LLHL}	ALE/$\overline{\text{ADV}}$ Low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low	2Tosc − 30	2Tosc + 55	ns
T_{QVHL}	Output Data Valid to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low	Tosc − 60		ns

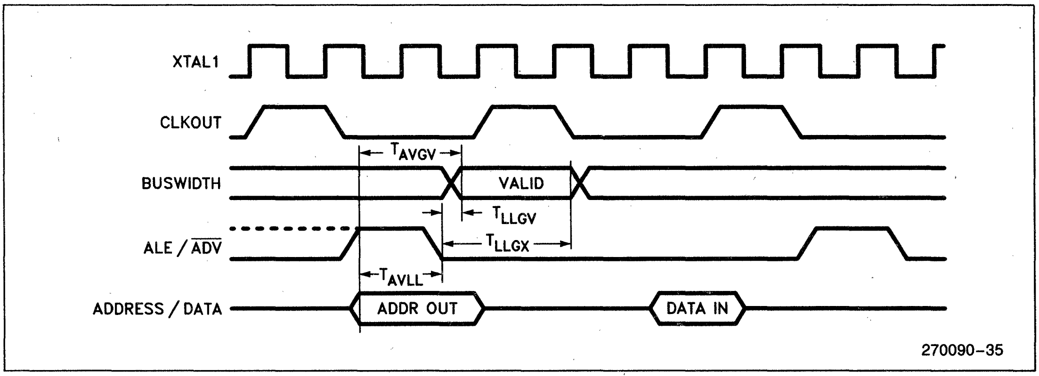
NOTES:
1. Pins not bonded out on 64-pin devices.
2. If more than one wait state is desired, add 3Tosc for each additional wait state.
3. CLKOUT is directly generated as a divide by 3 of the oscillator. The period will be 3Tosc ± 10 ns if Tosc is constant and the rise and fall times on XTAL1 are less than 10 ns.
4. Pins not bonded out on 48-pin devices.
5. Max spec applies only to ALE. Min spec applies to both ALE and $\overline{\text{ADV}}$.
6. The term "Address Valid" applies to AD0−15, $\overline{\text{BHE}}$ and INST.
7. The term " Address" in this definition applies to AD0−7 for 8-bit cycles, and AD0−15 for 16-bit cycles.

WAVEFORM

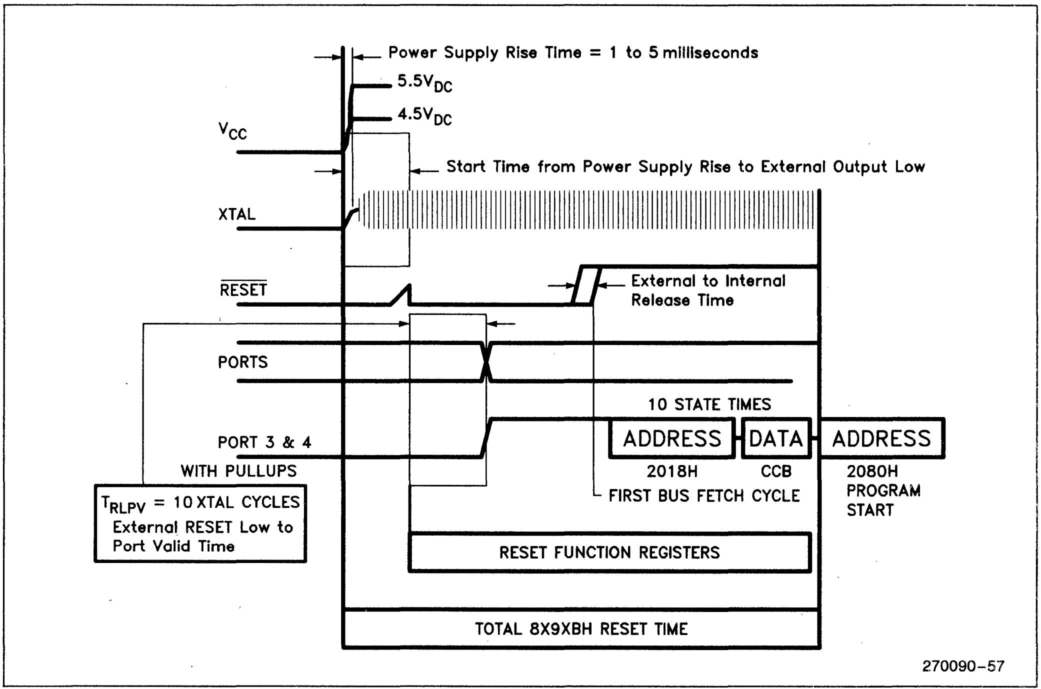


NOTES:
(1) 8-bit bus only.
(2) 8-bit or 16-bit bus and write strobe mode selected.
(3) When $\overline{\text{ADV}}$ selected.
(4) 8- or 16-bit bus and no write strobe mode selected.

WAVEFORM—BUSWIDTH PIN*



*Buswidth is not bonded out on 48- and 64-pin devices.


Power Supply Rise Time = 1 to 5 milliseconds
5.5V DC
4.5V DC
V CC
Start Time from Power Supply Rise to External Output Low
XTAL
RESET
External to Internal Release Time
PORTS
10 STATE TIMES
PORT 3 & 4 WITH PULLUPS
ADDRESS
DATA
ADDRESS
2018H
CCB
2080H PROGRAM START
FIRST BUS FETCH CYCLE
T RLPV = 10 XTAL CYCLES External RESET Low to Port Valid Time
RESET FUNCTION REGISTERS
TOTAL 8X9XBH RESET TIME

270090-57

A.C. CHARACTERISTICS—SERIAL PORT—SHIFT REGISTER MODE

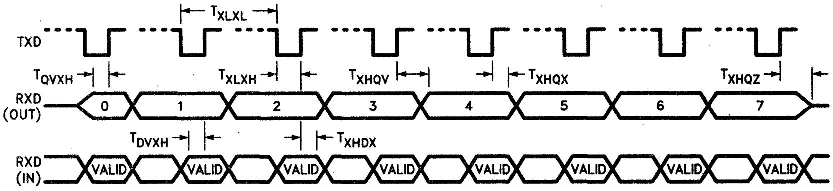
SERIAL PORT TIMING—SHIFT REGISTER MODE

Test Conditions: T_A = 0°C to +70°C; V_{CC} = 5V ±10%; V_{SS} = 0V; Load Capacitance = 80 pF

Symbol	Parameter	Min	Max	Units
T_{XLXL}	Serial Port Clock Period	$8T_{OSC}$		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge	$4T_{OSC} - 50$	$4T_{OSC} + 50$	ns
T_{QVXH}	Output Data Setup to Clock Rising Edge	$3T_{OSC}$		ns
T_{XHQX}	Output Data Hold After Clock Rising Edge	$2T_{OSC} - 50$		ns
T_{XHQV}	Next Output Data Valid After Clock Rising Edge		$2T_{OSC} + 50$	ns
T_{DVXH}	Input Data Setup to Clock Rising Edge	$2T_{OSC} + 200$		ns
T_{XHDX}	Input Data Hold After Clock Rising Edge	0		ns
T_{XHQZ}	Last Clock Rising to Output Float		$5T_{OSC}$	ns

WAVEFORM—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT WAVEFORM—SHIFT REGISTER MODE

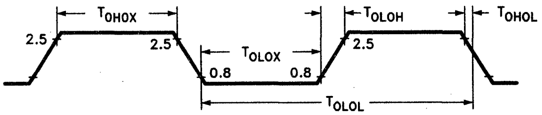


270090-36

EXTERNAL CLOCK DRIVE

Symbol	Parameter	Min	Max	Units
$1/T_{OLOL}$	Oscillator Frequency	6	12	MHz
T_{OHOX}	High Time	25		ns
T_{OLOX}	Low Time	25		ns
T_{OLOH}	Rise Time		15	ns
T_{OHOL}	Fall Time		15	ns

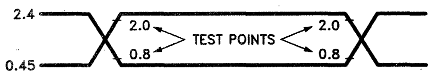
EXTERNAL CLOCK DRIVE WAVEFORMS



270090-48

An external oscillator may encounter as much as a 100 pF load at XTAL1 when it starts up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{IL} and V_{IH} specifications the capacitance will not exceed 20 pF.

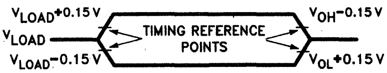
A.C. TESTING INPUT, OUTPUT WAVEFORM



270090-49

A.C. Testing inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0". Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

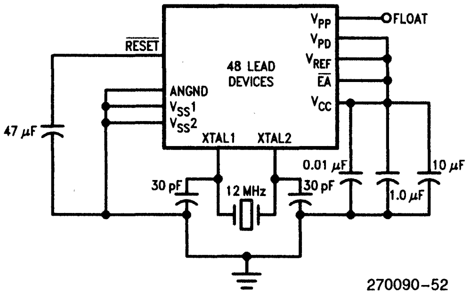
FLOAT WAVEFORM



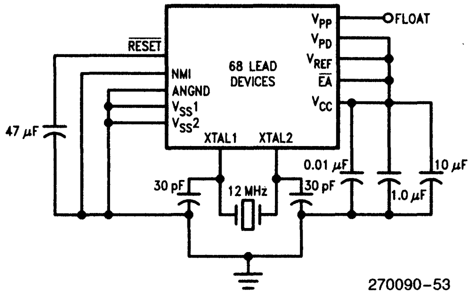
270090-51

For Timing Purposes a Port Pin is no Longer Floating when a 100 mV change from Load Voltage Occurs, and Begins to Float when a 100 mV change from the Loaded V_{OH}/V_{OL} Level occurs $I_{OL}/I_{OH} \geq \pm 15$ mA.

Minimum Hardware Configuration Circuits



270090-52



270090-53

A/D CONVERTER SPECIFICATIONS

A/D Converter operation is verified only on the 8097BH, 8397BH, 8095BH, 8395BH, 8797BH, 8795BH.

The absolute conversion accuracy is dependent on the accuracy of V_{REF}. The specifications given below assume adherence to the Operating Conditions section of these data sheets. Testing is done at V_{REF} = 5.120V.

OPERATING CONDITIONS

V_{CC}, V_{PD}, V_{REF}	4.5V to 5.5V
V_{SS}, ANGND	0.0V
T_A	0°C to 70°C
F_{OSC}	6.0 to 12.0 MHz
Test Conditions:	
V_{REF}	5.120V

Parameter	Typical*	Minimum	Maximum	Units**	Notes
Resolution		1024 10	1024 10	Levels Bits	
Absolute Error		0	±4	LSBs	
Full Scale Error	−0.5 ±0.5			LSBs	
Zero Offset Error	±0.5			LSBs	
Non-Linearity		0	±4	LSBs	
Differential Non-Linearity		> −1	+2	LSBs	
Channel-to-Channel Matching		0	±1	LSBs	
Repeatability	±0.25			LSBs	
Temperature Coefficients: Offset Full Scale Differential Non-Linearity	 0.009 0.009 0.009			 LSB/°C LSB/°C LSB/°C	
Off Isolation		−60		dB	1, 3
Feedthrough	−60			dB	1
V_{CC} Power Supply Rejection	−60			dB	1
Input Resistance		1K	5K	Ω	
D.C. Input Leakage		0	3.0	μA	
Sample Delay		$3T_{OSC} - 50$	$3T_{OSC} + 50$	ns	2
Sample Time		$12T_{OSC} - 50$	$12T_{OSC} + 50$	ns	
Sampling Capacitor			2	pF	

NOTES:

* These values are expected for most devices at 25°C.

** An "LSB", as used here, is defined in the glossary which follows and has a value of approximately 5 mV.

1. DC to 100 KHz.
2. For starting the A/D with an HSO Command.
3. Multiplexer Break-Before-Make Guaranteed.

EPROM SPECIFICATIONS

A.C. EPROM PROGRAMMING CHARACTERISTICS

Operating Conditions: Load Capacitance = 150 pF, T_A = 25°C ±5°C, V_{CC}, V_{PD}, V_{REF} = 5.0V ± 0.5V, V_{SS}, AGND = 0V, V_{PP} = 12.75V ± 0.25V, $\overline{EA}$ = 11V ± 2.0V, f_{osc} = 6.0 MHz

Symbol	Parameter	Min	Max	Units
T_{AVLL}	ADDRESS/COMMAND Valid to PALE Low	0		T_{osc}
T_{LLAX}	ADDRESS/COMMAND Hold After PALE Low	80		T_{osc}
T_{DVPL}	Output Data Setup Before $\overline{PROG}$ Low	0		T_{osc}
T_{PLDX}	Data Hold After $\overline{PROG}$ Falling	80		T_{osc}
T_{LLLH}	PALE Pulse Width	180		T_{osc}
T_{PLPH}	$\overline{PROG}$ Pulse Width	250 T_{osc}	100 μS + 144 T_{osc}	
T_{LHPL}	PALE High to $\overline{PROG}$ Low	250		T_{osc}
T_{PHLL}	$\overline{PROG}$ High to Next PALE Low	600		T_{osc}
T_{PHDX}	Data Hold After $\overline{PROG}$ High	30		T_{osc}
T_{PHVV}	$\overline{PROG}$ High to PVER/$\overline{PDO}$ Valid	500		T_{osc}
T_{LLVH}	PALE Low to PVER/$\overline{PDO}$ High	100		T_{osc}
T_{PLDV}	$\overline{PROG}$ Low to VERIFICATION/DUMP Data Valid	100		T_{osc}
T_{SHLL}	RESET High to First PALE Low (not shown)	2000		T_{osc}

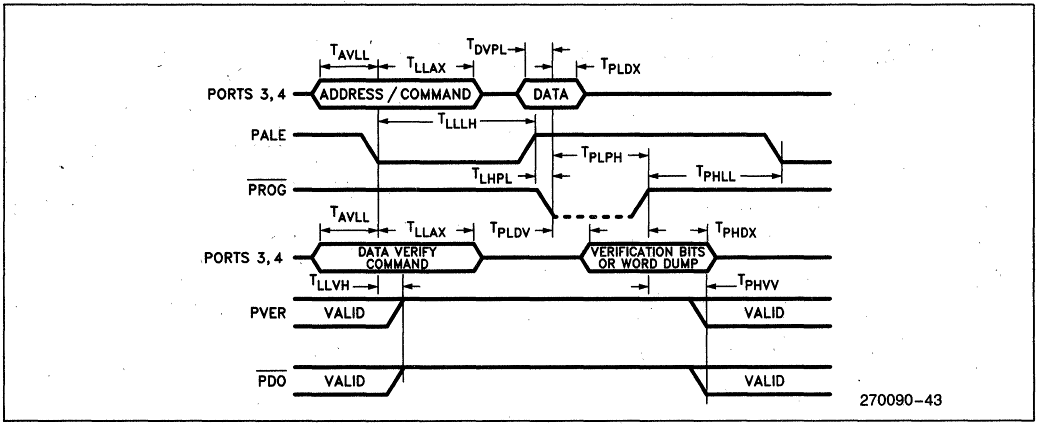
NOTE:
Run-time programming is done with F_{osc} = 6.0 MHz to 12.0 MHz, V_{CC}, V_{PD}, V_{REF} = 5V ± 0.5V, T_A = 25°C to ±5°C and V_{PP} = 12.75V ± 0.25V. For run-time programming over a full operating range, contact the factory. All windowed devices should be covered after programming.

D.C. EPROM PROGRAMMING CHARACTERISTICS

Symbol	Parameter	Min	Max	Units
I_{PP}	V_{PP} Supply Current (Whenever Programming)		100	mA
V_{PP}	Programming Supply Voltage	12.75 ±0.25		V
V_{EA}	$\overline{EA}$ Programming Voltage	11 ±2.0		V

NOTE:
V_{PP} must be within 1V of V_{CC} while V_{CC} < 4.5V. V_{PP} must not have a low impedance path to ground or V_{SS} while V_{CC} > 4.5V.

WAVEFORM—EPROM PROGRAMMING



Reserved location warning: Intel Reserved addresses can not be used by applications which use 8X9XBH internal ROM/EPROM. The data read from a reserved location is not guaranteed, and a write to any reserved location could cause unpredictable results. When attempting to program Intel Reserved addresses, the data must be 0FFFFH to ensure a harmless result. A memory map indicating reserved locations on the 8X9XBH is shown in Figure 2.

Intel Reserved locations, when mapped to external memory, must be filled with 0FFFFH to ensure compatibility with future devices.

POWER SUPPLY SEQUENCE WHILE PROGRAMMING

For any 879XBH that is in any programming mode, high voltages must be applied to the device. To avoid damaging the devices, the following rules must not be violated.

RULE #1— V_{PP} must not have a low impedance path to ground when V_{CC} is above 4.5V.

RULE #2— V_{CC} must be above 4.5V before V_{PP} can be higher than 5.0V.

RULE #3— V_{PP} must be within 1V of V_{CC} while V_{CC} is below 4.5V.

RULE #4— All voltages must be within tolerance and the oscillator stable before $\overline{RESET}$ rises.

RULE #5— $\overline{EA}$ must be brought high to place the devices in programming mode before V_{PP} is brought high.

To adhere to these rules, the following power up and power down sequences can be followed.

POWER UP

$\overline{RESET}$ = 0;
CLOCK ON; if using an external clock
; instead of an oscillator
$V_{CC} = V_{PP} = V_{EA} = 5V$;
PALE = $\overline{PROG}$ = PORT 34 = V_{IH};*
SID AND PMODE VALID;
$\overline{EA}$ = 12.75V;
V_{PP} = 12.75V;
WAIT; wait for supplies and clock to
; settle
$\overline{RESET}$ = 5V;
WAIT Tshll; See Data Sheet
BEGIN;

POWER DOWN

$\overline{RESET}$ = 0;
V_{PP} = 5V;
$\overline{EA}$ = 5V;
PALE = $\overline{PROG}$ = SID = PMODE = PORT34 = 0V;
$V_{CC} = V_{PP} = V_{EA} = 0V$;
CLOCK OFF;

NOTE:
*V_{IH} = Logical "1", 2.4V Minimum

One final note on power up, power down. The maximum limit on V_{PP} must never be violated, even for an instant. Therefore, an RC rise to the desired V_{PP} is recommended. V_{PP} is also sensitive to instantaneous voltage steps. This also can be avoided by using an RC ramp on V_{PP}.

8X9XBH ERRATA

This material is pertinent to all steppings unless otherwise noted.

1. INDEXED, 3 OPERAND MULTIPLY

The displacement portion of an indexed, three operand (byte or word) multiply may not be in the range of 200H thru 17FFH inclusive. If you must use these displacements, execute an indexed, two operand multiply and a move if necessary.

2. HIGH SPEED INPUT FIFO OPERATION

To ensure proper operation (no overflow) of the High Speed Input FIFO, it is imperative that any seven consecutive FIFO entries be completely cleared before further events occur at the pins. To clear the FIFO, some repetitive variation of the following instructions may be used...

```
READ_HSI_TASK:  DI                             ; NO INTERRUPTS
RELOAD_HREG:    JBC  IOSI, 7, RELOAD_HREG      ; WAIT FOR
                                               ; HOLDING
                                               ; REGISTER TO
                                               ;  LOAD
                STB  HSI_STAT, [PTR]+          ; READ THE
                                               ; STATUS
                ST   HSI_TIME, [PTR]+          ; READ TIME TAG
```

Allowing more than seven FIFO entries to occur between cleared conditions will result in incorrect status information for either the eighth or ninth consecutive entry. This effectively limits the total number of records to seven.

There is one exception that will allow the FIFO to correctly record eight events. If the first two events of a cleared FIFO are separated by greater than 16 state times, the overflow conditions will always cause the ninth event to be recorded with incorrect status information. This is true even if the event following the first two were only separated by eight states.

3. RESET AND THE QUASIBIDIRECTIONAL PORTS

Because RESET is asynchronous, it is possible to apply RESET during writes to the quasi-bidirectional port pins. If this occurs, the low impedance pullup may not turn on, and only the high impedance pullup is turned on. This causes the quasi-bidirectional pins to go to their RESET state (logical one) but will not do so before expiration of TRLPV max.

4. SOFTWARE RESET TIMING

The RESET pin will pull down for at least one state time if a software reset instruction executes or the watchdog timer overflows. Earlier documentation indicated two state times.

5. USING T2CLK AS THE SOURCE FOR TIMER2

TIMER2 has two selectable clock sources, the T2CLK or HSI.1 pins, selectable by bit IOC0.3. When using T2CLK as the clock for TIMER2, writing to IOC0 may cause TIMER2 to increment. The user should only write to IOC0 once during initialization, and then immediately clear TIMER2 using the HSO command OEH. Effectively, the customer cannot reset TIMER2 with bit IOC0.1, and can only use the external reset sources or the HSO command when using T2CLK as the clock source.

If the HSI.1 pin is the source for TIMER2, only the first write to IOC0 may cause TIMER2 to increment, further writes will not increment TIMER2.

DATA SHEET REVISION REVIEW

Differences between -006 and -005 datasheets.

1. All EPROM programming mode information has been deleted and moved to the 16-Bit Handbook, Hardware Design Information chapter.
2. Shrink-DIP package information has been added.
3. A new RESET timing specification has been added for clarity.
4. Software Reset pin timing information has been added.
5. HSO I_{OL} specifications have been improved so that all HSO pins have the same drive capability.
6. Port 3 and Port 4 pin descriptions were clarified, indicating the necessity of pullup if the pins are used as ports.
7. HSI FIFO overflow description added.

Differences between the -005 and the -004 version of the 8X9XBH data sheet.

1. Much of the description of device functionality has been deleted. All of this information is already in the Embedded controller handbooks in the MSC-96 8096BH Architectural Overview Chapter.
2. The A/D converter specification for Differential Non-linearity has been changed to be a minimum of > -1 lsbs to a maximum of $+2$ LSBs.
3. 8X9XBH errata Section. The JBS and JBC on Port 0 errata has been fixed on the latest device stepping.
4. 8X9XBH errata Section. The errata for the 48-pin devices has been fixed on the latest device stepping.This errata caused the upper 8 bits on the Address/Data bus to be latched when resetting into an 8-bit external memory system.
5. 8X9XBH errata Section. Errata 3 and 4 have been added to the errata list. These errata exist for all steppings of the device.

The following represents the key differences between the -004 and the -003 version of the 8X9XBH Data Sheet.

1. The bus control figures and bus timing diagrams were modified to more accurately describe their operation. In particular the 8-bit bus modes now reflect the use of Write Strobe Mode.
2. Additional text was added to the Analog/Digital description of the conversion process to clarify its operation and usefulness.
3. Text was added to the interrupt description section to indicate the maximum transition speed of the input signal relative to the CPU's state timing. A figure was included to graphically demonstrate the interrupt response timing.
4. The pin descriptions were modified to indicate that V_{PP} must normally float in the application.
5. The input low voltage specification (V_{IL1}) was deleted and is covered by the V_{IL} specification.
6. A suggested minimum configuration circuit was added to the material.
7. The A/D Converter Specifications for Differential Non-Linearity has been corrected to be a maximum of $+2$ LSB's.
8. The EPROM programming section figures were corrected to indicate the correct interface to a 2764A-2. A reset circuit was added to these figures and the signal $\overline{\text{PVAL}}$ (Port 3.X and Port 4.X) is now identified as the valid signal for program verification in the Auto Programming Mode. Text was added to this section to reference the requirement of using the Auto Configuration Byte Programming Mode for 48-lead devices. Figure 22A was edited for corrections to the text, and now indicates PVER (Port 2.0). The EPROM circuits were corrected to show 6 MHz operation for programming devices from internal microcode.
9. The protected memory section was edited to indicate that the CPU will enter a "JUMP ON SELF" condition when ROM/EPROM dump mode is complete.
10. An 8X9XBH ERRATA section was added.
11. This REVISION REVIEW was added.

PRELIMINARY

MCS®-96 809XBH/839XBH/879XBH *Express*

- **Extended Temperature Range (−40°C to +85°C)**
- **Burn-In**

The Intel EXPRESS system offers enhancements to the operational specifications of the MCS®-96 family of microcontrollers. These EXPRESS products are designed to meet the needs of those applications whose operating requirements exceed commercial standards.

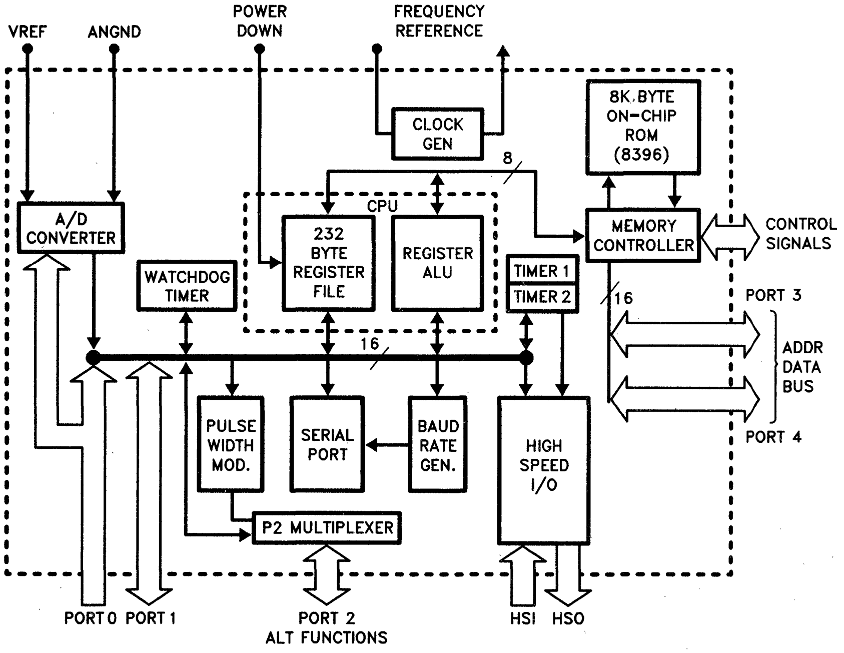
The EXPRESS program includes the commercial standard temperature range with burn-in, and an extended temperature range with or without burn-in.

With the commercial standard temperature range operational characteristics are guaranteed over the temperature range of 0°C to +70°C. With the extended temperature range option, operational characteristics are guaranteed over the range of −40°C to +85°C.

The optional burn-in is dynamic, for a minimum time of 160 hours at 125°C with V_{CC} = 5.5V ±0.5V, following guidelines in MIL-STD-883, Method 1015.

Package types and EXPRESS versions are identified by a one- or two-letter prefix to the device number. The prefixes are listed in Table 1.

This data sheet specifies the parameters for the extended temperature range option. The commercial temperature range data sheets are applicable otherwise.



270433–1

MCS®-96 Block Diagram

ELECTRICAL CHARACTERISTICS
ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias . −40°C to +85°C

Storage Temperature .... −40°C to +150°C

Voltage from V_{PP} or $\overline{\text{EA}}$
to V_{SS} or ANGND .... −0.3V to +13.0V

Voltage from Any Other Pin to
V_{SS} or ANGND .... −0.3V to +7.0V*

Average Output Current from Any Pin .... 10 mA

Power Dissipation.......................... 1.5W

*This includes V_{PP} on ROM and CPU devices.

**Notice: Stresses 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 specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

NOTICE: Specifications contained within the following tables are subject to change.

OPERATING CONDITIONS

Symbol	Parameter	Min	Max	Units
T_A	Ambient Temperature Under Bias	−40	+85	°C
V_{CC}	Digital Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	6.0	12	MHz
V_{PD}	Power-Down Supply Voltage	4.50	5.50	V

NOTE:
ANGND and V_{SS} should be nominally at the same potential.

D.C. CHARACTERISTICS (Under listed operating conditions)

Symbol	Parameter	Min	Max	Units	Test Conditions
I_{CC}	V_{CC} Supply Current ($-40°C \leq T_A \leq +85°C$)		270	mA	All Outputs Disconnected.
I_{CC1}	V_{CC} Supply Current ($T_A = +85°C$)		185	mA	
I_{PD}	V_{PD} Supply Current		1	mA	Normal operation and Power-Down.
I_{REF}	V_{REF} Supply Current		10	mA	
V_{IL}	Input Low Voltage (Except $\overline{\text{RESET}}$)	−0.3	+0.8	V	
V_{IL1}	Input Low Voltage, $\overline{\text{RESET}}$	−0.3	+0.7	V	
V_{IH}	Input High Voltage (Except $\overline{\text{RESET}}$, NMI, XTAL1)	2.0	V_{CC} +0.5	V	
V_{IH1}	Input High Voltage, $\overline{\text{RESET}}$ Rising	2.4	V_{CC} +0.5	V	
V_{IH2}	Input High Voltage, $\overline{\text{RESET}}$ Falling Hysteresis	2.1	V_{CC} +0.5	V	
V_{IH3}	Input High Voltage, NMI, XTAL1	2.3	V_{CC} +0.5	V	
I_{LI}	Input Leakage Current to each pin of HSI, P3, P4, and to P2.1.		±10	μA	V_{in} = 0 to V_{CC}
I_{LI1}	D.C. Input Leakage Current to each pin of P0		+3	μA	V_{in} = 0 to V_{CC}
I_{IH}	Input High Current to $\overline{\text{EA}}$		100	μA	V_{IH} = 2.4V
I_{IL}	Input Low Current to each pin of P1, and to P2.6, P2.7.		−150	μA	V_{IL} = 0.45V
I_{IL1}	Input Low Current to $\overline{\text{RESET}}$	−0.25	−2	mA	V_{IL} = 0.45V
I_{IL2}	Input Low Current P2.2, P2.3, P2.4, READY, BUSWIDTH		−50	μA	V_{IL} = 0.45V
V_{OL}	Output Low Voltage on Quasi-Bidirectional port pins and P3, P4 when used as ports		0.45	V	I_{OL} = 0.8 mA (Note 1)
V_{OL1}	Output Low Voltage on Quasi-Bidirectional port pins and P3, P4 when used as ports		0.75	V	I_{OL} = 2.0 mA (Notes 1, 2, 3)
V_{OL2}	Output Low Voltage on Standard Output pins, $\overline{\text{RESET}}$ and Bus/Control Pins		0.45	V	I_{OL} = 2.0 mA (Notes 1, 2, 3)

D.C. CHARACTERISTICS (Continued)

Symbol	Parameter	Min	Max	Units	Test Conditions
V_{OH}	Output High Voltage on Quasi-Bidirectional pins	2.4		V	$I_{OH} = -20\ \mu A$ (Note 1)
V_{OH1}	Output High Voltage on Standard Output pins and Bus/Control pins	2.4		V	$I_{OH} = -200\ \mu A$ (Note 1)
I_{OH3}	Output High Current on $\overline{RESET}$	−50		μA	$V_{OH} = 2.4V$
C_S	Pin Capacitance (Any Pin to V_{SS})		10	pF	fTEST = 1.0 MHz

NOTES:
1. Quasi-bidirectional pins include those on P1, for P2.6 and P2.7. Standard Output Pins include TXD, RXD (Mode 0 only), PWM, and HSO pins. Bus/Control pins include CLKOUT, ALE, $\overline{BHE}$, $\overline{RD}$, $\overline{WR}$, INST and AD0–15.
2. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V.
   I_{OL} on quasi-bidirectional pins and Ports 3 and 4 when used as ports: 4.0 mA
   I_{OL} on standard output pins and $\overline{RESET}$: 8.0 mA
   I_{OL} on Bus/Control pins: 2.0 mA
3. During normal (non-transient) operation the following limits apply:
   Total I_{OL} on Port 1 must not exceed 8.0 mA.
   Total I_{OL} on P2.0, P2.6, $\overline{RESET}$ and all HSO pins must not exceed 15 mA.
   Total I_{OL} on Port 3 must not exceed 10 mA.
   Total I_{OL} on P2.5, P2.7, and Port 4 must not exceed 20 mA.

A.C. CHARACTERISTICS (Under listed operating conditions)

Test Conditions: Load Capacitance on Output Pins = 80 pF
Oscillator Frequency = 10 MHz

TIMING REQUIREMENTS (Other system components must meet these specs.)

Symbol	Parameter	Min	Max	Units
T_{CLYX}(2, 3)	READY Hold after CLKOUT Edge	0(1)		ns
T_{LLYV}	End of ALE/$\overline{ADV}$ to READY Valid		2Tosc−70	ns
T_{LLYH}	End of ALE/$\overline{ADV}$ to READY High	2Tosc+40	4Tosc−80	ns
T_{YLYH}	Non-Ready Time		1000	ns
T_{AVDV}(4)	Address Valid to Input Data Valid		5Tosc−120	ns
T_{RLDV}	$\overline{RD}$ Active to Input Data Valid		3Tosc−100	ns
T_{RHDX}	Data Hold after $\overline{RD}$ Inactive	0		ns
T_{RHDZ}	$\overline{RD}$ Inactive to Input Data Float	0	Tosc−25	ns
T_{AVGV}(2, 4)	Address Valid to BUSWIDTH Valid		2 Tosc −125	ns
T_{LLGX}(2, 3)	BUSWIDTH Hold after ALE/$\overline{ADV}$ Low	Tosc +40		ns
T_{LLGV}(2, 3)	ALE/$\overline{ADV}$ Low to BUSWIDTH Valid		Tosc −75	ns
T_{RLPV}	Reset Low to Ports Valid		10 Tosc	ns

NOTES:
1. If the 48-pin or 64-pin device is being used then this timing can be generated by assuming that the CLKOUT falling edge has occurred at 2Tosc+55 (TLLCH(max) + TCHCL(max)) after the falling edge of ALE.
2. Pins not bonded out on 64-pin device.
3. Pins not bonded out on 48-pin device.
4. The term "Address Valid" applies to AD0–15, $\overline{BHE}$ and INST.

A.C. CHARACTERISTICS (Continued)

TIMING RESPONSES (MCS-96 devices meet these specs.)

Symbol	Parameter	Min	Max	Units
F_{XTAL}	Oscillator Frequency	6.0	12.0	MHz
T_{OSC}	Oscillator Period	83	166	ns
T_{OHCH}	XTAL1 Rising Edge to Clockout Rising Edge	0(4)	120(4)	ns
T_{CHCH}(1, 4)	CLKOUT Period(3)	3Tosc(3)	3Tosc(3)	ns
T_{CHCL}(1, 4)	CLKOUT High Time	Tosc − 35	Tosc + 10	ns
T_{CLLH}(1, 4)	CLKOUT Low to ALE High	−20	+25	ns
T_{LLCH}(1, 4)	ALE/$\overline{\text{ADV}}$ Low to CLKOUT High	Tosc − 25	Tosc + 45	ns
T_{LHLL}	ALE/$\overline{\text{ADV}}$ High Time	Tosc − 30	Tosc + 35(5)	ns
T_{AVLL}(6)	Address Setup to End of ALE/$\overline{\text{ADV}}$	Tosc − 50		ns
T_{RLAZ}(7)	$\overline{\text{RD}}$ or $\overline{\text{WR}}$ Low to Address Float		25	ns
T_{LLRL}	End of ALE/$\overline{\text{ADV}}$ to $\overline{\text{RD}}$ or $\overline{\text{WR}}$ Active	Tosc − 40		ns
T_{LLAX}(7)	Address Hold after End of ALE/$\overline{\text{ADV}}$	Tosc − 40		ns
T_{WLWH}	$\overline{\text{WR}}$ Pulse Width	3Tosc − 35		ns
T_{QVWH}	Output Data Valid to End of $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$	3Tosc − 60		ns
T_{WHQX}	Output Data Hold after $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$	Tosc − 50		ns
T_{WHLH}	End of $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$ to ALE/$\overline{\text{ADV}}$ High	Tosc − 75		ns
T_{RLRH}	$\overline{\text{RD}}$ Pulse Width	3Tosc − 30		ns
T_{RHLH}	End of $\overline{\text{RD}}$ to ALE/$\overline{\text{ADV}}$ High	Tosc − 45		ns
T_{CLLL}(1, 4)	CLOCKOUT Low to ALE/$\overline{\text{ADV}}$ Low	Tosc − 40	Tosc + 35	ns
T_{RHBX}(1, 4)	$\overline{\text{RD}}$ High to INST, $\overline{\text{BHE}}$, AD8-15 Inactive	Tosc − 25	Tosc + 30	ns
T_{WHBX}(1, 4)	$\overline{\text{WR}}$ High to INST, $\overline{\text{BHE}}$, AD8-15 Inactive	Tosc − 50	Tosc + 100	ns
T_{HLHH}	$\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ High	2Tosc − 35	2Tosc + 40	ns
T_{LLHL}	ALE/$\overline{\text{ADV}}$ Low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low	2Tosc − 30	2Tosc + 55	ns
T_{QVHL}	Output Data Valid to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low	Tosc − 60		ns

NOTES:

1. Pins not bonded out on 64-pin devices.
2. If more than one wait state is desired, add 3Tosc for each additional wait state.
3. CLKOUT is directly generated as a divide by 3 of the oscillator. The period will be 3Tosc ± 10 ns if Tosc is constant and the rise and fall times on XTAL1 are less than 10 ns.
4. Pins not bonded out on 48-pin devices.
5. Max spec applies only to ALE. Min spec applies to both ALE and $\overline{\text{ADV}}$.
6. The term "Address Valid" applies to AD0-15, $\overline{\text{BHE}}$ and INST.
7. The term " Address" in this definition applies to AD0-7 for 8-bit cycles, and AD0-15 for 16-bit cycles.

Table 1. MCS®-96 Packaging—8X9XBH

	Factory Masked ROM			CPU			User Programmable					
							EPROM			OTP		
	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin
ANALOG	8397BH	8397BH	8395BH	8097BH	8097BH	8095BH	8797BH		8795BH	8797BH	8797BH	
NO ANALOG	8396BH			8096BH								

48-pin devices have four Analog input pins.

64-pin devices have all 48-pin device features plus the following:

- Four additional Analog input channels
- One additional quasi-bidirectional 8-bit parallel port
- Four additional port 2 pins with multiplexed features
  - Timer 2 clock source pin
  - Timer 2 reset pin
  - Two additional quasi-bidirectional port pins

68-pin devices have all 48- and 64-pin features plus the following:

- Dynamic buswidth sizing (8- or 16-bit bus)
- Dedicated System Clock Output (CLKOUT)
- INST pin for memory expansion
- Non-Maskable interrupt for debugging

Package Designators:

- N = PLCC
- C = Ceramic DIP
- A = Ceramic Pin Grid Array
- P = Plastic DIP
- R = Ceramic LCC
- U = Shrink DIP

DATA SHEET REVISION REVIEW

This represents the differences between this datasheet and the -001 version.

T_{RLPV}—Reset low to ports valid specification added for clarity.

PRODUCT PREVIEW

MCS®-96
809XJF, 839XJF, 879XJF
ADVANCED 16-BIT MICROCONTROLLER WITH 8- OR 16-BIT EXTERNAL BUS

- **879XJF: an 809XJF with 16K Bytes of On-Chip EPROM**
- **839XJF: an 809XJF with 16K Bytes of On-Chip ROM**

- **232 Byte Register File**
- **256 Bytes XRAM for Code**
- **10-Bit A/D Converter with S/H**
- **Five 8-Bit I/O Ports**
- **20 Interrupt Sources**
- **Pulse-Width Modulated Output**
- **ROM/EPROM Lock**
- **Run-Time Programmable EPROM (OTP)**
- **High Speed I/O Subsystem**
- **Full Duplex Serial Port**
- **Dedicated Baud Rate Generator**
- **6.25 μs 16 x 16 Multiply**
- **6.25 μs 32/16 Divide**
- **16-Bit Watchdog Timer**
- **Four 16-Bit Software Timers**
- **Two 16-Bit Counter/Timers**

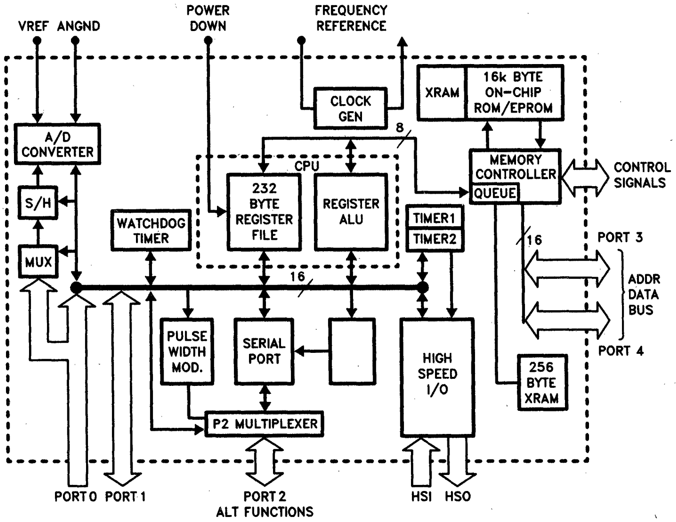
The MCS-96 family of 16-bit microcontrollers consists of many members, all of which are designed for high-speed control functions. The MCS-96 family members produced using Intel's HMOS-III process are described in this data sheet.

The CPU supports bit, byte, and word operations. Thirty-two bit double-words are supported for a subset of the instruction set. With a 12 MHz input frequency the 8097JF can do a 16-bit addition in 1.0 μs and a 16 x 16-bit multiply or 32/16 divide in 6.25 μs. Instruction execution times average 1 to 2 μs in typical applications.

Four high-speed trigger inputs are provided to record the times at which external events occur. Six high-speed pulse generator outputs are provided to trigger external events at preset times. The high-speed output unit can simultaneously perform software timer functions. Up to four 16-bit software timers can be in operation at once.

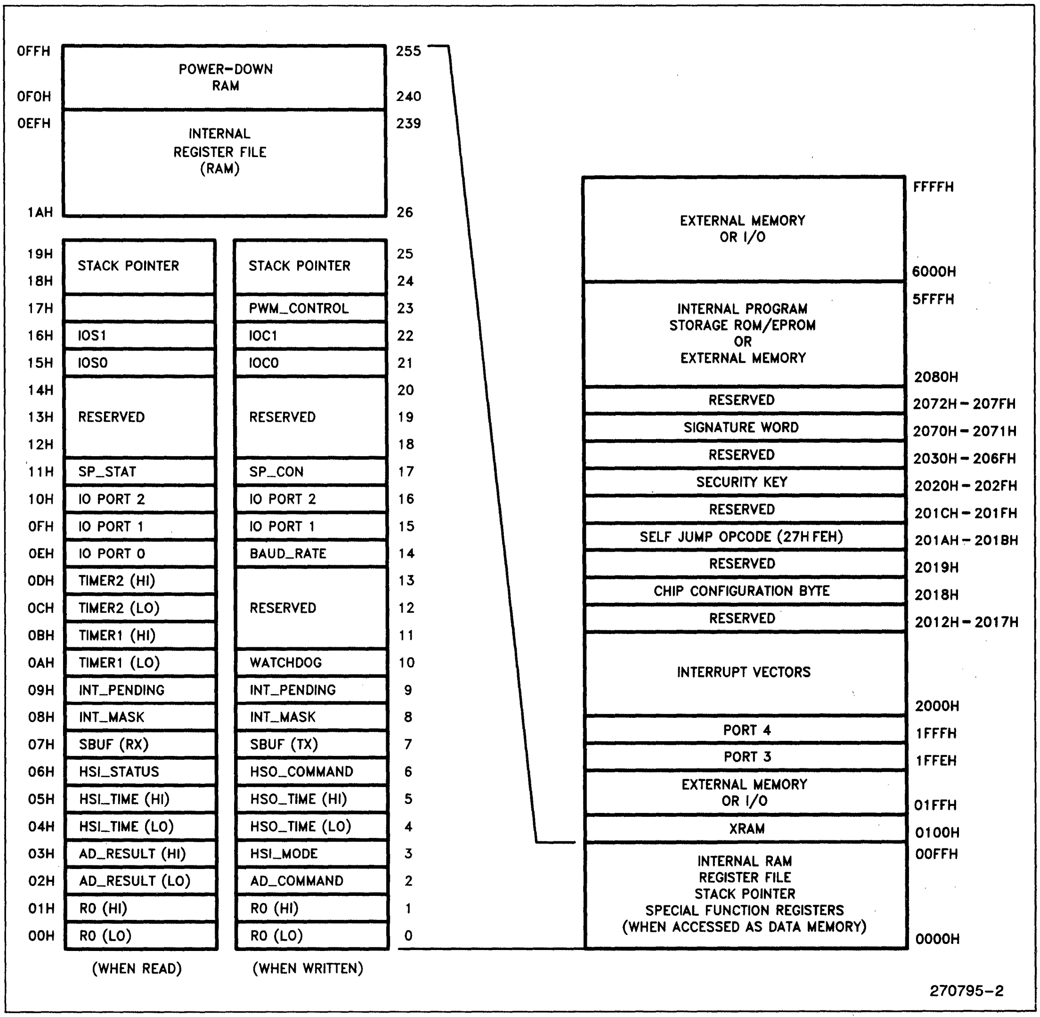
The on-chip A/D converter includes a Sample and Hold, and converts up to 8 multiplexed analog input channels to 10-bit digital values. With a 12 MHz crystal, each conversion takes 22 μs.

Also provided on-chip are a serial port, a Watchdog Timer, and a pulse-width modulated output signal.



270795–1

Figure 1. MCS®-96 Block Diagram



	(WHEN READ)	(WHEN WRITTEN)	
0FFH 0F0H	POWER-DOWN RAM		255 240
0EFH 1AH	INTERNAL REGISTER FILE (RAM)		239 26
19H 18H	STACK POINTER	STACK POINTER	25 24
17H		PWM_CONTROL	23
16H	IOS1	IOC1	22
15H	IOS0	IOC0	21
14H 13H 12H	RESERVED	RESERVED	20 19 18
11H	SP_STAT	SP_CON	17
10H	IO PORT 2	IO PORT 2	16
0FH	IO PORT 1	IO PORT 1	15
0EH	IO PORT 0	BAUD_RATE	14
0DH	TIMER2 (HI)	RESERVED	13
0CH	TIMER2 (LO)		12
0BH	TIMER1 (HI)		11
0AH	TIMER1 (LO)	WATCHDOG	10
09H	INT_PENDING	INT_PENDING	9
08H	INT_MASK	INT_MASK	8
07H	SBUF (RX)	SBUF (TX)	7
06H	HSI_STATUS	HSO_COMMAND	6
05H	HSI_TIME (HI)	HSO_TIME (HI)	5
04H	HSI_TIME (LO)	HSO_TIME (LO)	4
03H	AD_RESULT (HI)	HSI_MODE	3
02H	AD_RESULT (LO)	AD_COMMAND	2
01H	R0 (HI)	R0 (HI)	1
00H	R0 (LO)	R0 (LO)	0

EXTERNAL MEMORY OR I/O	FFFFH 6000H
INTERNAL PROGRAM STORAGE ROM/EPROM OR EXTERNAL MEMORY	5FFFH 2080H
RESERVED	2072H – 207FH
SIGNATURE WORD	2070H – 2071H
RESERVED	2030H – 206FH
SECURITY KEY	2020H – 202FH
RESERVED	201CH – 201FH
SELF JUMP OPCODE (27H FEH)	201AH – 201BH
RESERVED	2019H
CHIP CONFIGURATION BYTE	2018H
RESERVED	2012H – 2017H
INTERRUPT VECTORS	2000H
PORT 4	1FFFH
PORT 3	1FFEH
EXTERNAL MEMORY OR I/O	01FFH
XRAM	0100H
INTERNAL RAM REGISTER FILE STACK POINTER SPECIAL FUNCTION REGISTERS (WHEN ACCESSED AS DATA MEMORY)	00FFH 0000H

270795–2

Figure 2. Memory Map

PACKAGING

The 8097JF is available in 64-pin and 68-pin packages, with A/D, and with and without on-chip ROM or EPROM. The 8097JF numbering system is shown in Figure 3. Figures 4–6 show the pinouts for the 64- and 68-pin packages. The 64-pin version is offered in a Shrink-DIP package while the 68-pin versions come in a Plastic Leaded Chip Carrier (PLCC).

MCS96 PACKAGING 8X9XJF

	Factory Masked ROM			CPU			User Programmable					
							EPROM			OTP		
	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin	68 Pin	64 Pin	48 Pin
Analog	8397JF	8397JF		8097JF	8097JF					8797JF	8797JF	
No Analog												

Package Designators:
N = PLCC
C = Ceramic DIP
A = Ceramic Pin Grid Array
P = Plastic DIP
R = Ceramic LCC
U = Shrink DIP

Figure 3. The 809XJF Family Nomenclature

PLCC	Description	PLCC	Description	PLCC	Description
9	ACH7/P0.7/PMOD.3	54	AD6/P3.6	31	P1.6
8	ACH6/P0.6/PMOD.2	53	AD7/P3.7	30	P1.5
7	ACH2/P0.2	52	AD8/P4.0	29	HSO.1
6	ACH0/P0.0	51	AD9/P4.1	28	HSO.0
5	ACH1/P0.1	50	AD10/P4.2	27	HSO.5/HSI.3
4	ACH3/P0.3	49	AD11/P4.3	26	HSO.4/HSI.2
3	NMI	48	AD12/P4.4	25	HSI.1
2	$\overline{\text{EA}}$	47	AD13/P4.5	24	HSI.0
1	VCC	46	AD14/P4.6	23	P1.4
68	VSS	45	AD15/P4.7	22	P1.3
67	XTAL1	44	T2CLK/P2.3V	21	P1.2
66	XTAL2	43	READY	20	P1.1
65	CLKOUT	42	T2RST/P2.4	19	P1.0
64	BUSWIDTH	41	$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	18	TXD/P2.0/PVER/SALE
63	INST	40	$\overline{\text{WR}}$/$\overline{\text{WRL}}$	17	RXD/P2.1/PALE
62	ALE/$\overline{\text{ADV}}$	39	PWM/P2.5/$\overline{\text{PDO}}$/$\overline{\text{SPROG}}$	16	$\overline{\text{RESET}}$
61	$\overline{\text{RD}}$	38	P2.7	15	EXTINT/P2.2/$\overline{\text{PROG}}$
60	AD0/P3.0	37	VPP	14	VPD
59	AD1/P3.1	36	VSS	13	VREF
58	AD2/P3.2	35	HSO.3	12	ANGND
57	AD3/P3.3	34	HSO.2	11	ACH4/P0.4/PMOD.0
56	AD4/P3.4	33	P2.6	10	ACH5/P0.5/PMOD.1
55	AD5/P3.5	32	P1.7		

Figure 4. PLCC Function Pinout

SDIP	Description	SDIP	Description	SDIP	Description
1	$\overline{\text{EA}}$	23	HSI.1	45	AD13/P4.5
2	ACH3/P0.3	24	HSO.4/HSI.2	46	AD12/P4.4
3	ACH1/P0.1	25	HSO.5/HSI.3	47	AD11/P4.3
4	ACH0/P0.0	26	HSO.0	48	AD10/P4.2
5	ACH2/P0.2	27	HSO.1	49	AD9/P4.1
6	ACH6/P0.6/PMOD.2	28	P1.5	50	AD8/P4.0
7	ACH7/P0.7/PMOD.3	29	P1.6	51	AD7/P3.7
8	ACH5/P0.5/PMOD.1	30	P1.7	52	AD6/P3.6
9	ACH4/P0.4/PMOD.0	31	P2.6	53	AD5/P3.5
10	ANGND	32	HSO.2	54	AD4/P3.4
11	V_{REF}	33	HSO.3	55	AD3/P3.3
12	V_{PD}	34	V_{SS}	56	AD2/P3.2
13	EXINT/P2.2/$\overline{\text{PROG}}$	35	V_{PP}	57	AD1/P3.1
14	$\overline{\text{RESET}}$	36	P2.7	58	AD0/P3.0
15	RXD/P2.1/PALE	37	PWM/P2.5/$\overline{\text{PD0}}$/$\overline{\text{SPROG}}$	59	$\overline{\text{RD}}$
16	TXD/P2.0/PVER/SALE	38	$\overline{\text{WR}}$/$\overline{\text{WRL}}$	60	ALE/$\overline{\text{ADV}}$
17	P1.0	39	$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	61	XTAL2
18	P1.1	40	T2RST/P2.4	62	XTAL1
19	P1.2	41	READY	63	V_{SS}
20	P1.3	42	T2CLK/P2.3	64	V_{CC}
21	P1.4	43	AD15/P4.7		
22	HSI.0	44	AD14/P4.6		

Figure 5. SDIP 8X97JF Pin Out

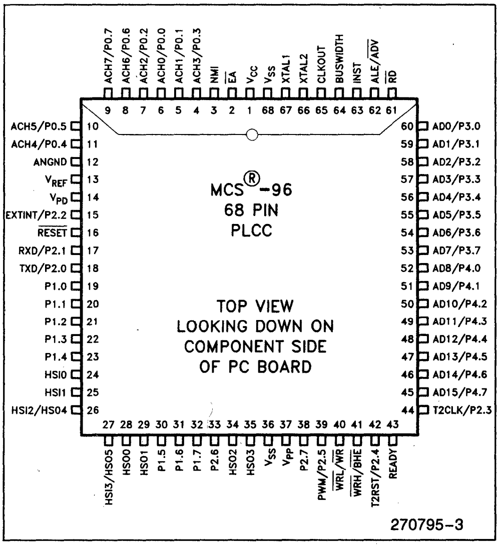


Figure 6. 68-Pin Package (PLCC - Top View)

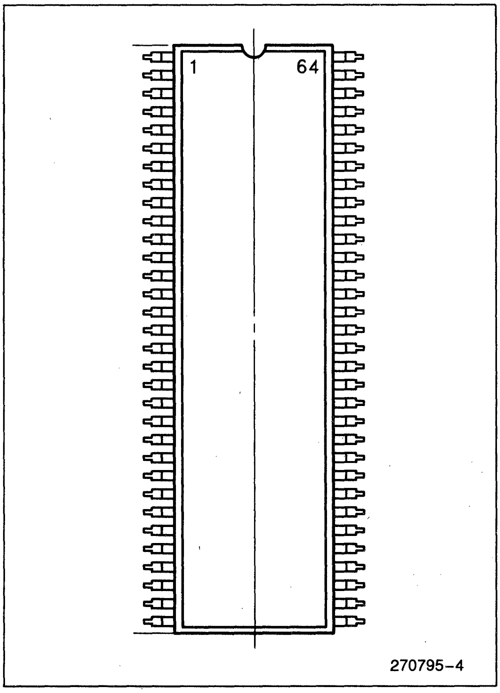


Figure 7. Shrink-DIP Package

PIN DESCRIPTIONS

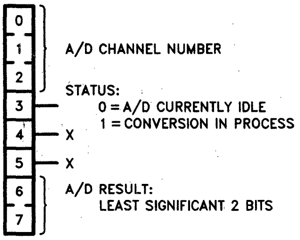
Symbol	Name and Function
V_{CC}	Main supply voltage (5V).
V_{SS}	Digital circuit ground (0V). There are two V_{SS} pins, both of which must be connected.
V_{PD}	RAM standby supply voltage (5V). This voltage must be present during normal operation. In a Power Down condition (i.e. V_{CC} drops to zero), if $\overline{\text{RESET}}$ is activated before V_{CC} drops below spec and V_{PD} continues to be held within spec., the top 16 bytes in the Register File will retain their contents. $\overline{\text{RESET}}$ must be held low during the Power Down and should not be brought high until V_{CC} is within spec and the oscillator has stabilized.
V_{REF}	Reference voltage for the A/D converter (5V). V_{REF} is also the supply voltage to the analog portion of the A/D converter and the logic used to read Port 0.
ANGND	Reference ground for the A/D converter. Must be held at nominally the same potential as V_{SS}.
V_{PP}	Programming voltage for the EPROM devices. It should be +12.75V for programming. This pin must be left floating in the application circuit.
XTAL1	Input of the oscillator inverter and of the internal clock generator.
XTAL2	Output of the oscillator inverter.
CLKOUT*	Output of the internal clock generator. The frequency of CLKOUT is ⅓ the oscillator frequency. It has a 33% duty cycle.
$\overline{\text{RESET}}$	Reset input to the chip. Input low for a minimum of 10 XTAL1 cycles to reset the chip. The subsequent low-to-high transition re-synchronizes CLKOUT and commences a 10-state-time sequence in which the PSW is cleared, a byte read from 2018H loads CCR, and a jump to location 2080H is executed. Input high for normal operation. $\overline{\text{RESET}}$ has an internal pullup.
BUSWIDTH*	Input for bus width selection. If CCR bit 1 is a one, this pin selects the bus width for the bus cycle in progress. If BUSWIDTH is a 1, a 16-bit bus cycle occurs. If BUSWIDTH is a 0 an 8-bit cycle occurs. If CCR bit 1 is a 0, the bus is always an 8-bit bus. If this pin is left unconnected, it will rise to V_{CC}.
NMI*	A positive transition causes a vector to external memory location 0000H. External memory from 00H through 0FFH is reserved for Intel development systems.
INST*	Output high during an external memory read indicates the read is an instruction fetch. INST is valid throughout the bus cycle.
$\overline{\text{EA}}$	Input for memory select (External Access). $\overline{\text{EA}}$ equal to a TTL-high causes memory accesses to locations 2000H through 5FFF to be directed to on-chip ROM/EPROM. $\overline{\text{EA}}$ equal to a TTL-low causes accesses to these locations to be directed to off-chip memory. $\overline{\text{EA}}$ = +12.5V causes execution to begin in the Programming Mode. $\overline{\text{EA}}$ has an internal pulldown, so it goes to 0 unless driven otherwise.
ALE/$\overline{\text{ADV}}$	Address Latch Enable or Address Valid output, as selected by CCR. Both pin options provide a latch to demultiplex the address from the address/data bus. When the pin is $\overline{\text{ADV}}$, it goes inactive high at the end of the bus cycle. $\overline{\text{ADV}}$ can be used as a chip select for a single external RAM memory. ALE/$\overline{\text{ADV}}$ is activated only during external memory accesses.

*Not available on Shrink-DIP Package

PIN DESCRIPTIONS (Continued)

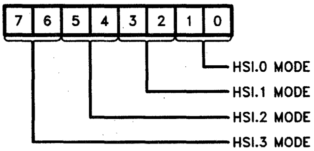
Symbol	Name and Function
$\overline{RD}$	Read signal output to external memory. $\overline{RD}$ is activated only during external memory reads.
$\overline{WR}/\overline{WRL}$	Write and Write Low output to external memory, as selected by the CCR. $\overline{WR}$ will go low for every external write, while $\overline{WRL}$ will go low only for external writes where an even byte is being written. $\overline{WR}/\overline{WRL}$ is activated only during external memory writes.
$\overline{BHE}/\overline{WRH}$	Bus High Enable or Write High output to external memory, as selected by the CCR. $\overline{BHE}$ = 0 selects the bank of memory that is connected to the high byte of the data bus. A0 = 0 selects the bank of memory that is connected to the low byte of the data bus. Thus accesses to a 16-bit wide memory can be to the low byte only (A0 = 0, $\overline{BHE}$ = 1), to the high byte only (A0 = 1, $\overline{BHE}$ = 0), or both bytes (A0 = 0, $\overline{BHE}$ = 0). If the $\overline{WRH}$ function is selected, the pin will go low if the bus cycle is writing to an odd memory location.
READY	Ready input to lengthen external memory cycles, for interfacing to slow or dynamic memory, or for bus sharing. If the pin is high, CPU operation continues in a normal manner. If the pin is low prior to the falling edge of CLKOUT, the memory controller goes into a wait mode until the next positive transition in CLKOUT occurs with READY high. The bus cycle can be lengthened by up to 1 μs. When the external memory is not being used, READY has no effect. Internal control of the number of wait states inserted into a bus cycle held not ready is available through configuration of CCR. READY has a weak internal pullup, so it goes to 1 unless externally pulled low.
HSI	Inputs to High Speed Input Unit. Four HSI pins are available: HSI.0, HSI.1, HSI.2, and HSI.3. Two of them (HSI.2 and HSI.3) are shared with the HSO Unit. The HSI pins are also used as inputs by EPROM devices in Programming Mode.
HSO	Outputs from High Speed Output Unit. Six HSO pins are available: HSO.0, HSO.1, HSO.2, HSO.3, HSO.4, and HSO.5. Two of them (HSO.4 and HSO.5) are shared with the HSI Unit.
Port 0	8-bit high impedance input-only port. These pins can be used as digital inputs and/or as analog inputs to the on-chip A/D converter. These pins are also a mode input to EPROM devices in the Programming Mode.
Port 1	8-bit quasi-bidirectional I/O port.
Port 2	8-bit multi-functional port. Six of its pins are shared with other functions in the 8096JF, the remaining 2 are quasi-bidirectional. These pins are also used to input and output control signals on EPROM devices in Programming Mode.
Ports 3 and 4	8-bit bi-directional I/O ports with open drain outputs. These pins are shared with the multiplexed address/data bus which has strong internal pullups. Ports 3 and 4 are also used as a command, address and data path by EPROM devices operating in the Programming Mode. When used as ports, pullups to V_{CC} may be needed.

A/D Result LO (02H)



270795–5

HSI_Mode (03H)

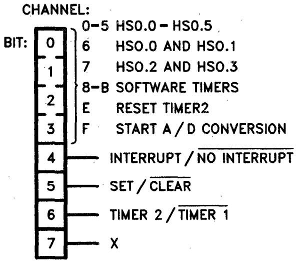


WHERE EACH 2-BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

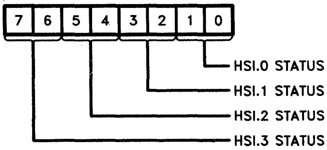
270795–6

HSO Command (06H)



270795–7

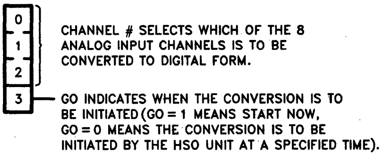
HSI_Status (06H)



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT INDICATES THE CURRENT STATUS OF THE PIN.

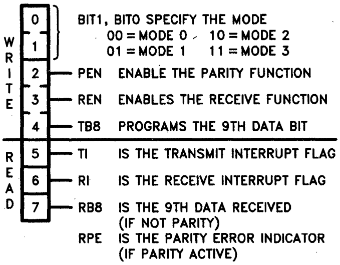
270795–8

A/D Command (02H)


0
1
2
CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.
3
GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).

270795–9

SPCON/SPSTAT (11H)


WRITE
0
1
BIT1, BIT0 SPECIFY THE MODE
00 = MODE 0 10 = MODE 2
01 = MODE 1 11 = MODE 3
2
PEN ENABLE THE PARITY FUNCTION
3
REN ENABLES THE RECEIVE FUNCTION
4
TB8 PROGRAMS THE 9TH DATA BIT
READ
5
TI IS THE TRANSMIT INTERRUPT FLAG
6
RI IS THE RECEIVE INTERRUPT FLAG
7
RB8 IS THE 9TH DATA RECEIVED (IF NOT PARITY)
RPE IS THE PARITY ERROR INDICATOR (IF PARITY ACTIVE)

270795–10

Baud Rate Calculations

Using XTAL1:

Mode 0: $\text{Baud Rate} = \dfrac{\text{XTAL1 frequency}}{4^*(B+1)}; B \neq 0$

Others: $\text{Baud Rate} = \dfrac{\text{XTAL1 frequency}}{64^*(B+1)}$

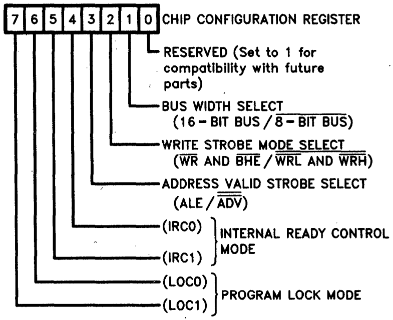
Using T2CLK:

Mode 0: $\text{Baud Rate} = \dfrac{\text{T2CLK frequency}}{B}; B \neq 0$

Others: $\text{Baud Rate} = \dfrac{\text{T2CLK frequency}}{16^*B}; B \neq 0$

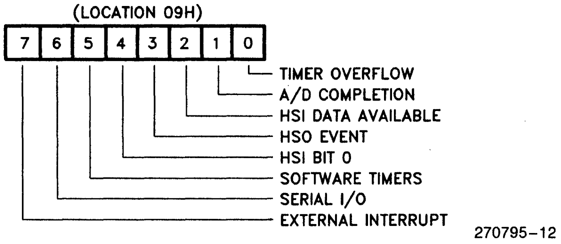
Note that B cannot equal 0, except when using XTAL1 in other than Mode 0.

Chip Configuration


7 6 5 4 3 2 1 0
CHIP CONFIGURATION REGISTER
RESERVED (Set to 1 for compatibility with future parts)
BUS WIDTH SELECT (16-BIT BUS / $\overline{\text{8-BIT BUS}}$)
WRITE STROBE MODE SELECT ($\overline{\text{WR}}$ AND $\overline{\text{BHE}}$ / $\overline{\text{WRL}}$ AND $\overline{\text{WRH}}$)
ADDRESS VALID STROBE SELECT (ALE / $\overline{\text{ADV}}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE

270795–11

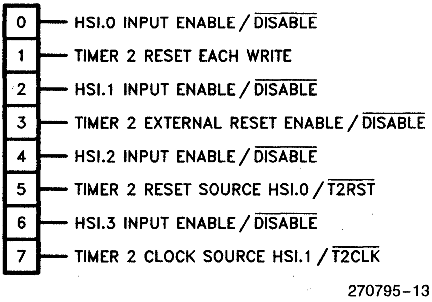
Interrupt Pending Register



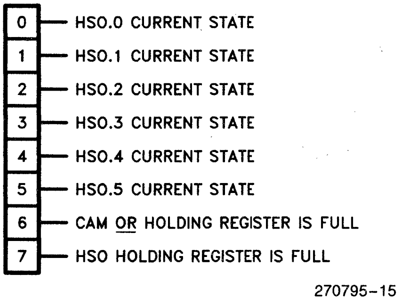
PSW Register

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Z	N	V	VT	C	—	I	ST	<Interrupt Mask Reg>							

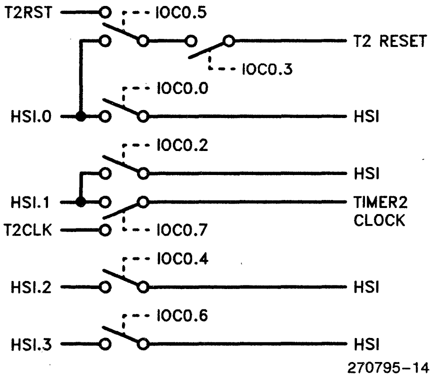
IOC0 (15H)


0 HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1 TIMER 2 RESET EACH WRITE
2 HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3 TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4 HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5 TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6 HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7 TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$
270795–13

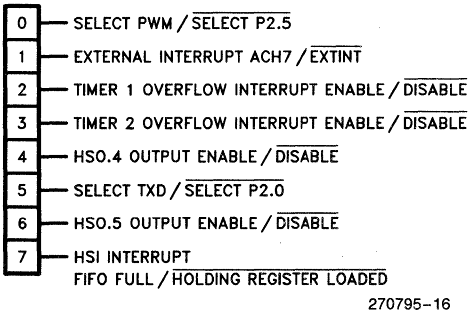
IOS0 (15H)



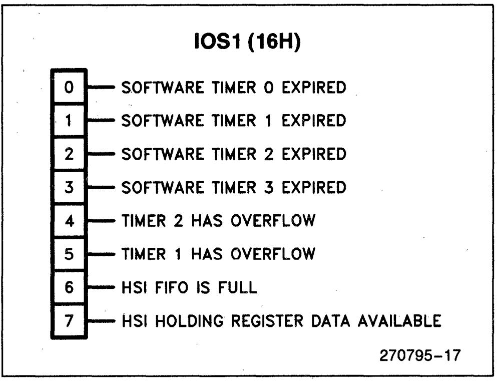
IOC0 (15H)



IOC1 (16H)


0 SELECT PWM / $\overline{\text{SELECT P2.5}}$
1 EXTERNAL INTERRUPT ACH7 / $\overline{\text{EXTINT}}$
2 TIMER 1 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
3 TIMER 2 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
4 HSO.4 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
5 SELECT TXD / $\overline{\text{SELECT P2.0}}$
6 HSO.5 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
7 HSI INTERRUPT FIFO FULL / $\overline{\text{HOLDING REGISTER LOADED}}$
270795–16

Vector	Vector Location		Priority
	(High Byte)	(Low Byte)	
Software Trap	2011H	2010H	Not Applicable
Extint	200FH	200EH	7 (Highest)
Serial Port	200DH	200CH	6
Software Timers	200BH	200AH	5
HSI.0	2009H	2008H	4
High Speed Outputs	2007H	2006H	3
HSI Data Available	2005H	2004H	2
A/D Conversion Complete	2003H	2002H	1
Timer Overflow	2001H	2000H	0 (Lowest)


IOS1 (16H)
0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE
270795-17

ELECTRICAL CHARACTERISTICS
ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias .... 0°C to +70°C

Storage Temperature .... -40°C to +150°C

Voltage from EA or V_{PP}
to V_{SS} or ANGND .... -0.3V to +13.0V

Voltage from Any Other Pin to
V_{SS} or ANGND .... -0.3V to +7.0V*

Average Output Current from Any Pin .... 10 mA

Power Dissipation 1.5W

•This includes V_{PP} on ROM and CPU only devices.

NOTICE: This document contains Information on products in the design phase of development. Contact the Intel Sales Office for current specifications.

** Notice: Stresses 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 operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

OPERATING CONDITIONS

Symbol	Parameter	Min	Max	Units
T_A	Ambient Temperture Under Bias	0	+70	°C
V_{CC}	Digitial Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	6.0	12	MHz
V_{PD}	Power-Down Supply Voltage	4.50	5.50	V

Note: ANGND and V_{SS} should be nominally at the same potential.

TARGETED D.C. CHARACTERISTICS (Test Conditions: V_{CC}, V_{REF}, V_{PD}, V_{PP}, V_{EA} = 5.0V ± 0.5V; F_{OSC} = 6.0 MHz; T_A = 0°C TO 70°C; V_{SS}, ANGND - 0V)

Symbol	Parameter	Min	Max	Units	Test Conditions
I_{CC}	V_{CC} Supply Current (°0 ≤ T_A ≤ 70°C)		300	mA	All Outputs Disconnected.
I_{CC1}	V_{CC} Supply Current (T_A = 70°C)		245	mA	
I_{PD}	V_{PD} Supply Current		1	mA	Normal operation and Power-Down.
I_{REF}	V_{REF} Supply Current		8	mA	
V_{IL}	Input Low Voltage	–0.3	+0.8	V	
V_{IH}	Input High Voltage (Except $\overline{\text{RESET}}$, NMI, XTAL1)	2.0	V_{CC} +0.5	V	
V_{IH1}	Input High Voltage, $\overline{\text{RESET}}$ Rising	2.4	V_{CC} +0.5	V	
V_{IH2}	Input High Voltage, $\overline{\text{RESET}}$ Falling Hysteresis	2.1	V_{CC} +0.5	V	
V_{IH3}	Input High Voltage, NMI(4), XTAL1	2.2	V_{CC} +0.5	V	
I_{LI}	Input Leakage Current to each pin of HSI, P3, P4 and to P2.1.		±10	µA	V_{IN} = 0 to V_{CC}
I_{LI1}	D.C. Input Leakage Current to each pin of P0		±3	µA	V_{IN} = 0 to V_{CC}
I_{IH}	Input High Current to $\overline{\text{EA}}$		100	µA	V_{IH} = 2.4V
I_{IL}	Input Low Current to each pin of P1, and to P2.6, P2.7.		–125	µA	V_{IL} = 0.45V
I_{IL1}	Input Low Current to $\overline{\text{RESET}}$	–0.25	–2	mA	V_{IL} = 0.45V
I_{IL2}	Input Low Current P2.2, P2.3, P2.4, READY, BUSWIDTH(4)		–50	µA	V_{IL} = .045V
V_{OL}	Output Low Voltage on Quasi-Bidirectional port pins and P3, P4 when used as ports		0.45	V	I_{OL} = 0.8 mA (Note 1)
V_{OL}	Output Low Voltage on Quasi-Bidirectional port pins and P3, P4 when used as ports		0.75	V	I_{OL} = 2.0 mA (Note 1, 2, 3)
V_{OL2}	Output Low Voltage on Standard Output pins, $\overline{\text{RESET}}$ and Bus/Control Pins		0.45	V	I_{OL} = 2.0 mA (Notes 1, 2, 3)

TARGETED D.C. CHARACTERISTICS (Test Conditions: V_{CC}, V_{REF}, V_{PD}, V_{PP}, V_{EA} = 5.0V ± 0.5V; F_{OSC} = 6.0 MHz; T_A = 0°C TO 70°C; V_{SS}, ANGND - 0V) (Continued)

Symbol	Parameter	Min	Max	Units	Test Conditions
V_{OH}	Output High Voltage on Quasi-Bidirectional pins	2.4		V	I_{OH} = –20 µA (Note 1)
V_{OH1}	Output High Voltage on Standard Output pins and Bus/Control pins	2.4		V	I_{OH} = –200 µA (Note 1)
I_{OH3}	Output High Current on $\overline{RESET}$	–50		µA	V_{OH} = 2.4V
C_S	Pin Capacitance (Any Pin to V_{SS})		10	pF	fTEST = 1.0 MHz

Notes:

1. Quasi-Bidirectional pins include those on P1, for P2.6 and P2.7. Standard Output Pins include TXD, RXD (Mode 0 only), PWM, and HSO pins. Bus/Control pins include CLKOUT, ALE, $\overline{BHE}$, $\overline{RD}$, $\overline{WR}$, INST and AD0–15.
2. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V.
   I_{OL} on quasi-bidirectional pins and Ports 3 and 4 when used as ports: 4.0 mA
   I_{OL} on standard output pins and $\overline{RESET}$: 8.0 mA
   I_{OL} on Bus/Control pins: 2.0 mA
3. During normal (non-transient) operation the following limits apply:
   Total I_{OL} on Port 1 must not exceed 8.0 mA.
   Total I_{OL} on P2.0. P2.6, $\overline{RESET}$ and all HSO pins must not exceed 15 mA.
   Total I_{OL} on Port 3 must not exceed 10 mA.
   Total I_{OL} on P2.5. P2.7, and Port 4 must not exceed 20 mA.

TARGETED A.C. CHARACTERISTICS V_{CC}, V_{PD} = 4.5 to 5.5V; T_A = 0°C TO 70°C; F_{OSC} = 6.0 to 12.0 MHz

Test Conditions: Load Capacitance on Output Pins = 80pF
Oscillator Frequency = 10 MHz

Symbol	Parameter	Min	Max	Units
T_{CLYX}(3)	READY Hold after CLKOUT Edge	0(1)		ns
T_{LLYV}	End of ALE/ADV to READY Valid		$2T_{OSC}$ –70	ns
T_{LLYH}	End of ALE/ADV to READY high	$2T_{OSC}$ +40	$4T_{OSC}$ –80	ns
T_{YLYH}	Non-Ready Time		1000	ns
T_{AVDV}(2)	Address Valid to Input Data Valid		$5T_{OSC}$ –120	ns
T_{RLDV}	$\overline{RD}$ Active to Input Valid		$3T_{OSC}$ –100	ns
T_{RHDX}	Data Hold after $\overline{RD}$ Inactive	0		ns
T_{RHDZ}	$\overline{RD}$ Inactive to Input Data Float	0	T_{OSC} –25	ns
T_{AVGV}(2,3)	Address Valid to BUSWIDTH Valid		$2T_{OSC}$ –125	ns
T_{LLGX}(3)	BUSWIDTH Hold after ALE/$\overline{ADV}$ Low	T_{OSC} +40		ns
T_{LLGV}(3)	ALE/$\overline{ADV}$ Low to BUSWIDTH Valid		T_{OSC} –75	ns
T_{RLPV}	Reset Low to Ports Valid		$10T_{OSC}$	ns

Notes:

1. If the 64-pin device is being used then this timing can be generated by assuming that the CLKOUT falling edge has occurred at 2Tosc+55 (TLLCH (max) + TCHCL (max)) after the falling edge of ALE.
2. The term "Address Valid" applies to AD0–15, $\overline{BHE}$ and INST.
3. Pins not bonded out on 64-pin devices.

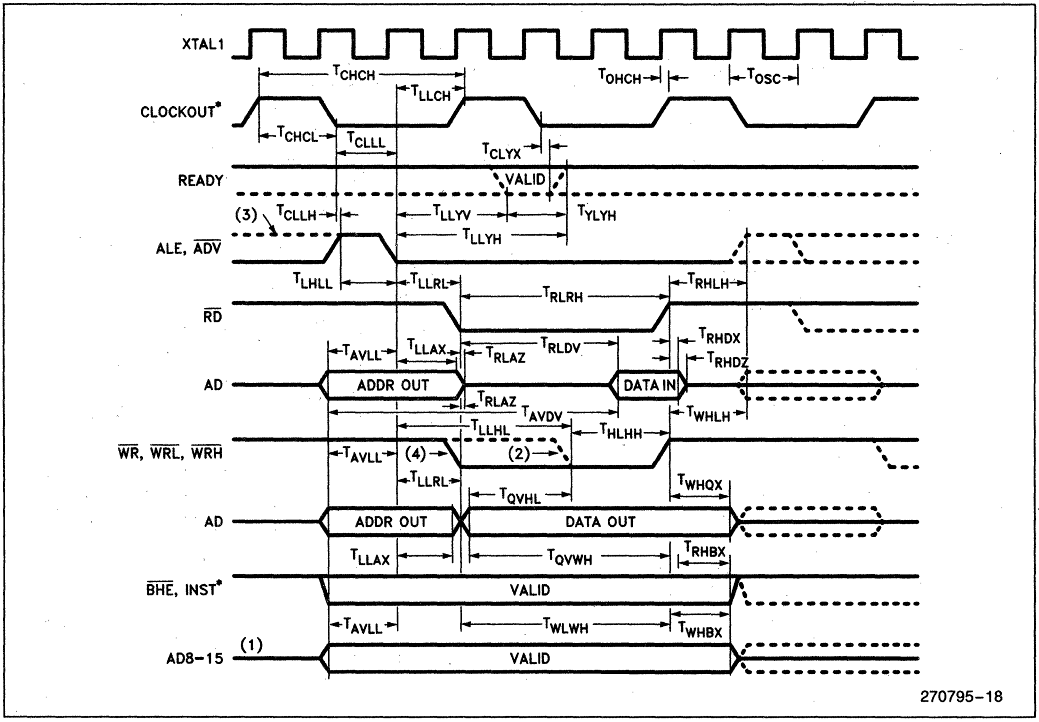
TIMING RESPONSES (MCS-96 devices meet these specs.)

Symbol	Parameter	Min	Max	Units
F_{XTAL}	Oscillator Frequency	6.0	12.0	MHz
T_{OSC}	Oscillator Period	83	166	ns
T_{OHCH}[3]	XTAL1 Rising Edge to Clockout Rising Edge	0	120[3]	ns
T_{CHCH}[3]	CLKOUT Period[2]	3Tosc[2]	3Tosc[2]	ns
T_{CHCL}[3]	CLKOUT High Time	Tosc − 35	Tosc + 10	ns
T_{CLLH}[3]	CLKOUT Low to ALE High	−20	+25	ns
T_{LLCH}[3]	ALE/$\overline{\text{ADV}}$ Low to CLKOUT High	Tosc − 25	Tosc + 45	ns
T_{LHLL}	ALE/$\overline{\text{ADV}}$ High Time	Tosc − 30	Tosc + 35[4]	ns
T_{AVLL}[5]	Address Setup to End of ALE/$\overline{\text{ADV}}$	Tosc − 50		ns
T_{RLAZ}[6]	$\overline{\text{RD}}$ or $\overline{\text{WR}}$ Low to Address Float	Typ. = 0	10	ns
T_{LLRL}	End of ALE/$\overline{\text{ADV}}$ to $\overline{\text{RD}}$ or $\overline{\text{WR}}$ Active	Tosc − 40		ns
T_{LLAX}[6]	Address Hold after End of ALE/$\overline{\text{ADV}}$	Tosc − 40		ns
T_{WLWH}	$\overline{\text{WR}}$ Pulse Width	3Tosc − 35		ns
T_{QVWH}	Output Data Valid to End of $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$	3Tosc − 60		ns
T_{WHQX}	Output Data Hold after $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$	Tosc − 50		ns
T_{WHLH}	End of $\overline{\text{WR}}$/$\overline{\text{WRL}}$/$\overline{\text{WRH}}$ to ALE/$\overline{\text{ADV}}$ High	Tosc − 75		ns
T_{RLRH}	$\overline{\text{RD}}$ Pulse Width	3Tosc − 30		ns
T_{RHLH}	End of $\overline{\text{RD}}$ to ALE/$\overline{\text{ADV}}$ High	Tosc − 45		ns
T_{CLLL}[3]	CLOCKOUT Low to ALE/$\overline{\text{ADV}}$ Low	Tosc − 40	Tosc + 35	ns
T_{RHBX}[3]	$\overline{\text{RD}}$ High to INST, $\overline{\text{BHE}}$, AD8-15 Inactive	Tosc − 25	Tosc + 30	ns
T_{WHBX}[3]	$\overline{\text{WR}}$ High to INST, $\overline{\text{BHE}}$, AD8-15 Inactive	Tosc − 50	Tosc + 100	ns
T_{HLHH}	$\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ High	2Tosc − 35	2Tosc + 40	ns
T_{LLHL}	ALE/$\overline{\text{ADV}}$ Low to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low	2Tosc − 30	2Tosc + 55	ns
T_{QVHL}	Output Data Valid to $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ Low	Tosc − 60		ns

NOTES:

1. If more than one wait state is desired, add 3Tosc for each additional wait state.
2. CLKOUT is directly generated as a divide by 3 of the oscillator. The period will be 3Tosc ± 10 ns if Tosc is constant and the rise and fall times on XTAL1 are less than 10 ns.
3. Pins not bonded out on 64-lead package.
4. Max spec applies only to ALE. Min spec applies to both ALE and $\overline{\text{ADV}}$.
5. The term "Address Valid" applies to AD0–15, $\overline{\text{BHE}}$ and INST.
6. The term " Address" in this definition applies to AD0–7 for 8-bit cycles, and AD0–15 for 16-bit cycles.

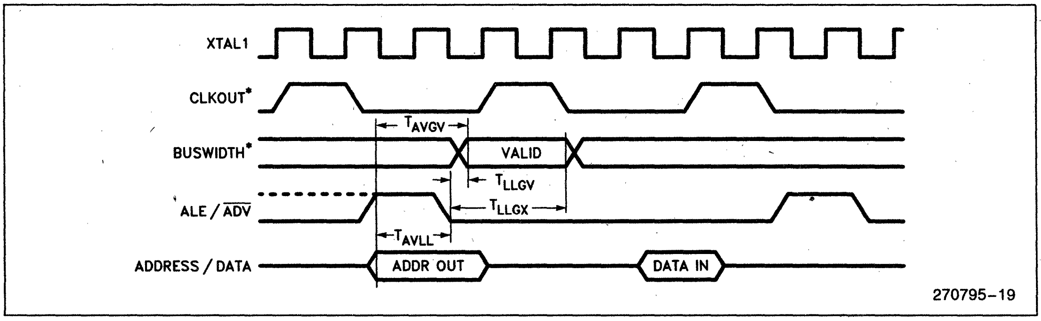
WAVEFORM



270795-18

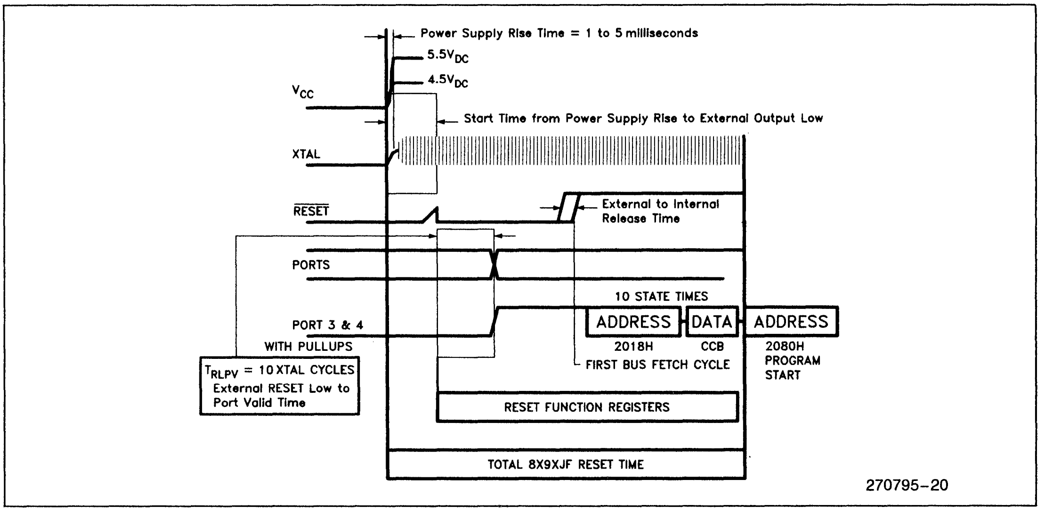
NOTES:
(1) 8-bit bus only.
(2) 8-bit or 16-bit bus and write strobe mode selected.
(3) When ADV selected.
(4) 8- or 16-bit bus and no write strobe mode selected.

WAVEFORM—BUSWIDTH PIN*



270795-19

*Not available on 64-lead package.


Power Supply Rise Time = 1 to 5 milliseconds
5.5V_{DC}
4.5V_{DC}
V_{CC}
Start Time from Power Supply Rise to External Output Low
XTAL
RESET
External to Internal Release Time
PORTS
10 STATE TIMES
PORT 3 & 4 WITH PULLUPS
ADDRESS
DATA
ADDRESS
2018H
CCB
2080H PROGRAM START
FIRST BUS FETCH CYCLE
T_{RLPV} = 10 XTAL CYCLES External RESET Low to Port Valid Time
RESET FUNCTION REGISTERS
TOTAL 8X9XJF RESET TIME

270795–20

TARGETED A.C. CHARACTERISTICS –SERIAL PORT–SHIFT REGISTER MODE

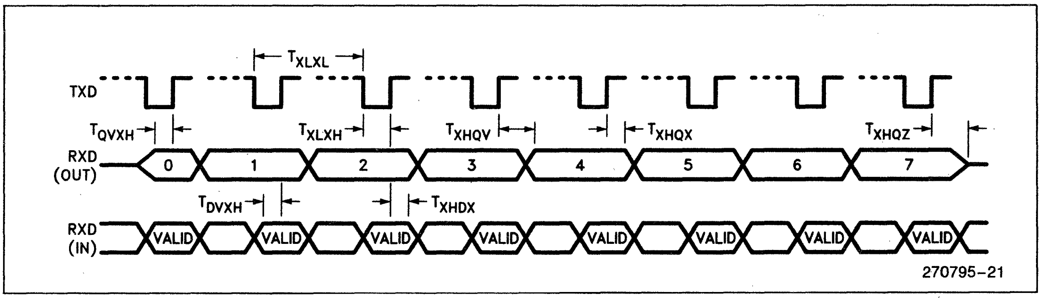
SERIAL PORT TIMING–SHIFT REGISTER MODE

Test Conditions: T_A = 0°C to +70°C; V_{CC} = 5V ± 10%; V_{SS} = 0V; Load Capacitance = 80 pF

Symbol	Parameter	Min	Max	Units
T_{XLXL}	Serial Port Clock Period	$8T_{OSC}$		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge	$4T_{OSC}$ –50	$4T_{OSC}$ +50	ns
T_{QVXH}	Output Data Setup to Clock Rising Edge	$3T_{OSC}$		ns
T_{XHQV}	Output Data Hold After Clock Rising Edge	$2T_{OSC}$ –50		ns
T_{XHQV}	Next Output Data Valid After Clock Rising Edge		$2T_{OSC}$ +50	ns
T_{DVXH}	Input Data Setup to Clock Rising Edge	$2T_{OSC}$ +200		ns
T_{XHDX}	Input Data Hold After Clock Rising Edge	0		ns
T_{XHQZ}	Last Clock Rising to Output Float		$5T_{OSC}$	ns

WAVEFORM–SERIAL PORT–SHIFT–REGISTER MODE

SERIAL PORT WAVEFORM–SHIFT REGISTER MODE

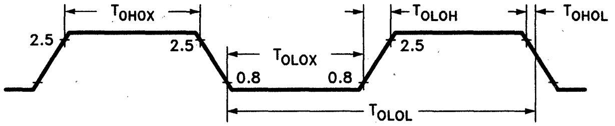


270795–21

EXTERNAL CLOCK DRIVE

Symbol	Parameter	Min	Max	Units
$1/T_{OLOL}$	Oscillator Frequency	6	12	MHz
T_{OHOX}	High Time	25		ns
T_{OLOX}	Low Time	25		ns
T_{OLOH}	Rise Time		15	ns
T_{OHOL}	Fall Time		15	ns

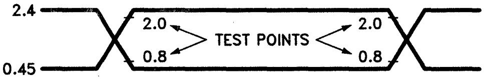
EXTERNAL CLOCK DRIVE WAVEFORMS



270795-22

An external oscillator may encounter as much as a 100 pF load at XTAL1 when it starts-up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{IL} and V_{IH} specifications the capacitance will not exceed 20 pF.

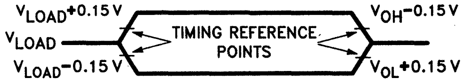
A.C. TESTING INPUT, OUTPUT WAVEFORM



270795-23

A.C. Testing inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0". Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

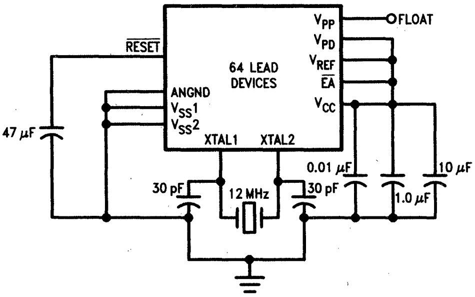
FLOAT WAVEFORM



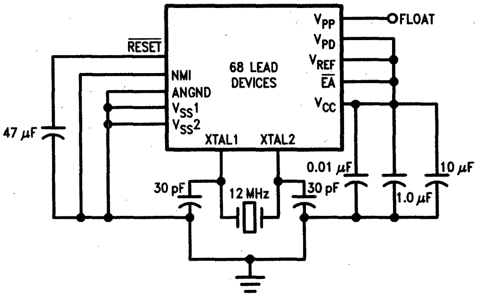
270795-24

For Timing Purposes a Port Pin is no Longer Floating when a 100 mV change from Load Voltage Occurs, and Begins to Float when a 100 mV change from the Loaded V_{OH}/V_{OL} Level occurs $I_{OL}/I_{OH} \geq \pm 15$ mA.

Minimum Hardware Configuration Circuits



270795-25



270795-26

TARGETED A/D CONVERTER SPECIFICATIONS

A/D Converter operation is verified only on the 8097JF, 8397JF, and 8797JF.

The absolute conversion accuracy is dependent on the accuracy of V_{REF}. The specifications given below assume adherence to the Operating Conditions section of these data sheets. Testing is done at V_{REF} = 5.120V.

Operating Conditions

V_{CC}, V_{PD}, V_{REF}	4.5V to 5.5V
V_{SS}, ANGND	0.0V
T_A	0°C to 70°C
F_{OSC}	6.0 to 12.0 MHz
Test Conditions:	
V_{REF}	5.120V

Parameter	Typical*	Minimum	Maximum	Units**	Notes
Resolution		1024 10	1024 10	Levels Bits	
Absolute Error		0	±4	LSBs	
Full Scale Error	−0.5 ±0.5			LSBs	
Zero Offset Error	±0.5			LSBs	
Non-Linearity		0	±4	LSBs	
Different Non-Linearity		> −1	+2	LSBs	
Channel-to-Channel Matching		0	±1	LSBs	
Repeatablity	±0.25			LSBs	
Temperature Coeffecients: Offset Full Scale Differential Non-Linearity	 0.009 0.009 0.009			 LSB/°C LSB/°C LSB/°C	
Off Isolation		−60		dB	1, 3
Feedthrough	−60			dB	1
V_{CC} Power Supply Rejection	−60			dB	1
Input Resistance		1K	5K	Ω	
D.C. Input Leakage		0	3.0	μA	
Sample Delay		$3T_{OSC}-50$	$3T_{OSC}+50$	ns	2
Sample Time		$12T_{OSC}-50$	$12T_{OSC}+50$	ns	
Sampling Capacitor			2	pF	

NOTES:

* These values are excepted for most devices at 25°C.

** An "LSB", as used here, is defined in the glossary which follows and has a value of approximately 5 mV.

1. DC to 100 KHz.
2. For starting the A/D with an HSO Command.
3. Multiplexer Break-Before-Make Guaranteed.

OTP EPROM SPECIFICATIONS

TARGETED A.C. EPROM PROGRAMMING CHARACTERISTICS

Operating Conditions: Load Capacitance = 150 pF, T_A = 25°C ±5°C, V_{CC}, V_{PD}, V_{REF} = 5.0V ± 0.5V, V_{SS}, AGND = 0V, V_{PP} = 12.75V ± 0.25V, EA = 11V ± 2.0V, f_{OSC} = 6.0 MHz

Symbol	Parameter	Min	Max	Units
T_{AVLL}	ADDRESS/COMMAND Valid to PALE Low	0		T_{OSC}
T_{LLAX}	ADDRESS/COMMAND Hold After PALE Low	80		T_{OSC}
T_{DVPL}	Output Data Setup Before $\overline{\text{PROG}}$ Low	0		T_{OSC}
T_{PLDX}	Data Hold After $\overline{\text{PROG}}$ Falling	80		T_{OSC}
T_{LLLH}	PALE Pulse Width	180		T_{OSC}
T_{PLPH}	$\overline{\text{PROG}}$ Pulse Width	250 T_{OSC}	100 µS + 144T_{OSC}	
T_{LHPL}	PALE High to $\overline{\text{PROG}}$ Low	250		T_{OSC}
T_{PHLL}	$\overline{\text{PROG}}$ High to Next PALE Low	600		T_{OSC}
T_{PHDX}	Data Hold After $\overline{\text{PROG}}$ High	30		T_{OSC}
T_{PHVV}	$\overline{\text{PROG}}$ HIgh to PVER/$\overline{\text{PDO}}$ Valid	500		T_{OSC}
T_{LLVH}	PALE Low to PVER/$\overline{\text{PDO}}$ High	100		T_{OSC}
T_{PLDV}	$\overline{\text{PROG}}$ Low to VERIFICATION/DUMP Data Valid	100		T_{OSC}
T_{SHLL}	RESET High to First PALE Low (not shown)	2000		T_{OSC}

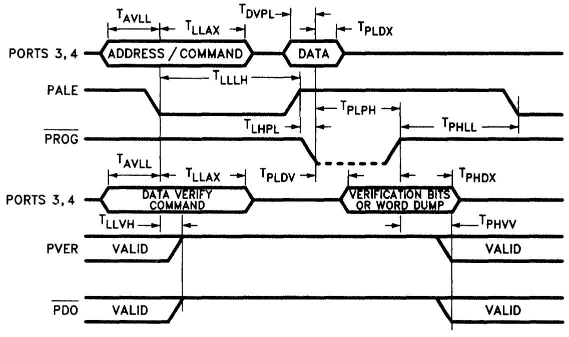
NOTE:
Run-time programming is done with F_{OSC} = 6.0 MHz to 12.0 MHz, V_{CC}, V_{PD}, V_{REF} = 5V ± 0.5V, T_A = 25°C to ±5°C and V_{PP} = 12.75V. For run-time programming over a full operating range, contact the factory.

TARGETED D.C. EPROM PROGRAMMING CHARACTERISTICS

Symbol	Parameter	Min	Max	Units
I_{PP}	V_{PP} Supply Current (Whenever Programming)		100	mA
V_{PP}	Programming Supply Voltage	12.75 ±0.25		V
V_{EA}	$\overline{\text{EA}}$ Programming Voltage	11 ±2.0		V

NOTE:
V_{PP} must be within 1V of V_{CC} while V_{CC} < 4.5V. V_{PP} must not have a low impedance path to ground or V_{SS} while V_{CC} > 4.5V.

WAVEFORM—EPROM PROGRAMMING (OTP)



270795-27

Reserved location warning: Intel Reserved addresses can not be used by applications which use 8X9XJF internal ROM/EPROM. The data read from a reserved location is not guaranteed, and a write to any reserved location could cause unpredictable results. When attempting to program Intel Reserved addresses, the data must be 0FFFFH to ensure a harmless result. A memory map indicating reserved locations on the 8X9XJF is shown in Figure 2.

Intel Reserved locations, when mapped to external memory, must be filled with 0FFFFH to ensure compatibility with future devices.

POWER SUPPLY SEQUENCE WHILE PROGRAMMING

For any 879XJF that is in any programming mode, high voltages must be applied to the device. To avoid damaging the devices, the following rules must not be violated. See the 1990 Embedded Controller Handbook MCS-96 Hardware Design Chapter for programming mode information.

RULE #1— V_{PP} must not have a low impedance path to ground when V_{CC} is above 4.5V.

RULE #2— V_{CC} must be above 4.5V before V_{PP} can be higher than 5.0V.

RULE #3— V_{PP} must be within 1V of V_{CC} while V_{CC} is below 4.5V.

RULE #4— All voltages must be within tolerance and the oscillator stable before $\overline{\text{RESET}}$ rises.

RULE #5— $\overline{\text{EA}}$ must be brought high to place the device in programming mode before V_{PP} is brought high.

To adhere to these rules, the following power up and power down sequences can be followed.

POWER UP

$\overline{\text{RESET}}$ = 0;
CLOCK ON; if using an external clock
    ; instead of an oscillator
$V_{CC} = V_{PP} = V_{EA} = 5V$;
PALE = $\overline{\text{PROG}}$ = PORT 34 = V_{IH};*
SID AND PMODE VALID;
$\overline{\text{EA}}$ = 12.75V;
V_{PP} = 12.75V;
WAIT; wait for supplies and clock to
    ; settle
$\overline{\text{RESET}}$ = 5V;
WAIT Tshll; See Data Sheet
BEGIN;

POWER DOWN

$\overline{\text{RESET}}$ = 0;
V_{PP} = 5V;
$\overline{\text{EA}}$ = 5V;
PALE = $\overline{\text{PROG}}$ = SID = PMODE = PORT34 = 0V;
$V_{CC} = V_{PP} = V_{EA} = 0V$;
CLOCK OFF;

NOTE:
*V_{IH} = Logical "1", 2.4V Minimum

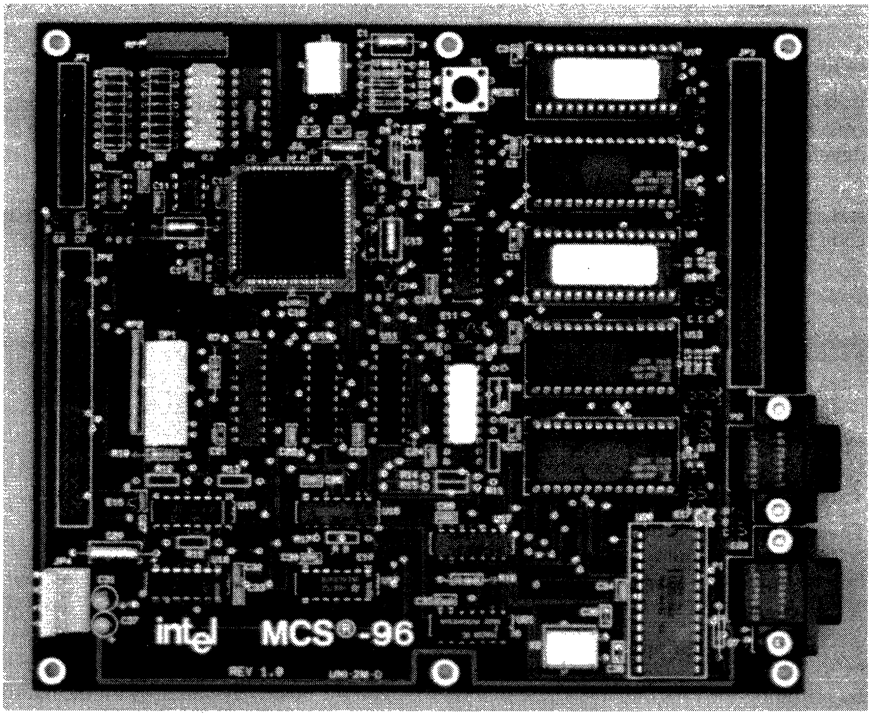
One final note on power up, power down. The maximum limit on V_{PP} must never be violated, even for an instant. Therefore, an RC rise to the desired V_{PP} is recommended. V_{PP} is also sensitive to instantaneous voltage steps. This also can be avoided by using an RC ramp on V_{PP}.

REVISION HISTORY

This is the first datasheet for the 8X9XJF.

EV8097BH EVALUATION BOARD

PRELIMINARY



EV8097BH FEATURES

- **Zero Wait-State 12 MHz Execution Speed**
- **24K Bytes of ROMsim**
- **Flexible Wait-State, Buswidth, Chip-Select Controller**
- **Concurrent Interrogation of Memory and Registers**
- **Sixteen Software Breakpoints**
- **Two Single-Step Modes**
- **High-Level Language Support**
- **Symbolic Debug**
- **RS-232-C Communication Link**

EVALUATION TOOL

Intel's EV8097BH evaluation board provides a hardware environment for code execution and software debugging at a relatively low cost. The board features the 8097BH 16-bit microcontroller from the industry standard MCS®-96 family. The board allows you to take full advantage of the power of the MCS-96. The EV8097BH provides zero wait-state, 12 MHz execution of a user's code. Plus, its memory (ROMsim) can be reconfigured to match your planned memory system, allowing for exact analysis of code execution speeds in a particular application.

**IBM PC, XT, AT and DOS are registered trademarks of International Business Corporation.

intel®

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied. Information contained herein supercedes previously published specifications on these devices from Intel.

MARCH 1989
Order number: 270739-001

Popular features such as a symbolic debug, single-line assembler, disassembler, single-step program execution, and sixteen software breakpoints are standard on the EV8097BH. Intel provides a complete code development environment using assembler (ASM-96) as well as high-level languages such as Intel's iC-96 or PL/M-96 to accelerate development schedules.

The evaluation board is hosted on an IBM PC* or BIOS-compatible clone, already a standard development solution in most of today's engineering environments. The source code for the on-board monitor (written in ASM-96) is public domain. The program is about 1K, and can be easily modified to be included in your target hardware. In this way, the provided PC host software can be used throughout the development phase.

FULL SPEED EXECUTION

The EV8097BH executes your code from on-board ROMsim at 12 MHz with zero wait-states. By changing crystals on the 8097BH, any slower execution speed can be evaluated. The board's host interface timing is not affected by this crystal change.

24K BYTES OF ROMSIM

The board comes with 24K bytes of SRAM to be used as ROMsim for your application code and as data memory if needed. 16K bytes of this memory are configured as sixteen bits wide and 8K bytes are configured as eight bits wide. You can therefore evaluate the speed of the part executing from either buswidth.

FLEXIBLE MEMORY DECODING

By changing the Programable Logic Device (PLD) on the board, the memory on the board can be made to look like the memory system planned for your hardware application. The PLD controls the buswidth of the 8097BH and the chip-select inputs on the board. It also controls the number of wait states (zero to four) generated by the 8097BH during a memory cycle. These features can all be selected with 256 byte boundaries of resolution.

CONCURRENT INTERROGATION OF MEMORY AND REGISTERS

The monitor for the EV8097BH allows you to read and modify internal registers and external memory while your code is running in the board.

SIXTEEN SOFTWARE BREAKPOINTS

There are sixteen breakpoints available which automatically substitute a TRAP instruction for your instruction at the breakpoint location. The substitution occurs when execution is started. If the code is halted or a breakpoint is reached, your code is restored in the ROMsim.

TWO STEP MODES

There are two single-step modes available. The first stepping mode locks out all interrupts which might occur during the step. The second mode enables interrupts, and treats subroutine calls and interrupt routines as indivisible instructions.

HIGH LEVEL LANGUAGE SUPPORT

The host software for the EV8097BH board is able to load absolute object code generated by ASM-96, iC-96, PL/M-96 or RL-96, all of which are available from Intel.

SYMBOLIC DEBUG

The host has a Single-Line Assembler, and a Disassmbler that recognize symbolics generated by Intel software tools.

RS-232-C COMMUNICATION LINK

The EV8097BH communicates with the host using an Intel 82510 UART provided on board. This frees the on-chip UART of the 8097BH for your application.

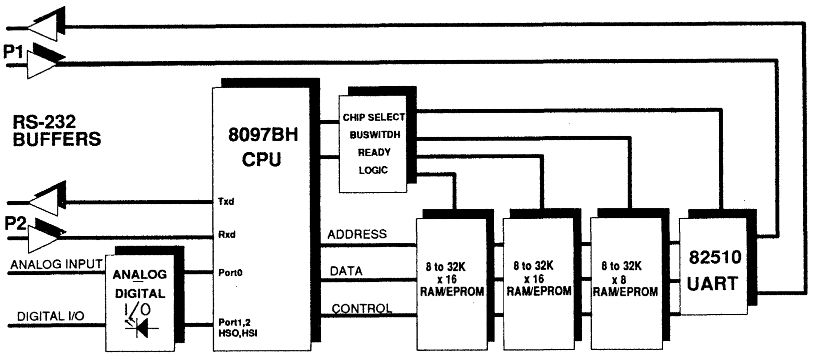
POWER REQUIREMENTS

The EV8097BH board requirs 5 volts at 450 mA. If the on-board LED's are disabled, the current drops to 300 mA. The board also requires +/- 12 volts at 25 mA.

PERSONAL COMPUTER REQUIREMENTS

The EV8097BH Evaluation Board is hosted on an IBM PC*, XT*, AT* or BIOS-compatible clone. The PC must meet the following minimum requirements:

- 512K Bytes of Memory
- One 360K Byte Floppy Disk Drive
- PC-DOS* 3.1 or Later
- A Serial Port (COM1 or COM2) at 9600 Baud
- ASM-96, iC-96 or PL/M-96
- An ASCII text editor such as AEDIT



Block Diagram of the EV8097BH Board

MCS®-96 Instruction Set

3

MCS®-96 INSTRUCTION SET

OVERVIEW

This chapter of the manual gives a description of each instruction recognized by the MCS®-96 architecture. The instructions are sorted alphabetically by the assembly language mnemonic.

Additional information including instruction execution times and a description of the different addressing modes can be found in the following documents:

MSC-96 MACRO ASSEMBLER USER'S GUIDE
Order Number 122048 (Intel systems)
Order Number 122351 (DOS systems)

MCS-96 UTILITIES USER'S GUIDE
Order Number 122049 (Intel systems)
Order Number 122356 (DOS systems)

PL/M-96 USER'S GUIDE
Order Number 122134 (Intel systems)
Order Number 122361 (DOS systems)

C-96 USER'S GUIDE
Order Number 167632

80C196KC USER'S GUIDE
Order Number 270704

80C196KB USER'S GUIDE
Order Number 270651

MCS-96 ARCHITECTURAL OVERVIEW
Order Number 270250

The instruction set descriptions in the following sections do not always show the effect on the program counter (PC). Unless otherwise specified, all instructions increment the PC by the number of bytes in the instruction.

A set of acronyms are used to make the instruction set descriptions easier to read, their definitions are listed below:

aa. A two bit field within an opcode which selects the basic addressing mode user. This field is only present in those opcodes which allow address mode options. The encoding of the field is as follows:

aa	Addressing mode
00	Register direct
01	Immediate
10	Indirect
11	Indexed

breg. A byte register in the internal register file. When confusion could exist as to whether this field refers to a source or a destination register it will be prefixed with an "S" or a "D".

baop. A byte operand which is addressed by any of the address modes.

bitno. A three bit field within an instruction op-code which selects one of the eight bits in a byte.

wreg. A word register in the internal register file. When confusion could exist as to whether this field refers to a source register or a destination register it will be prefixed with an "S" or a "D".

waop. A word operand which is addressed by any of the address modes.

Lreg. A 32-bit register in the internal register file.

cadd. An address in the program code.

Flag Settings. The modification to the flag setting is shown for each instruction. A checkmark (✓) means that the flag is set or cleared as appropriate. A hyphen means that the flag is not modified. A one or zero (1) or (0) indicates that the flag will be in that state after the instruction. An up arrow (↑) indicates that the instruction may set the flag if it is appropriate but will not clear the flag. A down arrow (↓) indicates that the flag can be cleared but not set by the instruction. A question mark (?) indicates that the flag will be left in an indeterminant state after the operation.

Generic Jumps and Calls. The assembler for the MCS-96 family provides for generic jumps and calls. For all of the conditional jump instructions a "B" can be substituted for the "J" and the assembler will generate a code sequence which is logically equivalent but can reach anywhere in the memory. A JH can only jump about 128 locations from the current program counter; a BH can jump anywhere in memory. In a like manner a BR will cause a SJMP or LJMP to be generated as appropriate and a CALL will cause a SCALL or LCALL to be generated. The assembler user's guide should be consulted for the algorithms used by the assembler to convert these generic instructions into actual machine instructions.

Indirect Shifts. The indirect shift operations use registers 24 through 255 (18H–0FFH), since 0–15 are direct operators and registers 16 through 23 are Special Function Registers. Note that indirect shifts through SFRs are illegal operations.

The maximum shift count is 31 (1FH). Count values above this will be truncated to the 5 least significant bits.

1. ADD (Two Operands) — ADD WORDS

Operation: The sum of the two word operands is stored into the destination (leftmost) operand.

(DEST) ← (DEST) + (SRC)

Assembly Language Format:

```
        DST     SRC
ADD     wreg,   waop
```

Object Code Format: [011001aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

2. ADD (Three Operands) — ADD WORDS

Operation: The Sum of the second and third word operands is stored into the destination (leftmost) operand.

(DEST) ← (SRC1) + (SRC2)

Assembly Language Format:

```
        DST     SRC1    SRC2
ADD     Dwreg,  Swreg,  waop
```

Object Code Format: [010001aa] [waop] [Swreg] [Dwreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

3. ADDB (Two Operands) — ADD BYTES

Operation: The sum of the two byte operands is stored into the destination (leftmost) operand.

(DEST) ← (DEST) + (SRC)

Assembly Language Format:

	DST	SRC
ADDB	breg,	baop

Object Code Format: [011101aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

4. ADDB (Three Operands) — ADD BYTES

Operation: The sum of the second and third byte operands is stored into the destination (leftmost) operand.

(DEST) ← (SRC1) + (SRC2)

Assembly Language Format:

	DST	SRC1	SRC2
ADDB	Dbreg,	Sbreg,	baop

Object Code Format: [010101aa] [baop] [Sbreg] [Dbreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

5. ADDC — ADD WORDS WITH CARRY

Operation: The sum of the two word operands and the carry flag (0 or 1) is stored into the destination (leftmost) operand.

(DEST) ← (DEST) + (SRC) + C

Assembly Language Format:

	DST	SRC
ADDC	wreg,	waop

Object Code Format: [101001aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
↓	✓	✓	✓	↑	—

6. ADDCB — ADD BYTES WITH CARRY

Operation: The sum of the two byte operands and the carry flag (0 or 1) is stored into the destination (leftmost) operand.

(DEST) ← (DEST) + (SRC) + C

Assembly Language Format:

	DST	SRC
ADDCB	breg,	baop

Object Code Format: [101101aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
↓	✓	✓	✓	↑	—

7. AND (Two Operands) — LOGICAL AND WORDS

Operation: The two word operands are ANDed, the result having a 1 only in those bit positions where both operands had a 1, with zeroes in all other bit positions. The result is stored into the destination (leftmost) operand.

(DEST) ← (DEST) AND (SRC)

Assembly Language Format:

```
        DST    SRC
AND     wreg,  waop
```

Object Code Format: [011000aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

8. AND (Three Operands) — LOGICAL AND WORDS

Operation: The second and third word operands are ANDed, the result having a 1 only in those bit positions where both operands had a 1, with zeroes in all other bit positions. The result is stored into the destination (leftmost) operand.

(DEST) ← (SRC1) AND (SRC2)

Assembly Language Format:

```
        DST     SRC1    SRC2
AND     Dwreg,  Swreg,  waop
```

Object Code Format: [010000aa] [waop] [Swreg] [Dwreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

9. ANDB (Two Operands) — LOGICAL AND BYTES

Operation: The two byte operands are ANDed, the result having a 1 only in those bit positions where both operands had a 1, with zeroes in all other bit positions. The result is stored into the destination (leftmost) operand.

(DEST) ← (DEST) AND (SRC)

Assembly Language Format:

	DST	SRC
ANDB	breg,	baop

Object Code Format: [011100aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

10. ANDB (Three Operands) — LOGICAL AND BYTES

Operation: The second and third byte operands are ANDed, the result having a 1 only in those bit positions where both operands had a 1, with zeroes in all other bit positions. The result is stored into the destination (leftmost) operand.

(DEST) ← (SRC1) AND (SRC2)

Assembly Language Format:

	DST	SRC1	SRC2
ANDB	Dbreg,	Sbreg,	baop

Object Code Format: [010100aa] [baop] [Sbreg] [Dbreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

11. BMOV — BLOCK MOVE (80C196KB and 80C196KC only)

Operation: This instruction is used to move a block of word data from one location in memory to another. The source and destination addresses are calculated using the indirect with auto-increment addressing modes. A long register addresses the source and destination pointers which are stored in adjacent word registers. The number of transfers is specified by a word register. The blocks of data can reside anywhere in memory, but should not overlap.

```
COUNT ← (CNTREG)
LOOP: SRCPTR ← (PTRS)
DSTPTR ← (PTRS + 2)
(DSTPTR) ← (SRCPTR)
(PTRS) ← SRCPTR + 2
(PTRS) + 2) ← DSTPTR + 2
COUNT ← COUNT − 1
if COUNT <> 0 then go to LOOP
```

Assembly Language Format:

```
      PTRS CNTREG
BMOV  Lreg, wreg
```

Object Code Format: [11000001] [wreg] [Lreg]

Flags Affected							
Z	N	V	VT	C	X	I	ST
—	—	—	—	—	X	—	—

NOTES:

1. CNTREG does not get decremented during the instruction.
2. It is easy to unintentionally create a very long un-interruptable operation with this instruction.

To provide an interruptable version of the BMOV for large blocks, the BMOV instruction can be used with the DJNZ instruction. This is possible because the pointers are modified, but CNTREG is not. Consider the example:

```
       LD    PTRS, SRC          ;Pointer to base of sources table
       LD    PTRS+2, DST        ;Pointer to base of destination table
       LD    CNTREG, #COUNT     ;Number of bytes to move per set
       LD    CNTSET, #SETS      ;Number of sets to move
Move:  BMOV  PTRS, CNTREG       ;Move one set
       DJNZ  CNTSET, MOVE       ;Decrement set counters and move again
```

12. BMOVI — INTERRUPTABLE BLOCK MOVE (80C196KC only)

Operation: This instruction is used to move a block of word data from one location in memory to another and is interruptable. The source and destination addresses are calculated using the indirect with auto-increment addressing modes. A long register addresses the source and destination pointers which are stored in adjacent word registers. The number of transfers is specified by a word register. The blocks of data can reside anywhere in memory, but should not overlap.

```
COUNT ← (CNTREG)
LOOP: SRCPTR ← (PTRS)
DSTPTR ← (PTRS + 2)
(DSTPTR) ← (SRCPTR)
(PTRS) ← SRCPTR + 2
(PTRS) + 2) ← DSTPTR + 2
COUNT ← COUNT − 1
if COUNT <> 0 then go to LOOP
```

PTRS CNTREG

Assembly Language Format: BMOVI Lreg, wreg

Object Code Format: [10101101] [wreg] [Lreg]

Flags Affected							
Z	N	V	VT	C	X	I	ST
—	—	—	—	—	X	—	—

NOTES:

1. CNTREG does not get decremented during the instruction. However, if the BMOVi is interrupted, CNTREG will be updated. Therefore, CNTREG must be reloaded before starting a BMOVi.

13. BR (Indirect) — BRANCH INDIRECT

Operation: The execution continues at the address specified in the operand word register.

PC ← (DEST)

Assembly Language Format: BR [wreg]

Object Code Format: [11100011] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

14. CLR — CLEAR WORD

Operation: The value of the word operand is set to zero.

(DEST) ← 0

Assembly Language Format: CLR wreg

Object Code Format: [00000001] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
1	0	0	0	—	—

15. CLRB — CLEAR BYTE

Operation: The value of the byte operand is set to zero.
(DEST) ← 0

Assembly Language Format: CLRB breg

Object Code Format: [00010001] [breg]

Flags Affected					
Z	N	C	V	VT	ST
1	0	0	0	—	—

16. CLRC — CLEAR CARRY FLAG

Operation: The value of the carry flag is set to zero.
C ← 0

Assembly Language Format: CLRC

Object Code Format: [11111000]

Flags Affected					
Z	N	C	V	VT	ST
—	—	0	—	—	—

17. CLRVT — CLEAR OVERFLOW TRAP

Operation: The value of the overflow-trap flag is set to zero.

VT ← 0

Assembly Language Format: CLRVT

Object Code Format: [11111100]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	0	—

18. CMP — COMPARE WORDS

Operation: The source (rightmost) word operand is subtracted from the destination (leftmost) word operand. The flags are altered but the operands remain unaffected. The carry flag is set as complement of borrow.

(DEST) — (SRC)

Assembly Language Format:

```
        DST     SRC
CMP     wreg,   waop
```

Object Code Format: [100010aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

19. CMPB — COMPARE BYTES

Operation: The source (rightmost) byte operand is subtracted from the destination (leftmost) byte operand. The flags are altered but the operands remain unaffected. The carry flag is set as complement of borrow.

(DEST) — (SRC)

Assembly Language Format:

	DST	SRC
CMPB	breg,	baop

Object Code Format: [100110aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

20. CMPL — COMPARE LONG (80C196KB and 80C196KC only)

Operation: This instruction is used to compare the magnitudes of two double word (long) operands. The operands are specified using the direct addressing mode. Five PSW flags are set following this operation, but the operands are not affected.

DST-SRC

Assembly Language Format:

	DST	SRC
CMPL	Lreg,	Lreg

Object Code Format: [11000101] [src Lreg] [dst#Lreg]

Flags Affected							
Z	N	V	VT	C	X	I	ST
✓	✓	✓	✓	✓	X	—	—

21. DEC — DECREMENT WORD

Operation: The value of the word operand is decremented by one.
(DEST) ← (DEST) — 1

Assembly Language Format: DEC wreg

Object Code Format: [00000101] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

22. DECB — DECREMENT BYTE

Operation: The value of the byte operand is decremented by one.
(DEST) ← (DEST) — 1

Assembly Language Format: DECB breg

Object Code Format: [00010101] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

23. DI — DISABLE INTERRUPTS

Operation: Interrupts are disabled. Interrupt-calls will not occur after this instruction.
Interrupt Enable (PSW.9) ← 0

Assembly Language Format: DI

Object Code Format: [11111010]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

24. DIV — DIVIDE INTEGERS

Operation: This instruction divides the contents of the destination LONG-INTEGER operand by the contents of the INTEGER word operand, using signed arithmetic. The low order word of the destination (i.e., the word with the lower address) will contain the quotient; the high order word will contain the remainder.

(low word DEST) ← (DEST) / (SRC)
(high word DEST) ← (DEST) MOD (SRC)
The above two statements are performed concurrently.

Assembly Language Format:

```
        DST     SRC
DIV     lreg,   waop
```

Object Code Format: [11111110] [100011aa] [waop] [lreq]

Flags Affected						
Z	N	C	V	VT	ST	
—	—	—	?	↑	—	8096BH
—	—	—	✓	↑	—	80C196KB, 80C196KC

25. DIVB — DIVIDE SHORT-INTEGERS

Operation: This instruction divides the contents of the destination INTEGER operand by the contents of the source SHORT-INTEGER operand, using signed arithmetic. The low order byte of the destination (i.e. the byte with the lower address) will contain the quotient; the high order byte will contain the remainder.

(low byte DEST) ← (DEST) / (SRC)
(high byte DEST) ← (DEST) MOD (SRC)
The above two statements are performed concurrently.

Assembly Language Format:

```
        DST     SRC
DIVB    wreg,   baop
```

Object Code Format: [11111110] [100111aa] [baop] [wreg]

Flags Affected						
Z	N	C	V	VT	ST	
—	—	—	?	↑	—	8096BH
—	—	—	✓	↑	—	80C196KB, 80C196KC

26. DIVU — DIVIDE WORDS

Operation: This instruction divides the content of the destination DOUBLE-WORD operand by the contents of the source WORD operand, using unsigned arithmetic. The low order word will contain the quotient; the high order WORD will contain the remainder.

(low word DEST) ← (DEST) / (SRC)
(high word DEST) ← (DEST) MOD (SRC)
The above two statements are performed concurrently.

Assembly Language Format:

```
        DST     SRC
DIVU    lreg,   waop
```

Object Code Format: [100011aa] [waop] [lreq]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	✓	↑	—

27. DIVUB — DIVIDE BYTES

Operation: This instruction divides the contents of the destination WORD operand by the contents of the source BYTE operand, using unsigned arithmetic. The low order byte of the destination, (i.e., the byte with the lower address) will contain the quotient; the high order byte will contain the remainder.

(low byte DEST) ← (DEST) / (SRC)
(high byte DEST) ← (DEST) MOD (SRC)
The above two statements are performed concurrently.

Assembly Language Format:

```
        DST     SRC
DIVUB   wreg,   baop
```

Object Code Format: [100111aa] [baop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	✓	↑	—

28. DJNZ — DECREMENT AND JUMP IF NOT ZERO

Operation: The value of the byte operand is decremented by 1. If the result is not equal to 0, the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the result of the decrement is zero then control passes to the next sequential instruction.

```
(COUNT) ← (COUNT) — 1
if (COUNT) <> 0 then
  PC ← PC + disp (sign-extended to 16 bits)
end_if
```

Assembly Language Format: DJNZ breg,cadd

Object Code Format: [11100000] [breg] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

29. DJNZW — DECREMENT AND JUMP IF NOT ZERO WORD (80C196KB and 80C196KC only)

Operation: This instruction is the same as the DJNZ except that the count is a word operand. A counter word is decremented; if the result is not zero the jump is taken. The range of the jump is −128 to +127.

```
COUNT ← COUNT ← − 1
if COUNT <> 0 then
PC ← PC + disp (sign extended)
```

Assembly Language Format: DJNZW wreg,cadd

Object Code Format: [11100001] [wreg] [disp]

Flags Affected							
Z	N	V	VT	C	X	I	ST
—	—	—	—	—	X	—	—

30. DPTS — DISPOSABLE PERIPHERAL TRANSACTION SERVER (PTS — 80C196KC only)

Operation: The PTS is disabled following the execution of this instruction.

PTS Disable (PSW.10) ← 0

Assembly Language Format: DPTS

Object Code Format: [11101010]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

31. EI — ENABLE INTERRUPTS

Operation: Interrupts are enabled following the execution of the next statement. Interrupt-calls cannot occur immediately following this instruction.

Interrupt Enable (PSW.9) ← 1

Assembly Language Format: EI

Object Code Format: [11111011]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

32. EPTS — ENABLE PERIPHERAL TRANSACTION SERVER (PTS — 80C196KC only)

Operation: The PTS is enabled following the execution of this instruction.

```
  PTS Enable (PSW.10) ← 1
```

Assembly Language Format: EPTS

Object Code Format: [11101101]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

33. EXT — SIGN EXTEND INTEGER INTO LONG-INTEGER

Operation: The low order word of the operand is sign-extended throughout the high order word of the operand.

```
if (low word DEST) < 8000H then
  (high word DEST) ← 0
else
  (high word DEST) ← 0FFFFH
end_if
```

Assembly Language Format: EXT lreg

Object Code Format: [00000110] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

34. EXTB — SIGN EXTEND SHORT-INTEGER INTO INTEGER

Operation: The low order byte of the operand is sign-extended throughout the high order byte of the operand.

```
if (low byte DEST) < 80H then
  (high byte DEST) ← 0
else
  (high byte DEST) ← 0FFH
end_if
```

Assembly Language Format: EXTB wreg

Object Code Format: [00010110] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

35. INC — INCREMENT WORD

Operation: The value of the word operand is incremented by 1.

(DEST) ← (DEST) + 1

Assembly Language Format: INC wreg

Object Code Format: [00000111] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

36. INCB — INCREMENT BYTE

Operation: The value of the byte operand is incremented by 1.
(DEST) ← (DEST) + 1

Assembly Language Format: INCB breg

Object Code Format: [00010111] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

37. IDLPD — IDLE/POWERDOWN (80C196KB and 80C196KC only)

Operation: This instruction is used for entry into the idle and powerdown modes. Selecting IDLE or POWERDOWN is done using the key operand. If the operand is not a legal key, the part executes a reset sequence. The bus controller will complete any prefetch cycle in progress before the CPU stops or resets.

```
if KEY = 1 then enter IDLE
else if KEY = 2 then enter
POWERDOWN
else execute reset.
```

Assembly Language Format: IDLPD #key (key is 8-bit value)

Object Code Format: [11110110] [key]

	Flags Affected							
	Z	N	V	VT	C	x	I	ST
Legal Key:	—	—	—	—	—	X	—	—
Illegal Key:	0	0	0	0	0	X	0	0

(— = Unchanged)

38. JBC — JUMP IF BIT CLEAR

Operation: The specified bit is tested. If it is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the bit is set (i.e., 1), control passes to the next sequential instruction.

```
if (specified bit) = 0 then
  PC <— PC + disp (sign-extened to 16 bits)
```

Assembly Language Format: JBC breg,bitno,cadd

Object Code Format: [00110bbb] [breg] [disp]
where bbb is the bit number within the specified register.

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

39. JBS — JUMP IF BIT SET

Operation: The specified bit is tested. If it is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the bit is clear (i.e., 0), control passes to the next sequential instruction.

```
if (specified bit) = 1 then
  PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JBS breg,bitno,cadd

Object Code Format: [00111bbb][breg] [disp]
where bbb is the bit number within the specified register.

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

40. JC — JUMP IF CARRY FLAG IS SET

Operation: If the carry flag is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the carry flag is clear (i.e., 0), control passes to the next sequential instruction.

```
if C = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JC cadd

Object Code Format: [11011011] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

41. JE — JUMP IF EQUAL

Operation: If the zero flag is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the zero flag is clear (i.e., 0), control passes to the next sequential instruction.

```
if Z = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JE cadd

Object Code Format: [11011111] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

42. JGE — JUMP IF SIGNED GREATER THAN OR EQUAL

Operation: If the negative flag is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the negative flag is set (i.e., 1), control passes to the next sequential instruction.

```
if N = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JGE cadd

Object Code Format: [11010110] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

43. JGT — JUMP IF SIGNED GREATER THAN

Operation: If both the negative flag and the zero flag are clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If either the negative flag or the zero flag are set (i.e., 1,) control passes to the next sequential instruction.

```
if N = 0 AND Z = 0 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JGT cadd

Object Code Format: [11010010] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

44. JH — JUMP IF HIGHER (UNSIGNED)

Operation: If the carry flag is set (i.e., 1), but the zero flag is not, the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If either the carry flag is clear or the zero flag is set, control passes to the next sequential instruction.

```
if C = 1 AND Z = 0 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JH cadd

Object Code Format: [11011001] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

45. JLE — JUMP IF SIGNED LESS THAN OR EQUAL

Operation: If either the negative flag or the zero flag are set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to + 127. If both the negative flag and the zero flag are clear (i.e., 0), control passes to the next sequential instruction.

```
if N = 1 OR Z = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JLE cadd

Object Code Format: [11011010] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

46. JLT — JUMP IF SIGNED LESS THAN

Operation: If the negative flag is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the negative flag is clear (i.e., 0), control passes to the next sequential instruction.

```
if N = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JLT cadd

Object Code Format: [11011110] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

47. JNC — JUMP IF CARRY FLAG IS CLEAR

Operation: If the carry flag is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the carry flag is set (i.e., 1), control passes to the next sequential instruction.

```
if C = 0 then
    PC <- PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JNC cadd

Object Code Format: [11010011] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

48. JNE — JUMP IF NOT EQUAL

Operation: If the zero flag is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the zero flag is set (i.e., 1), control passes to the next sequential instruction.

```
if Z = 0 then
    PC <- PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JNE cadd

Object Code Format: [11010111 [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

49. JNH — JUMP IF NOT HIGHER (UNSIGNED)

Operation: If either the carry flag is clear (i.e., 0), or the zero flag is set (i.e., 1), the distance from the end of this instruction to the target label is added to program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to + 127. If the carry flag is set (i.e., 1) and the zero flag is not, control passes to the next sequential instruction.

```
if C = 0 OR Z = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JNH cadd

Object Code Format: [11010001] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

50. JNST — JUMP IF STICKY BIT IS CLEAR

Operation: If the sticky bit flag is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the sticky bit flag is set (i.e., 1), control passes to the next sequential instruction.

```
if ST = 0 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JNST cadd

Object Code Format: [11010000] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

51. JNV — JUMP IF OVERFLOW FLAG IS CLEAR

Operation: If the overflow flag is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the overflow flag is set (i.e., 1), control passes to next sequential instruction.

```
if V = 0 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JNV cadd

Object Code Format: [11010101] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

52. JNVT — JUMP IF OVERFLOW TRAP IS CLEAR

Operation: If the overflow trap flag is clear (i.e., 0), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the overflow trap flag is set (i.e., 1), control passes to the next sequential instruction. The VT flag is cleared.

```
if VT = 0 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JNVT cadd

Object Code Format: [11010100] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	0	—

53. JST — JUMP IF STICKY BIT IS SET

Operation: If the sticky bit flag is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the sticky bit flag is clear (i.e., 0), control passes to the next sequential instruction.

```
if ST = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JST cadd

Object Code Format: [11011000] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

54. JV — JUMP IF OVERFLOW FLAG IS SET

Operation: If the overflow is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the overflow flag is clear (i.e., 0), control passes to the next sequential instruction.

```
if V = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JV cadd

Object Code Format: [11011101] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

55. JVT — JUMP IF OVERFLOW TRAP IS SET

Operation: If the overflow trap flag is set (i.e., 1), the distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the target label must be in the range of −128 to +127. If the overflow trap flag is clear (i.e., 0), control passes to the next sequential instruction. The VT flag is cleared.

```
if VT = 1 then
    PC ← PC + disp (sign-extended to 16 bits)
```

Assembly Language Format: JVT cadd

Object Code Format: [11011100] [disp]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	0	—

56. LCALL — LONG CALL

Operation: The contents of the program counter (the return address) is pushed onto the stack. Then the distance from the end of this instruction to the target label is added to the program counter, effecting the call. The operand may be any address in the entire address space.

```
SP ← SP − 2
(SP) ← PC
PC ← PC + disp
```

Assembly Language Format: LCALL cadd

Object Code Format: [11101111] [disp-low] [disp-hi]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

57. LD — LOAD WORD

Operation: The value of the source (rightmost) word operand is stored into the destination (leftmost) operand.

(DEST) ← (SRC)

Assembly Language Format:

```
        DST     SRC
LD      wreg,   waop
```

Object Code Format: [101000aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

58. LDB — LOAD BYTE

Operation: The value of the source (rightmost) byte operand is stored into the destination (leftmost) operand.

(DEST) ← (SRC)

Assembly Language Format:

```
        DST     SRC
LDB     breg,   baop
```

Object Code Format: [101100aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

59. LDBSE — LOAD INTEGER WITH SHORT-INTEGER

Operation: The value of the source (rightmost) byte operand is sign-extended and stored into the destination (leftmost) word operand.

```
(low byte DEST) ← (SRC)
if (SRC) < 80H then
    (high byte DEST) ← 0
else
    (high byte DEST) ← 0FFH
end_if
```

Assembly Language Format:

```
        DST     SRC
LDBSE   wreg,   baop
```

Object Code Format: [101111aa] [baop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

60. LDBZE — LOAD WORD WITH BYTE

Operation: The value of the source (rightmost) byte operand is zero-extended and stored into the destination (leftmost) word operand.

```
(low byte DEST) ← (SRC)
(high byte DEST) ← 0
```

Assembly Language Format:

```
        DST     SRC
LDBZE   wreg,   baop
```

Object Code Format: [101011aa] [baop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

61. LJMP — LONG JUMP

Operation: The distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The operand may be any address in the entire address space.

PC ← PC + disp

Assembly Language Format: LJMP cadd

Object Code Format: [11100111] [disp-low] [disp-hi]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

62. MUL (Two Operands) — MULTIPLY INTEGERS

Operation: The two INTEGER operands are multiplied using signed arithmetic and the 32-bit result is stored into the destination (leftmost) LONG-INTEGER operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (DEST) * (SRC)

Assembly Language Format:

```
        DST     SRC
MUL     lreg,   waop
```

Object Code Format: [11111110] [011011aa] [waop] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

63. MUL (Three Operands) — MULTIPLY INTEGERS

Operation: The second and third INTEGER operands are multiplied using signed arithmetic and the 32-bit result is stored into the destination (leftmost) LONG INTEGER operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (SRC1) * (SRC2)

Assembly Language Format:

```
        DST     SRC1    SRC2
MUL     lreg,   wreg,   waop
```

Object Code Format: [11111110] [010011aa] [waop] [wreg] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

64. MULB (Two Operands) — MULTIPLY SHORT-INTEGERS

Operation: The two SHORT-INTEGER operands are multiplied using signed arithmetic and the 16-bit result is stored into the destination (leftmost) INTEGER operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (DEST) * (SRC)

Assembly Language Format:

```
        DST     SRC
MULB    wreg,   baop
```

Object Code Format: [11111110] [011111aa] [baop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

65. MULB (Three Operands) — MULTIPLY SHORT-INTEGERS

Operation: The second and third SHORT-INTEGER operands are multiplied using signed arithmetic and the 16-bit result is stored into the destination (leftmost) INTEGER operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (SRC1) * (SRC2)

Assembly Language Format:

	DST	SRC1	SRC2
MULB	wreg,	breg	baop

Object Code Format: [11111110] [010111aa] [baop] [breg] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

66. MULU (Two Operands) — MULTIPLY WORDS

Operation: The two WORD operands are multiplied using unsigned arithmetic and the 32-bit result is stored into the destination (leftmost) DOUBLE-WORD operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (DEST) * (SRC)

Assembly Language Format:

	DST	SRC
MULU	lreg,	waop

Object Code Format: [011011aa] [waop] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

67. MULU (Three Operands) — MULTIPLY WORDS

Operation: The second and third WORD operands are multiplied using unsigned arithmetic and the 32-bit result is stored into the destination (leftmost) DOUBLE-WORD operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (SRC1) * (SRC2)

Assembly Language Format:

```
        DST     SRC1    SRC2
MULU    lreg,   wreg,   waop
```

Object Code Format: [010011aa] [waop] [wreg] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

68. MULUB (Two Operands) — MULTIPLY BYTES

Operation: The two BYTE operands are multiplied using unsigned arithmetic and the WORD result is stored into the destination (leftmost) operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (DEST) * (SRC)

Assembly Language Format:

```
        DST     SRC
MULUB   wreg,   baop
```

Object Code Format: [011111aa] [baop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

69. MULUB (Three Operands) — MULTIPLY BYTES

Operation: The second and third BYTE operands are multiplied using unsigned arithmetic and the WORD result is stored into the destination (leftmost) operand. The sticky bit flag is undefined after the instruction is executed.

(DEST) ← (SRC1) * (SRC2)

Assembly Language Format:

	DST	SRC1	SRC2
MULUB	wreg,	breg,	baop

Object Code Format: [010111aa] [baop] [breg] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	?

70. NEG — NEGATE INTEGER

Operation: The value of the INTEGER operand is negated.

(DEST) ← −(DEST)

Assembly Language Format: NEG wreg

Object Code Format: [00000011] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

71. NEGB — NEGATE SHORT-INTEGER

Operation: The value of the SHORT-INTEGER operand is negated.
(DEST) ← −(DEST)

Assembly Language Format: NEGB breg

Object Code Format: [00010011] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

72. NOP — NO OPERATION

Operation: Nothing is done. Control passes to the next sequential instruction.

Assembly Language Format: NOP

Object Code Format: [11111101]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

73. NORML — NORMALIZE LONG-INTEGER

Operation: The LONG-INTEGER operand is normalized; i.e., it is shifted to the left until its most significant bit is 1. If the most significant bit is still 0 after 31 shifts, the process stops and the zero flag is set. The number of shifts actually performed is stored in the second operand.

```
(COUNT) ← 0
do while (MSB(DEST) = 0) AND ((COUNT) < 31)
    (DEST) ← (DEST) * 2
    (COUNT) ← (COUNT) + 1
end_while
```

Assembly Language Format: NORML lreg,breg

Object Code Format: [00001111] [breg] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	?	0	—	—	—

74. NOT — COMPLEMENT WORD

Operation: The value of the WORD operand is complemented: each 1 is replaced with a 0, and each 0 with a 1.

```
(DEST) ← NOT (DEST)
```

Assembly Language Format: NOT wreg

Object Code Format: [00000010] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

75. NOTB — COMPLEMENT BYTE

Operation: The vaule of the BYTE operand is complemented: each 1 is replaced with a 0, and each 0 with a 1.

(DEST) ← NOT (DEST)

Assembly Language Format: NOTB breg

Object Code Format: [00010010] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

76. OR — LOGICAL OR WORDS

Operation: The source (rightmost) WORD is ORed with the destination (leftmost) WORD operand. Each bit is set to 1 if the corresponding bit in either the source operand or the destination operand is 1. The result replaces the original destination operand.

(DEST) ← (DEST) OR (SRC)

Assembly Language Format:

```
        DST     SRC
OR      wreg,   waop
```

Object Code Format: [100000aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

77. ORB — LOGICAL OR BYTES

Operation: The source (rightmost) BYTE operand is ORed with the destination (leftmost) BYTE operand. Each bit is set to 1 if the corresponding bit in either the source operand or the destination operand was 1. The result replaces the original destination operand.

(DEST) ← (DEST) OR (SRC)

Assembly Language Format: ORB breg,baop

Object Code Format: [100100aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

78. POP — POP WORD

Operation: The word on top of the stack is popped and placed at the destination operand.

(DEST) ← (SP)
SP ← SP + 2

Assembly Language Format: POP waop

Object Code Format: [110011aa] [waop]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

79. POPA — POP ALL (80C196KB and 80C196KC only)

Operation: This instruction is used instead of POPF to support the 8 additional interrupts. It is similar to POPF, but pops two words instead of one. The first word is popped into the INT_MASK1/WSR register pair, while the second word is popped into the PSW/INT_MASK register pair. As a result of this instruction the SP is incremented by 4. Interrupts cannot occur between this instruction and the one following it.

INT_MASK1/WSR ← (SP)
SP ← SP + 2
PSW/INT_MASK ← (SP)
SP ← SP + 2

Assembly Language Format: POPA

Object Code Format: [11110101]

Flags Affected							
Z	N	V	VT	C	X	I	ST
✓	✓	✓	✓	✓	X	✓	✓

(✓ = changed)

80. POPF — POP FLAGS

Operation: The word on top of the stack is popped and placed in the PSW. Interrupt calls cannot occur immediately following this instruction.

(PSW) ← (SP)
SP ← SP + 2

Assembly Language Format: POPF

Object Code Format: [11110011]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	✓	✓

81. PUSH — PUSH WORD

Operation: The specified operand is pushed onto the stack.

SP ← SP − 2
(SP) ← (DEST)

Assembly Language Format: PUSH waop

Object Code Format: [110010aa] [waop]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

82. PUSHA — PUSH ALL (80C196KB and 80C196KC only)

Operation: This instruction is used instead of PUSHF to support the 8 additional interrupts. It is similar to PUSHF, but pushes two words instead of one. The first word pushed is the same as for the PUSHF instruction, PSW/INT_MASK. The second word pushed is formed by the INT_MASK1/WSR register pair. As a result of this instruction the PSW, INT_MASK, and INT_MASK1 registers are cleared, and the SP is decremented by 4. Interrupts are disabled in two ways by this instruction since both PSW.9 and the interrupt masks are cleared. Interrupts cannot occur between this instruction and the one following it.

SP ← SP − 2
(SP) ← PSW/INT_MASK
PSW/INT_MASK ← 0
SP ← SP − 2
(SP) ← INT_MASK1/WSR
INT_MASK1 ← 0

Assembly Language Format: PUSHA

Object Code Format: [11110100]

Flags Affected							
Z	N	V	VT	C	X	I	ST
0	0	0	0	0	X	0	0

83. PUSHF — PUSH FLAGS

Operation: The PSW is pushed on top of the stack, and then set to all zeroes. This implies that all interrupts are disabled. Interrupt-calls cannot occur immediately following this instruction.

SP ← SP − 2
(SP) ← PSW
PSW ← 0

Assembly Language Format: PUSHF

Object Code Format: [11110010]

Flags Affected					
Z	N	C	V	VT	ST
0	0	0	0	0	0

84. RET — RETURN FROM SUBROUTINE

Operation: The PC is popped off the top of the stack.

PC ← (SP)
SP ← SP + 2

Assembly Language Format: RET

Object Code Format: [11110000]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

85. RST — RESET SYSTEM

Operation: The PSW is initialized to zero, and the PC is initialized to 2080H. The I/O registers are set to their initial value. Executing this instruction will cause a pulse to appear on the reset pin.

PSW ← 0
PC ← 2080H

Assembly Language Format: RST

Object Code Format: [11111111]

Flags Affected					
Z	N	C	V	VT	ST
0	0	0	0	0	0

86. SCALL — SHORT CALL

Operation: The contents of the program counter (the return address) is pushed onto the stack. Then the distance from the end of this instruction to the target label is added to the program counter, effecting the call. The offset from the end of this instruction to the target label must be in the range of −1024 to +1023 inclusive.

SP ← SP − 2
(SP) ← PC
PC ← PC + disp (sign-extended to 16 bits)

Assembly Language Format: SCALL cadd

Object Code Format: [00101xxx] [disp-low]
where xxx holds the three high-order bits of displacement.

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

87. SETC — SET CARRY FLAG

Operation: The carry flag is set.

```
C ← 1
```

Assembly Language Format: SETC

Object Code Format: [11111001]

Flags Affected					
Z	N	C	V	VT	ST
—	—	1	—	—	—

88. SHL — SHIFT WORD LEFT

Operation: The destination (leftmost) word operand is shifted left as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. The right bits of the result are filled with zeroes. The last bit shifted out is saved in the carry flag.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← High order bit of (DEST)
    (DEST) ← (DEST) * 2
    Temp ← Temp — 1
end_while
```

Assembly Language Format:

```
          SHL     wreg,#count
or
          SHL     wreg,breg
```

Object Code Format: [00001001] [cnt/breg] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	?	✓	✓	↑	—

89. SHLB—SHIFT BYTE LEFT

Operation: The destination (leftmost) byte operand is shifted left as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. The right bits of the result are filled with zeroes. The last bit shifted out is saved in the carry flag.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← High order bit of (DEST)
    (DEST) ← (DEST) * 2
    TEMP ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHLB    breg,#count
or
        SHLB    breg,breg
```

Object Code Format: [00011001] [cnt/breg] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	?	✓	✓	↑	—

90. SHLL — SHIFT DOUBLE-WORD LEFT

Operation: The destination (leftmost) double-word operand is shifted left as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. The right bits of the result are filled with zeroes. The last bit shifted out is saved in the carry flag.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← High order bit of (DEST)
    (DEST) ← (DEST) * 2
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHLL    lreg,#count
or
        SHLL    lreg, breg
```

Object Code Format: [00001101] [cnt/breg] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	?	✓	✓	↑	—

91. SHR — LOGICAL RIGHT SHIFT WORD

Operation: The destination (leftmost) word operand is shifted right as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. The left bits of the result are filled with zeroes. The last bit shifted out is saved to the carry. The sticky bit flag is cleared at the beginning of the instruction, and set if at any time during the shift a 1 is shifted first into the carry flag, and a further shift cycle occurs.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← Low order bit of (DEST)
    (DEST) ← (DEST) / 2 where / is unsigned division
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHR     wreg,#count
or
        SHR     wreg,breg
```

Object Code Format: [00001000] [cnt/breg] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	0	✓	0	—	✓

92. SHRA — ARITHMETIC RIGHT SHIFT WORD

Operation: The destination (leftmost) word operand is shifted right as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. If the original high order bit value was 0, zeroes are shifted in. If the value was 1, ones are shifted in. The last bit shifted out is saved in the carry. The sticky bit flag is cleared at the beginning of the instruction, and set if at any time during the shift a 1 is shifted first into the carry flag, and a further shift cycle occurs.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← Low order bit of (DEST)
    (DEST) ← (DEST) / 2 where / is signed division
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHRA    wreg,#count
or
        SHRA    wreg,breg
```

Object Code Format: [00001010] [cnt/breg] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	0	—	✓

93. SHRAB — ARITHMETIC RIGHT SHIFT BYTE

Operation: The destination (leftmost) byte operand is shifted right as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. If the original high order bit value was 0, zeroes are shifted in. If that value was 1, ones are shifted in. The last bit shifted out is saved in the carry. The sticky bit flag is cleared at the beginning of the instruction, and set if at any time during the shift a 1 is shifted first into the carry flag, and a further shift cycle occurs.

```
Temp ← (COUNT)
do while Temp <> 0
    C, = Low order bit of (DEST)
    (DEST) ← (DEST) / 2 where / is signed division
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHRAB   breg,#count
or
        SHRAB   breg,breg
```

Object Code Format: [00011010] [cnt/breg] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	0	—	✓

94. SHRAL — ARITHMETIC RIGHT SHIFT DOUBLE-WORD

Operation: The destination (leftmost) double-word operand is shifted right as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. If the original high order bit value was 0, zeroes are shifted in. If the value was 1, ones are shifted in. The sticky bit is cleared at the beginning of the instruction, and set if at any time during the shift a 1 is shifted first into the carry flag, and a further shift cycle occurs.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← Low order bit of (DEST)
    (DEST) ← (DEST) / 2 where / is signed division
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHRAL     lreg,#count
or
        SHRAL     lreg,breg
```

Object Code Format: [00001110] [cnt/breg] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	0	—	✓

95. SHRB — LOGICAL RIGHT SHIFT BYTE

Operation: The destination (leftmost) byte operand is shifted right as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. The left bits of the result are filled with zeroes. The last bit shifted out is saved in the carry. The sticky bit flag is cleared at the beginning of the instruction, and set if at any time during the shift a 1 is shifted first into the carry flag, and a further shift cycle occurs.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← Low order bit of (DEST)
    (DEST) ← (DEST) / 2 where / is unsigned division
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHRB    breg,#count
or
        SHRB    breg,breg
```

Object Code Format: [00011000] [cnt/breg] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	0	✓	0	—	✓

96. SHRL — LOGICAL RIGHT SHIFT DOUBLE-WORD

Operation: The destination (leftmost) double-word operand is shifted right as many times as specified by the count (rightmost) operand. The count may be specified either as an immediate value in the range of 0 to 15 (0FH) inclusive, or as the content of any register. Details on indirect shifts can be found in the Overview. The left bits of the result are filled with zeroes. The last bit shifted out is saved in the carry. The sticky bit flag is cleared at the beginning of the instruction, and set if at any time during the shift a 1 is shifted first into the carry flag, and a further shift cycle occurs.

```
Temp ← (COUNT)
do while Temp <> 0
    C ← Low order bit of (DEST)
    (DSET) ← (DEST) / 2 where / is unsigned division
    Temp ← Temp − 1
end_while
```

Assembly Language Format:

```
        SHRL    lreg,#count
or
        SHRL    lreg,breg
```

Object Code Format: [00001100] [cnt/breg] [lreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	0	✓	0	—	✓

97. SJMP — SHORT JUMP

Operation: The distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The offset from the end of this instruction to the label must be in the range of −1024 to +1023 inclusive.

PC ← PC + disp (sign-extended to 16 bits)

Assembly Language Format: SJMP cadd

Object Code Format: [00100xxx] [disp-low]
where xxx holds the three high order bits of the displacement.

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

98. SKIP — TWO BYTE NO-OPERATION

Operation: Nothing is done. This is actually a two-byte NOP where the second byte can be any value, and is simply ignored. Control passes to the next sequential instruction.

Assembly Language Format: SKIP breg

Object Code Format: [00000000] [breg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

99. ST — STORE WORD

Operation: The value of the leftmost word operand is stored into the rightmost operand.
(DEST) ← (SRC)

Assembly Language Format:

```
        SRC     DST
ST      wreg,   waop
```

Object Code Format: [110000aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

100. STB — STORE BYTE

Operation: The value of the leftmost byte operand is stored into the rightmost operand.
(DEST) ← (SRC)

Assembly Language Format:

```
        SRC     DST
STB     breg,   baop
```

Object Code Format: [110001aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

101. SUB (Two Operands) — SUBTRACT WORDS

Operation: The source (rightmost) word operand is subtracted from the destination (leftmost) word operand, and the result is stored in the destination. The carry flag is set as complement of borrow.

(DEST) ← (DEST) — (SRC)

Assembly Language Format:

	DST	SRC
SUB	wreg,	waop

Object Code Format: [011010aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

102. SUB (Three Operands) — SUBTRACT WORDS

Operation: The source (rightmost) word operand is subtracted from the second word operand, and the result is stored in the destination (the leftmost operand). The carry flag is set as complement of borrow.

(DEST) ← (SRC1) — (SRC2)

Assembly Language Format:

	DST	SRC1	SRC2,
SUB	wreg,	wreg,	waop

Object Code Format: [010010aa] [waop] [Sweg] [Dwreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

103. SUBB (Two Operands) — SUBTRACT BYTES

Operation: The source (rightmost) byte is subtracted from the destination (leftmost) byte operand, and the result is stored in the destination. The carry flag is set as complement of borrow.

(DEST) ← (DEST) — (SRC)

Assembly Language Format:

```
        DST     SRC
SUBB    breg,   baop
```

Object Code Format: [011110aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

104. SUBB (Three Operands) — SUBTRACT BYTES

Operation: The source (rightmost) byte operand is subtracted from the second byte operand, and the result is stored in the destination (the leftmost operand). The carry flag is set as complement of borrow.

(DEST) ← (SRC1) — (SRC2)

Assembly Language Format:

```
        DST     SRC1    SRC2
SUBB    breg,   Sbreg   baop
```

Object Code Format: [010110aa] [baop] [Sbreg] [Dbreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	✓	✓	↑	—

105. SUBC — SUBTRACT WORDS WITH BORROW

Operation: The source (rightmost) word operand is subtracted from the destination (leftmost) word operand. If the carry flag was clear, 1 is subtracted from the above result. The result replaces the orignal destination operand. The carry flag is set as complement of borrow.

(DEST) ← (DEST) — (SRC) — (1-C)

Assembly Language Format:

```
        DST     SRC
SUBC    wreg,   waop
```

Object Code Format: [101010aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
↓	✓	✓	✓	↑	—

106. SUBCB — SUBTRACT BYTES WITH BORROW

Operation: The source (rightmost) byte operand is subtracted from the destination (leftmost) byte operand. If the carry flag was clear, 1 is subtracted from the above result. The result replaces the original destination operand. The carry flag is set as complement of borrow.

(DEST) ← (DEST) — (SRC) — (1-C)

Assembly Language Format:

```
        DST     SRC
SUBCB   breg,   baop
```

Object Code Format: [101110aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
↓	✓	✓	✓	↑	—

107. TIJMP — TABLE INDIRECT JUMP (80C196KC only)

Operation: The execution continues at an address selected out of a table of addresses. TBASE is a word register which contains the 16-bit address of the beginning of the table. INDEX is a word register containing the 16-bit address of a byte which contains the index into the table. INDEX_MASK is ANDed with the index. The index must be between 0 and 128.

ADDRESS CALCULATION

```
[INDEX] AND INDEX_MASK = OFFSET
          (2 * OFFSET) + [TBASE] = DEST X
```

Assembly Language Format:

```
      TBASE [INDEX] #INDEX_MASK
TIJMP wreg,   wreg    #byte
```

Object Code Format: [11100010] [wreg] [#byte] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

108. TRAP — SOFTWARE TRAP

Operation: This instruction causes an interrupt-call which is vectored through location 2010H. The operation of this instruction is not effected by the state of the interrupt enable flag in the PSW (I). Interrupt-calls cannot occur immediately following this instruction. This instruction is intended for use by Intel provided development tools. These tools will not support user-application of this instruction.

SP ← SP − 2
(SP) ← PC
PC ← (2010H)

Assembly Language Format: This instruction is not supported by revision 1.2 of the 8096 assembly language.

Object Code Format: [11110111]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

109. XOR — LOGICAL EXCLUSIVE-OR WORDS

Operation: The source (rightmost) word operand is XORed with the destination (leftmost) word operand. Each bit is set to 1 if the corresponding bit in either the source operand or the destination operand was 1, but not both. The result replaces the original destination operand.

(DEST) ← (DEST) XOR (SRC)

Assembly Language Format:

```
       DST    SRC
XOR    wreg,  waop
```

Object Code Format: [100001aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

110. XCH — EXCHANGE WORD (80C196KC only)

Operation: The value of the source (rightmost) word operand is exchanged with the destination (leftmost) operand.

(DEST) ← (SRC)

Assembly Language Format:

	DST	SRC
XCH	wreg,	waop

Object Code Format: [000001aa] [waop] [wreg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

111. XCHB — EXCHANGE BYTE (80C196KC only)

Operation: The value of the source (rightmost) byte operand is exchanged with the destination (leftmost) operand.

(DEST) ← (SRC)

Assembly Language Format:

	DST	SRC
XCHB	breg,	baop

Object Code Format: [000101aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
—	—	—	—	—	—

112. XORB — LOGICAL EXCLUSIVE-OR BYTES

Operation: The source (rightmost) byte operand is XORed with the destination (leftmost) byte operand. Each bit is set to 1 if the corresponding bit in either the source operand or the destination operand was 1, but not both. The result replaces the original destination operand.

(DEST) ← (DEST) XOR (SRC)

Assembly Language Format:

```
        DST     SRC
XORB    breg,   baop
```

Object Code Format: [100101aa] [baop] [breg]

Flags Affected					
Z	N	C	V	VT	ST
✓	✓	0	0	—	—

80C196KB User's Guide and Data Sheets

4

The 80C196KB family is a CHMOS branch of the MCS®-96 family. Other members of the MCS-96 family include the 8096BH and 8098. All of the MCS-96 components share a common instruction set and architecture. However the CHMOS components have enhancements to provide higher performance at lower power consumptions. To further decrease power usage, these parts can be placed into idle and powerdown modes.

MCS-96 family members are all high-performance microcontrollers with a 16-bit CPU and at least 230 bytes of on-chip RAM. They are register-to-register machines, so no accumulator is needed, and most operations can be quickly performed from or to any of the registers. In addition, the register operations can control the many peripherals which are available on the chips. These peripherals include a serial port, A/D converter, PWM output, up to 48 I/O lines and a High-Speed I/O subsystem which has 2 16-bit timer/counters, an 8-level input capture FIFO and an 8-entry programmable output generator.

Typical applications for MCS-96 products are closed-loop control and mid-range digital signal processing. MCS-96 products are being used in modems, motor controls, printers, engine controls, photocopiers, antilock brakes, air conditioner temperature controls, disk drives, and medical instrumentation.

There are many members of the 80C196KB family, so to provide easier reading this manual will refer to the 80C196KB family generically as the 80C196KB. Where information applies only to specific components it will be clearly indicated.

The 80C196KB can be separated into four sections for the purpose of describing its operation. A block diagram is shown in Figure 1-1. There is the CPU and architecture, the instruction set, the peripherals and the bus unit. Each of the sections will be sub-divided as the discussion progresses. Let us first examine the CPU.

1.0 CPU OPERATION

The major components of the CPU on the 80C196KB are the Register File and the Register/Arithmetic Logic Unit (RALU). Communication with the outside world is done through either the Special Function Registers (SFRs) or the Memory Controller. The RALU does not use an accumulator. Instead, it operates directly on the 256-byte register space made up of the Register File and the SFRs. Efficient I/O operations are possible by directly controlling the I/O through the SFRs. The main benefits of this structure are the ability to quickly change context, absence of accumulator bottleneck, and fast throughput and I/O times.

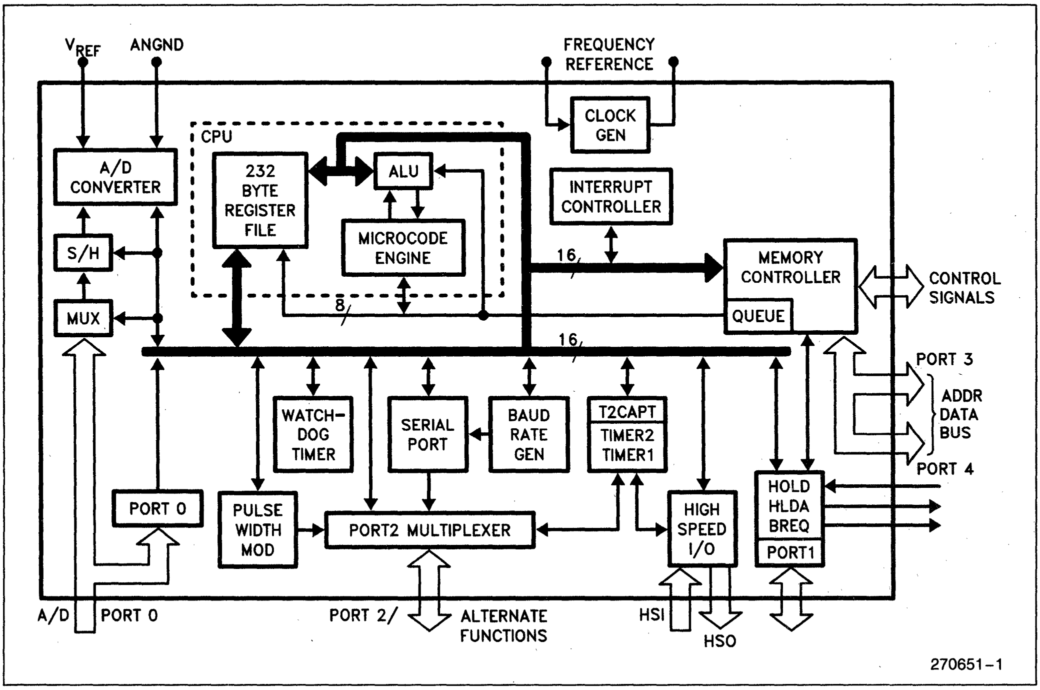


Figure 1-1. 80C196KB Block Diagram

The CPU on the 80C196KB is 16 bits wide and connects to the interrupt controller and the memory controller by a 16-bit bus. In addition, there is an 8-bit bus which transfers instruction bytes from the memory controller to the CPU. An extension of the 16-bit bus connects the CPU to the peripheral devices.

1.1 Memory Controller

The RALU talks to the memory, except for the locations in the register file and SFR space, through the memory controller. Within the memory controller is a bus controller, a four byte queue and a Slave Program Counter (Slave PC). Both the internal ROM/EPROM bus and the external memory bus are driven by the bus controller. Memory access requests to the bus controller can come from either the RALU or the queue, with queue accesses having priority. Requests from the queue are always for data at the address in the slave PC.

By having program fetches from memory referenced to the slave PC, the processor saves time as addresses seldom have to be sent to the memory controller. If the address sequence changes because of a jump, interrupt, call or return, the slave PC is loaded with a new value, the queue is flushed, and processing continues.

Execution speed is increased by using a queue since it usually keeps the next instruction byte available. The instruction execution times shown in Section 3 show the normal execution times with no wait states added and the 16-bit bus selected. Reloading the slave PC and fetching the first byte of the new instruction stream takes 4 state times. This is reflected in the jump taken/not-taken times shown in the table.

When debugging code using a logic analyzer, one must be aware of the queue. It is not possible to determine when an instruction will begin executing by simply watching when it is fetched, since the queue is filled in advance of instruction execution.

1.2 CPU Control

A microcode engine controls the CPU, allowing it to perform operations with any byte, word or double word in the 256 byte register space. Instructions to the CPU are taken from the queue and stored temporarily in the instruction register. The microcode engine decodes the instructions and generates the correct sequence of events to have the RALU perform the desired function. Figure 1-2 shows the memory controller, RALU, instruction register and the control unit.

REGISTER/ALU (RALU)

Most calculations performed by the 80C196KB take place in the RALU. The RALU, shown in Figure 1-2, contains a 17-bit ALU, the Program Status Word (PSW), the Program Counter (PC), a loop counter, and three temporary registers. All of the registers are 16-bits or 17-bits (16+ sign extension) wide. Some of the registers have the ability to perform simple operations to off-load the ALU.

A separate incrementor is used for the Program Counter (PC) as it accesses operands. However, PC changes due to jumps, calls, returns and interrupts must be handled through the ALU. Two of the temporary registers have their own shift logic. These registers are used for the operations which require logical shifts, including Normalize, Multiply, and Divide. The "Lower Word" and "Upper Word" are used together for the 32-bit instructions and as temporary registers for many instructions. Repetitive shifts are counted by the 6-bit "Loop Counter".

A third temporary register stores the second operand of two operand instructions. This includes the multiplier during multiplications and the divisor during divisions. To perform subtractions, the output of this register can be complemented before being placed into the "B" input of the ALU.

Several constants, such as 0, 1 and 2 are stored in the RALU to speed up certain calculations. (e.g. making a 2's complement number or performing an increment or decrement instruction.) In addition, single bit masks for bit test instructions are generated in the constant register based on the 3-bit Bit Select register.

1.3 Internal Timing

The 80C196KB requires an input clock on XTAL1 to function. Since XTAL1 and XTAL2 are the input and output of an inverter a crystal can be used to generate the clock. Details of the circuit and suggestions for its use can be found in Section 13.

Internal operation of the 80C196KB is based on the crystal or external oscillator frequency divided by 2. Every 2 oscillator periods is referred to as one "state time", the basic time measurement for all 80C196KB operations. With a 12 MHz oscillator, a state time is 167 nanoseconds. With an 8 MHz oscillator, a state time is 250 nanoseconds, the same as an 8096BH running with a 12 MHz oscillator. Since the 80C196KB will be run at many frequencies, the times given throughout this chapter will be in state times or "states", unless otherwise specified. A clock out

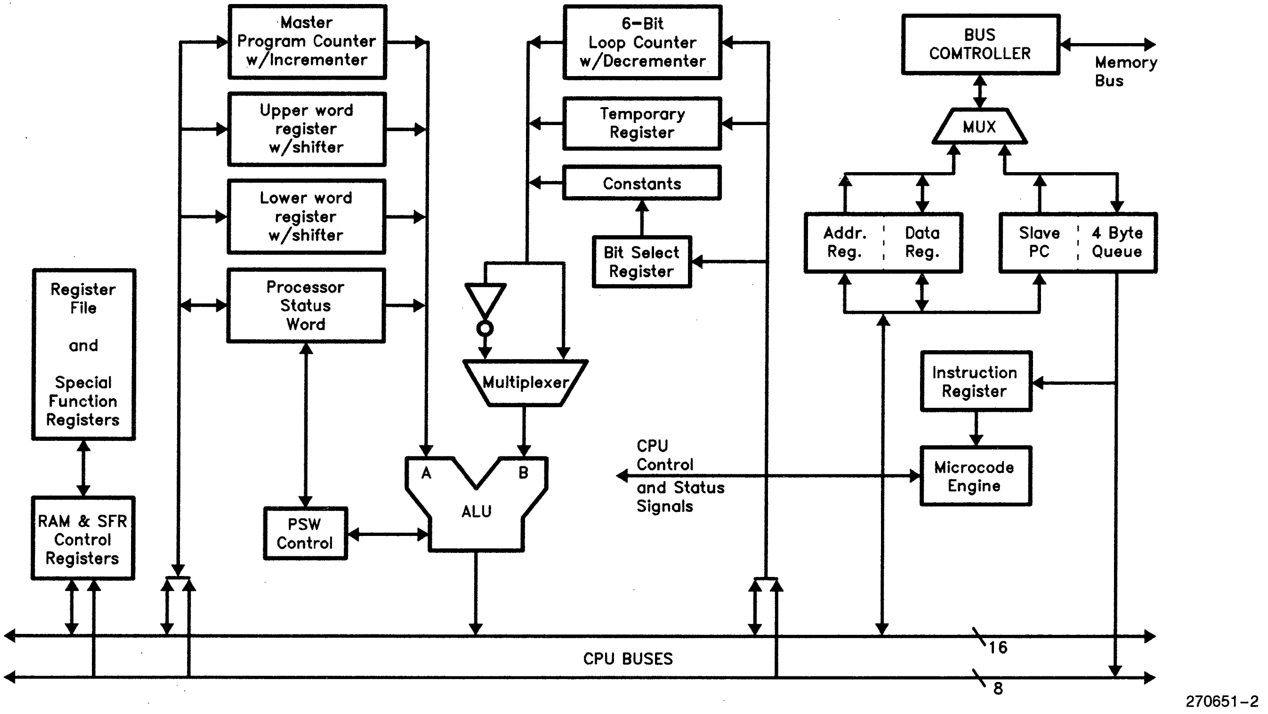


Figure 1-2. RALU and Memory Controller Block Diagram

(CLKOUT) signal, shown in Figure 1-3, is provided as an indication of the internal machine state. Details on timing relationships can be found in Section 13.

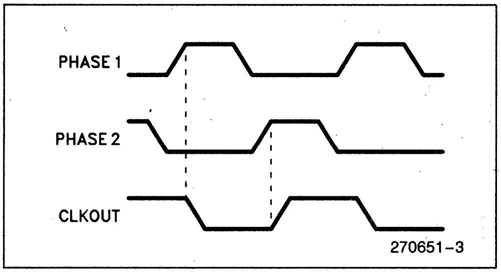


Figure 1-3. Internal Clock Waveforms

2.0 MEMORY SPACE

The addressable memory space on the 80C196KB consists of 64K bytes, most of which is available to the user for program or data memory. Locations which have special purposes are 0000H through 00FFH and 1FFEH through 2080H. All other locations can be used for either program or data storage or for memory mapped peripherals. A memory map is shown in Figure 2-1.

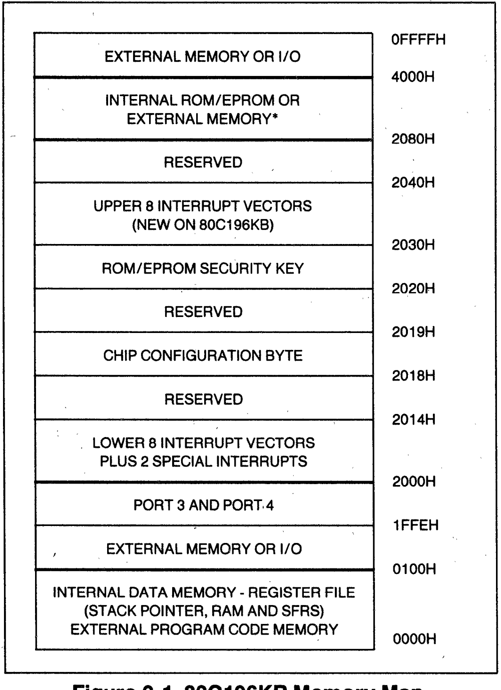

0FFFFH
EXTERNAL MEMORY OR I/O
4000H
INTERNAL ROM/EPROM OR EXTERNAL MEMORY*
2080H
RESERVED
2040H
UPPER 8 INTERRUPT VECTORS (NEW ON 80C196KB)
2030H
ROM/EPROM SECURITY KEY
2020H
RESERVED
2019H
CHIP CONFIGURATION BYTE
2018H
RESERVED
2014H
LOWER 8 INTERRUPT VECTORS PLUS 2 SPECIAL INTERRUPTS
2000H
PORT 3 AND PORT 4
1FFEH
EXTERNAL MEMORY OR I/O
0100H
INTERNAL DATA MEMORY - REGISTER FILE (STACK POINTER, RAM AND SFRS) EXTERNAL PROGRAM CODE MEMORY
0000H

Figure 2-1. 80C196KB Memory Map

2.1 Register File

Locations 00H through 0FFH contain the Register File and Special Function Registers, (SFRs). The RALU can operate on any of these 256 internal register locations, but code can not be executed from them. If an attempt to execute instructions from locations 000H through 0FFH is made, the instructions will be fetched from *external* memory. This section of external memory is reserved for use by Intel development tools

The internal RAM from location 018H (24 decimal) to 0FFH is the Register File. It contains 232 bytes of RAM which can be accessed as bytes (8 bits), words (16 bits), or double-words (32 bits). Since each of these locations can be used by the RALU, there are essentially 232 "accumulators". This memory region, as well as the status of the majority of the chip, is kept intact while the chip is in the Powerdown Mode. Details on Powerdown Mode are discussed in Section 14.

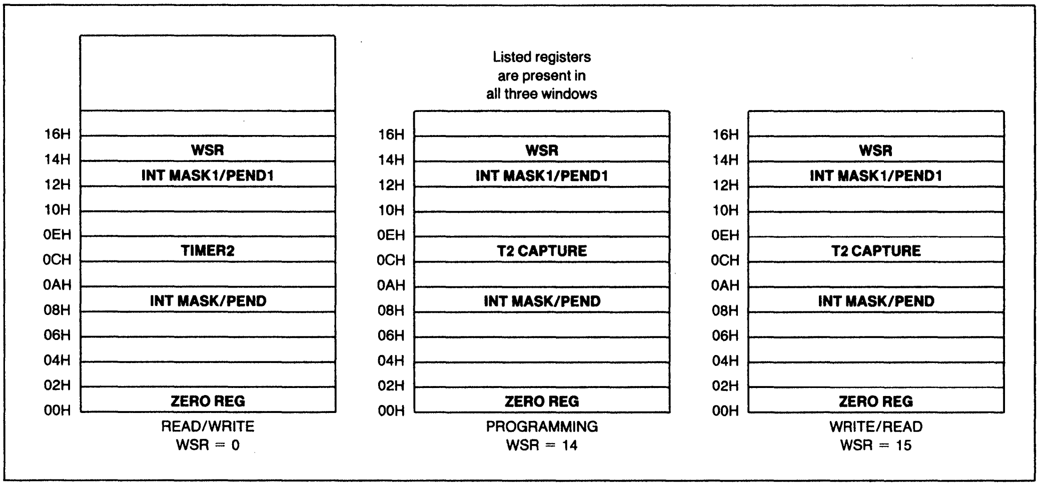
Locations 18H and 19H contain the stack pointer. These are not SFRs and may be used as standard RAM if stack operations are not being performed. Since the stack pointer is in this area, the RALU can easily operate on it. The stack pointer must be initialized by the user program and can point anywhere in the 64K memory space. Operations to the stack cause it to build down, so the stack pointer should be initialized to 2 bytes above the highest stack location, and must be word aligned.

2.2 Special Function Registers

Locations 00H through 17H are the I/O control registers or SFRs. All of the peripheral devices on the 80C196KB (except Ports 3 and 4) are controlled through these registers. As shown in Figure 2-2, three SFR windows are provided on the 80C196KB.

Switching between the windows is done using the Window Select Register (WSR) at location 14H in all of the windows. The PUSHA and POPA instructions push and pop the WSR so it is easy to change between windows. Only three values may be written to the WSR, 0, 14 and 15. Other values are reserved for use in future parts and will cause unpredictable operation.

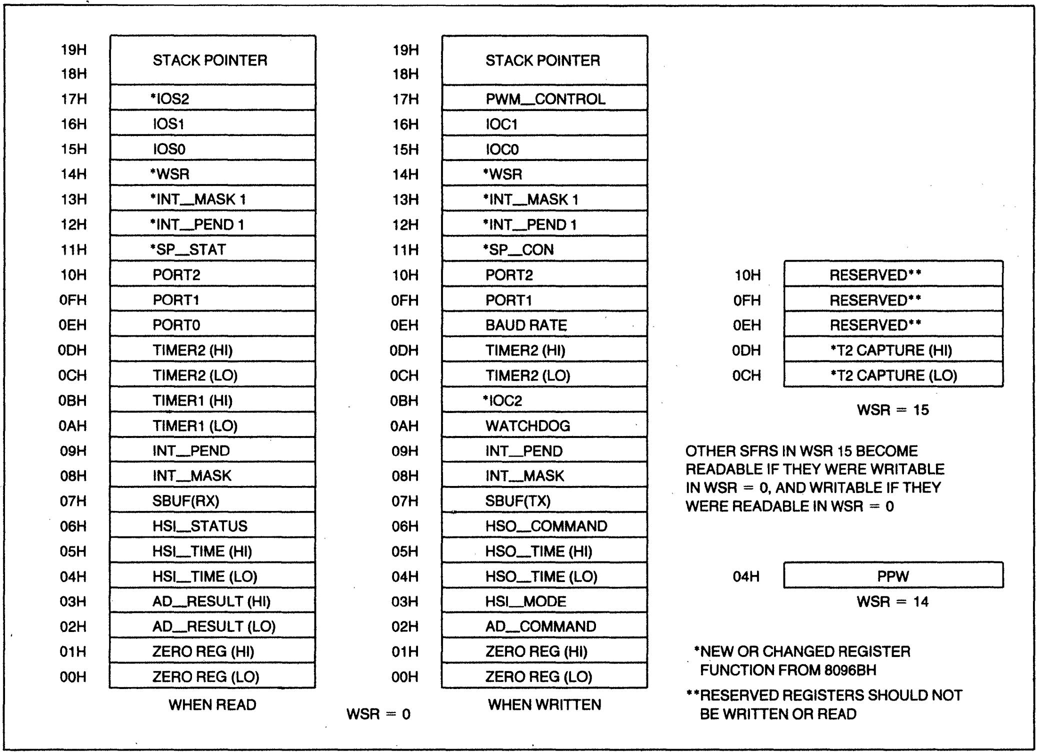
Window 0, the register window selected with WSR = 0, is a superset of the one used on the 8096BH. As depicted in Figure 2-3, it has 24 registers, some of which have different functions when read than when written. Registers which are new to the 80C196KB or have changed functions from the 8096 are indicated in the figure.



Listed registers are present in all three windows

	READ/WRITE WSR = 0	PROGRAMMING WSR = 14	WRITE/READ WSR = 15
16H			
14H	WSR	WSR	WSR
12H	INT MASK1/PEND1	INT MASK1/PEND1	INT MASK1/PEND1
10H			
0EH			
0CH	TIMER2	T2 CAPTURE	T2 CAPTURE
0AH			
08H	INT MASK/PEND	INT MASK/PEND	INT MASK/PEND
06H			
04H			
02H			
00H	ZERO REG	ZERO REG	ZERO REG

Figure 2-2. Multiple Register Windows



WSR = 0

	WHEN READ	WHEN WRITTEN
19H 18H	STACK POINTER	STACK POINTER
17H	*IOS2	PWM_CONTROL
16H	IOS1	IOC1
15H	IOS0	IOC0
14H	*WSR	*WSR
13H	*INT_MASK 1	*INT_MASK 1
12H	*INT_PEND 1	*INT_PEND 1
11H	*SP_STAT	*SP_CON
10H	PORT2	PORT2
0FH	PORT1	PORT1
0EH	PORT0	BAUD RATE
0DH	TIMER2 (HI)	TIMER2 (HI)
0CH	TIMER2 (LO)	TIMER2 (LO)
0BH	TIMER1 (HI)	*IOC2
0AH	TIMER1 (LO)	WATCHDOG
09H	INT_PEND	INT_PEND
08H	INT_MASK	INT_MASK
07H	SBUF(RX)	SBUF(TX)
06H	HSI_STATUS	HSO_COMMAND
05H	HSI_TIME (HI)	HSO_TIME (HI)
04H	HSI_TIME (LO)	HSO_TIME (LO)
03H	AD_RESULT (HI)	HSI_MODE
02H	AD_RESULT (LO)	AD_COMMAND
01H	ZERO REG (HI)	ZERO REG (HI)
00H	ZERO REG (LO)	ZERO REG (LO)

WSR = 15

10H	RESERVED**
0FH	RESERVED**
0EH	RESERVED**
0DH	*T2 CAPTURE (HI)
0CH	*T2 CAPTURE (LO)

OTHER SFRS IN WSR 15 BECOME READABLE IF THEY WERE WRITABLE IN WSR = 0, AND WRITABLE IF THEY WERE READABLE IN WSR = 0

WSR = 14

04H	PPW

*NEW OR CHANGED REGISTER FUNCTION FROM 8096BH

**RESERVED REGISTERS SHOULD NOT BE WRITTEN OR READ

Figure 2-3. Special Function Registers

Register	Description
R0	Zero Register - Always reads as a zero, useful for a base when indexing and as a constant for calculations and compares.
AD_RESULT	A/D Result Hi/Low - Low and high order results of the A/D converter
AD_COMMAND	A/D Command Register - Controls the A/D
HSI_MODE	HSI Mode Register - Sets the mode of the High Speed Input unit.
HSI_TIME	HSI Time Hi/Lo - Contains the time at which the High Speed Input unit was triggered.
HSO_TIME	HSO Time Hi/Lo - Sets the time or count for the High Speed Output to execute the command in the Command Register.
HSO_COMMAND	HSO Command Register - Determines what will happen at the time loaded into the HSO Time registers.
HSI_STATUS	HSI Status Registers - Indicates which HSI pins were detected at the time in the HSI Time registers and the current state of the pins. In Window 15 - Writes to pin detected bits, but not current state bits.
SBUF(TX)	Transmit buffer for the serial port, holds contents to be outputted. Last written value is readable in Window 15.
SBUF(RX)	Receive buffer for the serial port, holds the byte just received by the serial port. Writable in Window 15.
INT_MASK	Interrupt Mask Register - Enables or disables the individual interrupts.
INT_PEND	Interrupt Pending Register - Indicates that an interrupt signal has occurred on one of the sources and has not been serviced. (also INT_PENDING)
WATCHDOG	Watchdog Timer Register - Written periodically to hold off automatic reset every 64K state times. Returns upper byte of WDT counter in Window 15.
TIMER1	Timer 1 Hi/Lo - Timer1 high and low bytes.
TIMER2	Timer 2 Hi/Lo - Timer2 high and low bytes.
IOPORT0	Port 0 Register - Levels on pins of Port 0. Reserved in Window 15.
BAUD_RATE	Register which determines the baud rate, this register is loaded sequentially. Reserved in Window 15.
IOPORT1	Port 1 Register - Used to read or write to Port 1. Reserved in Window 15
IOPORT2	Port 2 Register - Used to read or write to Port 2. Reserved in Window 15
SP_STAT	Serial Port Status - Indicates the status of the serial port.
SP_CON	Serial Port Control - Used to set the mode of the serial port.
IOS0	I/O Status Register 0 - Contains information on the HSO status. Writes to HSO pins in Window 15.
IOS1	I/O Status Register 1 - Contains information on the status of the timers and of the HSI.
IOC0	I/O Control Register 0 - Controls alternate functions of HSI pins, Timer 2 reset sources and Timer 2 clock sources.
IOC1	I/O Control Register 1 - Controls alternate functions of Port 2 pins, timer interrupts and HSI interrupts.
PWM_CONTROL	Pulse Width Modulation Control Register - Sets the duration of the PWM pulse.
INT_PEND1	Interrupt Pending register for the 8 new interrupt vectors (also INT_PENDING1)
INT_MASK1	Interrupt Mask register for the 8 new interrupt vectors
IOC2	I/O Control Register 2 - Controls new 80C196KB features
IOS2	I/O Status Register 2 - Contains information on HSO events
WSR	Window Select Register - Selects register window

Figure 2-4. Special Function Register Description

Programming control and test operations are done in Window 14. Registers in this window that are not labeled should be considered reserved and should not be either read or written.

In register Window 15 (WSR = 15), the operation of the SFRs is changed, so that those which were read-only in Window 0 space are write-only and vice versa. The only major exception to this is that Timer2 is read/write in Window 0, and T2 Capture is read/write in Window 15. (Timer2 was read-only on the 8096.) Registers which can be read and written in Window 0 can also be read and written in Window 15.

Figure 2-4 contains brief descriptions of the SFR registers. Detailed descriptions are contained in the section which discusses the peripheral controlled by the register. Figure 2-5 contains a description of the alternate function in Window 15.

AD_COMMAND (02H)	— Read the last written command
AD_RESULT (02H, 03H)	— Write a value into the result register
HSI_MODE (03H)	— Read the value in HSI_MODE
HSI_TIME (04H, 05H)	— Write to FIFO Holding register
HSO_TIME (04H, 05H)	— Read the last value placed in the holding register
HSI_STATUS (06H)	— Write to status bits but not to HSI pin bits. (Pin bits are 1, 3, 5, 7)
HSO_COMMAND (06H)	— Read the last value placed in the holding register
SBUF(RX) (07H)	— Write a value into the receive buffer
SBUF(TX) (07H)	— Read the last value written to the transmit buffer
WATCHDOG (0AH)	— Read the value in the upper byte of the WDT
TIMER1 (0AH, 0BH)	— Write a value to Timer1
TIMER2 (0CH, 0DH)	— Read/Write the Timer2 capture register. (Timer2 read/write is done with WSR = 0)
IOC2 (0BH)	— Last written value is readable, except bit 7 (Note 1)
BAUD_RATE (0EH)	— No function, cannot be read
PORT0 (0EH)	— No function, no output drivers on the pins
SP_STAT (11H)	— Set the status bits, TI and RI can be set, but it will not cause an interrupt
SP_CON (11H)	— Read the current control byte
IOS0 (15H)	— Writing to this register controls the HSO pins. Bits 6 and 7 are inactive for writes.
IOC0 (15H)	— Last written value is readable, except bit 1 (Note 1)
IOS1 (16H)	— Writing to this register will set the status bits, but not cause interrupts. Bits 6 and 7 are not functional.
IOC1 (16H)	— Last written value is readable
IOS2 (17H)	— Writing to this register will set the status bits, but not cause interrupts.
PWM_CONTROL (17H)	— Read the duty cycle value written to PWM_CONTROL

NOTE:
1. IOC2.7 (CAM CLEAR) and IOC0.1 (T2RST) are not latched and will read as a 1 (precharged bus).

Being able to write to the read-only registers and vice-versa provides a lot of flexibility. One of the most useful advantages is the ability to set the timers and HSO lines for initial conditions other than zero.

Figure 2-5. Alternate SFR Function in Window 15

Within the SFR space are several registers and bit locations labeled "RESERVED". These locations should never be written or read. A reserved bit location should always be written with 0 to maintain compatibility with future parts. Values read from these locations may change from part to part or over temperature and voltage. Registers and bits which are not labeled should be treated as reserved registers and bits. Note that the default state of internal registers is 0, while that for external memory is 1. This is because SFR functions are typically disabled with a zero, while external memory is typically erased to all 1s.

Caution must be taken when using the SFRs as sources of operations or as base or index registers for indirect or indexed operations. It is possible to get undesired results, since external events can change SFRs and some SFRs clear when read. The potential for an SFR to change value must be taken into account when operating on these registers. This is particularly important when high level languages are used as they may not always make allowances for SFR-type registers. SFRs can be operated on as bytes or words unless otherwise specified.

2.3 Reserved Memory Spaces

Locations 1FFEH and 1FFFH are used for Ports 3 and 4 respectively, allowing easy reconstruction of these ports if external memory is used. An example of reconstructing the I/O ports is given in Section 15. If ports 3 and 4 are not going to be reconstructed and internal ROM/EPROM is not used, these locations can be treated as any other external memory location.

Many reserved and special locations are in the memory area between 2000H and 2080H. In this area the 18 interrupt vectors, chip configuration byte, and security key are located. Figure 2-6 shows the locations and functions of these registers. The interrupts, chip configuration, and security key registers are discussed in Sections 5, 16, and 17 respectively. All unspecified addresses in locations 2000H through 207FH, including those marked "Reserved" are reserved by Intel for use in testing or future products. They must be filled with the Hex value FFH to insure compatibility with future devices.

EXTERNAL MEMORY OR I/O	FFFFH 4000H
INTERNAL PROGRAM STORAGE ROM/EPROM OR EXTERNAL MEMORY	2080H
RESERVED	2074H–207FH
VOLTAGE LEVELS	2072H–2073H
SIGNATURE WORD	2070H–2071H
RESERVED	2040H–206FH
INTERRUPT VECTORS	2030H–203FH
SECURITY KEY	2020H–202FH
RESERVED	2019H–201FH
CHIP CONFIGURATION BYTE	2018H
RESERVED	2015H–2017H
PPW	2014H
INTERRUPT VECTORS	2000H–2013H

Figure 2-6. Reserved Memory Spaces

Resetting the 80C196KB causes instructions to be fetched starting from location 2080H. This location was chosen to allow a system to have up to 8K of RAM continuous with the register file. Further information on reset can be found in Section 13.

2.4 Internal ROM and EPROM

When a ROM part is ordered, or an EPROM part is programmed, the internal memory locations 2080H through 3FFFH are user specified, as are the interrupt vectors, Chip Configuration Register and Security Key in locations 2000H through 207FH. Location 2014H contains the PPW (Programming Pulse Width) register. The PPW is used solely to program 87C196KB EPROM devices and is a reserved location on ROM and ROMless devices.

Instruction and data fetches from the internal ROM or EPROM occur only if the part has ROM or EPROM, $\overline{EA}$ is tied high, and the address is between 2000H and 3FFFH. At all other times data is accessed from either the internal RAM space or external memory and instructions are fetched from external memory. The $\overline{EA}$ pin is latched on $\overline{RESET}$ rising. Information on programming EPROMs can be found in Section 16.

The 80C196KB provides a ROM/EPROM lock feature to allow the program to be locked against reading and/or writing the internal program memory. In order to maintain security, code can not be executed out of the last three locations of internal ROM/EPROM if the lock is enabled. Details on this feature are in Section 17.

2.5 System Bus

There are several modes of system bus operation on the 80C196KB. The standard bus mode uses a 16-bit multiplexed address/data bus. Other bus modes include an 8-bit mode and a mode in which the bus size can dynamically be switched between 8-bits and 16-bits.

Hold/Hold Acknowledge ($\overline{\text{HOLD}}/\overline{\text{HLDA}}$) and Ready signals are available to create a variety of memory systems. The READY line extends the width of the $\overline{\text{RD}}$ (read) and $\overline{\text{WR}}$ (write) pulses to allow access of slow memories. Multiple processor systems with shared memory can be designed using $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ to keep the 80C196KB off the bus. Details on the System Bus are in Section 15.

3.0 SOFTWARE OVERVIEW

This section provides information on writing programs to execute in the 80C196KB. Additional information can be found in the following documents:

MCS®-96 MACRO ASSEMBLER USER'S GUIDE
Order Number 122048 (Intel Systems)
Order Number 122351 (DOS Systems)

MCS®-96 UTILITIES USER'S GUIDE
Order Number 122049 (Intel Systems)
Order Number 122356 (DOS Systems)

PL/M-96 USER'S GUIDE
Order Number 122134 (Intel Systems)
Order Number 122361 (DOS Systems)

C-96 USER'S GUIDE
Order Number 167632 (DOS Systems)

Throughout this chapter short sections of code are used to illustrate the operation of the device. For these sections it is assumed that the following set of temporary registers has been declared:

AX, BX, CX, and DX are 16-bit registers.

AL is the low byte of AX, AH is the high byte.

BL is the low byte of BX

CL is the low byte of CX

DL is the low byte of DX

These are the same as the names for the general data registers used in the 8086. It is important to note that in the 80C196KB these are not dedicated registers but merely the symbolic names assigned by the programmer to an eight byte region within the on-board register file.

3.1 Operand Types

The MCS-96 architecture supports a variety of data types likely to be useful in a control application. To avoid confusion, the name of an operand type is capitalized. A "BYTE" is an unsigned eight bit variable; a "byte" is an eight bit unit of data of any type.

BYTES

BYTES are unsigned 8-bit variables which can take on the values between 0 and 255. Arithmetic and relational operators can be applied to BYTE operands but the result must be interpreted in modulo 256 arithmetic. Logical operations on BYTES are applied bitwise. Bits within BYTES are labeled from 0 to 7, with 0 being the least significant bit.

WORDS

WORDS are unsigned 16-bit variables which can take on the values between 0 and 65535. Arithmetic and relational operators can be applied to WORD operands but the result must be interpreted modulo 65536. Logical operations on WORDS are applied bitwise. Bits within words are labeled from 0 to 15 with 0 being the least significant bit. WORDS must be aligned at even byte boundaries in the MCS-96 address space. The least significant byte of the WORD is in the even byte address and the most significant byte is in the next higher (odd) address. The address of a word is the address of its least significant byte. Word operations to odd addresses are not guaranteed to operate in a consistent manner.

SHORT-INTEGERS

SHORT-INTEGERS are 8-bit signed variables which can take on the values between −128 and +127. Arithmetic operations which generate results outside of the range of a SHORT-INTEGER will set the overflow indicators in the program status word. The actual numeric result returned will be the same as the equivalent operation on BYTE variables.

INTEGERS

INTEGERS are 16-bit signed variables which can take on the values between −32,768 and +32,767. Arithmetic operations which generate results outside of the range of an INTEGER will set the overflow indicators in the program status word. The actual numeric result returned will be the same as the equivalent operation on WORD variables. INTEGERS conform to the same alignment and addressing rules as do WORDS.

BITS

BITS are single-bit operands which can take on the Boolean values of true and false. In addition to the normal support for bits as components of BYTE and WORD operands, the 80C196KB provides for the direct testing of any bit in the internal register file. The MCS-96 architecture requires that bits be addressed as components of BYTES or WORDS, it does not support the direct addressing of bits that can occur in the MCS-51 architecture.

DOUBLE-WORDS

DOUBLE-WORDS are unsigned 32-bit variables which can take on the values between 0 and 4,294,967,295. The MCS-96 architecture provides direct support for this operand type for shifts, as the dividend in a 32-by-16 divide and the product of a 16-by-16 multiply, and for double-word comparisons. For these operations a DOUBLE-WORD variable must reside in the on-board register file of the 80C196KB and be aligned at an address which is evenly divisible by 4. A DOUBLE-WORD operand is addressed by the address of its least significant byte. DOUBLE-WORD operations which are not directly supported can be easily implemented with two WORD operations. For consistency with Intel provided software the user should adopt the conventions for addressing DOUBLE-WORD operands which are discussed in Section 3.5.

LONG-INTEGERS

LONG-INTEGERS are 32-bit signed variables which can take on the values between −2,147,483,648 and +2,147,483,647. The MCS-96 architecture provides direct support for this data type for shifts, as the dividend in a 32-by-16 divide and the product of a 16-by-16 multiply, and for double-word comparisons.

LONG-INTEGERS can also be normalized. For these operations a LONG-INTEGER variable must reside in the onboard register file of the 80C196KB and be aligned at an address which is evenly divisible by 4. A LONG-INTEGER is addressed by the address of its least significant byte.

LONG-INTEGER operations which are not directly supported can be easily implemented with two INTEGER operations. For consistency with Intel provided software, the user should adopt the conventions for addressing LONG operands which are discussed in Section 3.6.

3.2 Operand Addressing

Operands are accessed within the address space of the 80C196KB with one of six basic addressing modes. Some of the details of how these addressing modes work are hidden by the assembly language. If the programmer is to take full advantage of the architecture, it is important that these details be understood. This section will describe the addressing modes as they are handled by the hardware. At the end of this section the addressing modes will be described as they are seen through the assembly language. The six basic address modes which will be described are termed register-direct, indirect, indirect with auto-increment, immediate, short-indexed, and long-indexed. Several other useful addressing operations can be achieved by combining these basic addressing modes with specific registers such as the ZERO register or the stack pointer.

REGISTER-DIRECT REFERENCES

The register-direct mode is used to directly access a register from the 256 byte on-board register file. The register is selected by an 8-bit field within the instruction and the register address must conform to the operand type's alignment rules. Depending on the instruction, up to three registers can take part in the calculation.

Examples

```
ADD    AX,BX,CX       ; AX:=BX+CX
MUL    AX,BX          ; AX:=AX*BX
INCB   CL             ; CL:=CL+1
```

INDIRECT REFERENCES

The indirect mode is used to access an operand by placing its address in a WORD variable in the register file. The calculated address must conform to the alignment rules for the operand type. Note that the indirect address can refer to an operand anywhere within the address space of the 80C196KB, including the register file. The register which contains the indirect address is selected by an eight bit field within the instruction. An instruction can contain only one indirect reference and the remaining operands of the instruction (if any) must be register-direct references.

Examples

```
LD    AX,[AX]      ; AX:=MEM_WORD(AX)
ADDB  AL,BL,[CX]   ; AL:=BL+MEM_BYTE(CX)
POP   [AX]         ; MEM_WORD(AX):=MEM_WORD(SP); SP:=SP+2
```

INDIRECT WITH AUTO-INCREMENT REFERENCES

This addressing mode is the same as the indirect mode except that the WORD variable which contains the indirect address is incremented *after* it is used to address the operand. If the instruction operates on BYTES or SHORT-INTEGERS the indirect address variable will be incremented by one. If the instruction operates on WORDS or INTEGERS the indirect address variable will be incremented by two.

Examples

```
LD    AX,[BX]+      ; AX:=MEM_WORD(BX); BX:=BX+2
ADDB  AL,BL,[CX]+   ; AL:=BL+MEM_BYTE(CX); CX:=CX+1
PUSH  [AX]+         ; SP:=SP-2;
                    ;   MEM_WORD(SP):=MEM_WORD(AX)
                    ;   AX:=AX+2
```

IMMEDIATE REFERENCES

This addressing mode allows an operand to be taken directly from a field in the instruction. For operations on BYTE or SHORT-INTEGER operands this field is eight bits wide. For operations on WORD or INTEGER operands the field is 16 bits wide. An instruction can contain only one immediate reference and the remaining operand(s) must be register-direct references.

Examples

```
ADD  AX,#340  ; AX:=AX+340
PUSH #1234H   ; SP:=SP-2; MEM_WORD(SP):=1234H
DIVB AX,#10   ; AL:=AX/10; AH:=AX MOD 10
```

SHORT-INDEXED REFERENCES

In this addressing mode an eight bit field in the instruction selects a WORD variable in the register file which contains an address. A second eight bit field in the instruction stream is sign-extended and summed with the WORD variable to form the address of the operand which will take part in the calculation. Since the eight bit field is sign-extended, the effective address can be up to 128 bytes before the address in the WORD variable and up to 127 bytes after it. An instruction can contain only one short-indexed reference and the remaining operand(s) must be register-direct references.

Examples

```
LD    AX,12[BX]     ; AX:=MEM_WORD(BX+12)
MULB  AX,BL,3[CX]   ; AX:=BL*MEM_BYTE(CX+3)
```

LONG-INDEXED REFERENCES

This addressing mode is like the short-indexed mode except that a *16-bit* field is taken from the instruction and added to the WORD variable to form the address of the operand. No sign extension is necessary. An instruction can contain only one long-indexed reference and the remaining operand(s) must be register-direct references.

Examples

```
AND  AX,BX,TABLE[CX]     ; AX:=BX AND MEM_WORD(TABLE+CX)
ST   AX,TABLE[BX]        ; MEM_WORD(TABLE+BX):=AX
ADDB AL,BL,LOOKUP[CX]    ; AL:=BL+MEM_BYTE(LOOKUP+CX)
```

ZERO REGISTER ADDRESSING

The first two bytes in the register file are fixed at zero by the 80C196KB hardware. In addition to providing a fixed source of the constant zero for calculations and comparisons, this register can be used as the WORD variable in a long-indexed reference. This combination of register selection and address mode allows any location in memory to be addressed directly.

Examples

```
ADD  AX,1234[0]   ; AX:=AX+MEM_WORD(1234)
POP  5678[0]      ; MEM_WORD(5678):=MEM_WORD(SP)
                  ;  SP:=SP+2
```

STACK POINTER REGISTER ADDRESSING

The system stack pointer in the 80C196KB is accessed as register 18H of the internal register file. In addition to providing for convenient manipulation of the stack pointer, this also facilitates the accessing of operands in the stack. The top of the stack, for example, can be accessed by using the stack pointer as the WORD variable in an indirect reference. In a similar fashion, the stack pointer can be used in the short-indexed mode to access data within the stack.

Examples

```
PUSH  [SP]       ; DUPLICATE TOP_OF_STACK
LD    AX,2[SP]   ; AX:=NEXT_TO_TOP
```

ASSEMBLY LANGUAGE ADDRESSING MODES

The MCS-96 assembly language simplifies the choice of addressing modes to be used in several respects:

Direct Addressing. The assembly language will choose between register-direct addressing and long-indexed with the ZERO register depending on where the operand is in memory. The user can simply refer to an operand by its symbolic name: if the operand is in the register file, a register-direct reference will be used, if the operand is elsewhere in memory, a long-indexed reference will be generated.

Indexed Addressing. The assembly language will choose between short and long indexing depending on the value of the index expression. If the value can be expressed in eight bits then short indexing will be used, if it cannot be expressed in eight bits then long indexing will be used.

These features of the assembly language simplify the programming task and should be used wherever possible.

3.3 Program Status Word

The program status word (PSW) is a collection of Boolean flags which retain information concerning the state of the user's program. There are two bytes in the PSW; the actual status word and the low byte of the interrupt mask. Figure 3-1 shows the status bits of the PSW. The PSW can be saved in the system stack with a single operation (PUSHF) and restored in a like manner (POPF). Only the interrupt section of the PSW can be accessed directly. There is no SFR for the PSW status bits.

CONDITION FLAGS

The PSW bits on the 80C196KB are set as follows:

PSW:	7	6	5	4	3	2	1	0
	Z	N	V	VT	C	X	I	ST

Figure 3-1. PSW Register

Z: The Z (Zero) flag is set to indicate that the operation generated a result equal to zero. For the add-with-carry (ADDC) and subtract-with-borrow (SUBC) operations the Z flag is cleared if the result is non-zero but is never set. These two instructions are normally used in conjunction with the ADD and SUB instructions to perform multiple precision arithmetic. The operation of the Z flag for these instructions leaves it indicating the proper result for the entire multiple precision calculation.

N: The Negative flag is set to indicate that the operation generated a negative result. Note that the N flag will be in the algebraically correct state even if an overflow occurs. For shift operations, including the normalize operation and all three forms (SHL, SHR, SHRA) of byte, word and double word shifts, the N flag will be set to the same value as the most significant bit of the result. This will be true even if the shift count is 0.

V: The oVerflow flag is set to indicate that the operation generated a result which is outside the range for the destination data type. For the SHL, SHLB and SHLL instructions, the V flag will be set if the most significant bit of the operand changes at any time during the shift. For divide operations, the following conditions are used to determine if the V flag is set:

For the operation:	V is set if Quotient is:
UNSIGNED BYTE DIVIDE	> 255(0FFH)
UNSIGNED WORD DIVIDE	> 65535(0FFFFH)
SIGNED BYTE DIVIDE	< −127(81H) or > 127(7FH)
SIGNED WORD DIVIDE	< −32767(8001H) or > 32767(7FFFH)

VT: The oVerflow Trap flag is set when the V flag is set, but it is only cleared by the CLRVT, JVT and JNVT instructions. The operation of the VT flag allows for the testing for a possible overflow condition at the end of a sequence of related arithmetic operations. This is normally more efficient than testing the V flag after each instruction.

C: The Carry flag is set to indicate the state of the arithmetic carry from the most significant bit of the ALU for an arithmetic operation, or the state of the last bit shifted out of an operand for a shift. Arithmetic Borrow after a subtract operation is the complement of the C flag (i.e. if the operation generated a borrow then C=0.)

X: Reserved. Should always be cleared when writing to the PSW for compatibility with future products.

I: The global Interrupt disable bit disables all interrupts when cleared except NMI, TRAP, and unimplemented opcode.

ST: The ST (STicky bit) flag is set to indicate that during a right shift a 1 has been shifted first into the C flag and then been shifted out. The ST flag is undefined after a multiply operation. The ST flag can be used along with the C flag to control rounding after a right shift. Consider multiplying two eight bit quantities and then scaling the result down to 12 bits:

```
MULUB  AX,CL,DL    ;AX:=CL*DL
SHR    AX,#4       ;Shift right 4
                    places
```

If the C flag is set after the shift, it indicates that the bits shifted off the end of the operand were greater-than or equal-to one half the least significant bit (LSB) of the result. If the C flag is clear after the shift, it indicates that the bits shifted off the end of the operand were less than half the LSB of the result. Without the ST flag, the rounding decision must be made on the basis of the C flag alone. (Normally the result would be rounded up if the C flag is set.) The ST flag allows a finer resolution in the rounding decision:

C	ST	Value of the Bits Shifted Off
0	0	Value = 0
0	1	0 < Value < ½ LSB
1	0	Value = ½ LSB
1	1	Value > ½ LSB

Figure 3-2. Rounding Alternatives

Imprecise rounding can be a major source of error in a numerical calculation; use of the ST flag improves the options available to the programmer.

INTERRUPT FLAGS

The lower eight bits of the PSW individually mask the lowest 8 sources of interrupt to the 80C196KB. These mask bits can be accessed as an eight bit byte (INT_MASK—address 8) in the on-board register file. A separate register (INT_MASK1—address 13H) contains the control bits for the higher 8 interrupts. A logical '1' in these bit positions enables the servicing of the corresponding interrupt. Bit 9 in the PSW is the global interrupt disable. If this bit is cleared then interrupts will be locked out. Note that the interrupts are collected in the INT_PEND registers even if they are locked out. Execution of the corresponding service routines will proceed according to their priority when they become enabled. Further information on the interrupt structure of the 80C196KB can be found in Section 5.

3.4 Instruction Set

The MCS-96 instruction set contains a full set of arithmetic and logical operations for the 8-bit data types BYTE and SHORT INTEGER and for the 16-bit data types WORD and INTEGER. The DOUBLE-WORD and LONG data types (32 bits) are supported for the products of 16-by-16 multiplies and the dividends of 32-by-16 divides, for shift operations, and for 32-bit compares. The remaining operations on 32-bit variables can be implemented by combinations of 16-bit operations. As an example the sequence:

```
ADD     AX,CX
ADDC    BX,DX
```

performs a 32-bit addition, and the sequence

```
SUB     AX,CX
SUBC    BX,DX
```

performs a 32-bit subtraction. Operations on REAL (i.e. floating point) variables are not supported directly by the hardware but are supported by the floating point library for the 80C196KB (FPAL-96) which implements a single precision subset of draft 10 of the IEEE standard for floating point arithmetic. The performance of this software is significantly improved by the 80C196KB NORML instruction which normalizes a 32-bit variable and by the existence of the ST flag in the PSW.

In addition to the operations on the various data types, the 80C196KB supports conversions between these types. LDBZE (load byte zero extended) converts a BYTE to a WORD and LDBSE (load byte sign extended) converts a SHORT-INTEGER into an INTEGER. WORDS can be converted to DOUBLE-WORDS by simply clearing the upper WORD of the DOUBLE-WORD (CLR) and INTEGERS can be converted to LONGS with the EXT (sign extend) instruction.

The MCS-96 instructions for addition, subtraction, and comparison do not distinguish between unsigned words and signed integers. Conditional jumps are provided to allow the user to treat the results of these operations as either signed or unsigned quantities. As an example, the CMPB (compare byte) instruction is used to compare both signed and unsigned eight bit quantities. A JH (jump if higher) could be used following the compare if unsigned operands were involved or a JGT (jump if greater-than) if signed operands were involved.

Tables 3-1 and 3-2 summarize the operation of each of the instructions. Complete descriptions of each instruction and its timings can be found in the MCS-96 family Instruction Set chapter.

The execution times for the instruction set are given in Figure 3-3. These times are given for a 16-bit bus with no wait states. On-chip EPROM/ROM space is a 16-bit, zero wait state bus. When executing from an 8-bit external memory system or adding wait states, the CPU becomes bus limited and must sometimes wait for the prefetch queue. The performance penalty for an 8-bit external bus is difficult to measure, but has shown to be between 10 and 30 percent based on the instruction mix. The best way to measure code performance is to actually benchmark the code and time it using an emulator or with TIMER1.

The indirect and indexed instruction timings are given for two memory spaces: SFR/Internal RAM space (0–0FFH), and a memory controller reference (100H–0FFFFH). Any instruction that uses an operand that is referenced through the memory controller (ex. Add r1,5000H[0]) takes 2–3 states longer than if the operand was in the SFR/Internal RAM space. Any data access to on-chip ROM/EPROM is considered to be a memory controller reference.

Flag Settings. The modification to the flag setting is shown for each instruction. A checkmark (✓) means that the flag is set or cleared as appropriate. A hyphen means that the flag is not modified. A one or zero (1) or (0) indicates that the flag will be in that state after the instruction. An up arrow (↑) indicates that the instruction may set the flag if it is appropriate but will not clear the flag. A down arrow (↓) indicates that the flag can be cleared but not set by the instruction. A question mark (?) indicates that the flag will be left in an indeterminant state after the operation.

Table 3-1A. Instruction Summary

Mnemonic	Operands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
ADD/ADDB	2	D ← D + A	✓	✓	✓	✓	↑	–	
ADD/ADDB	3	D ← B + A	✓	✓	✓	✓	↑	–	
ADDC/ADDCB	2	D ← D + A + C	↓	✓	✓	✓	↑	–	
SUB/SUBB	2	D ← D – A	✓	✓	✓	✓	↑	–	
SUB/SUBB	3	D ← B – A	✓	✓	✓	✓	↑	–	
SUBC/SUBCB	2	D ← D – A + C – 1	↓	✓	✓	✓	↑	–	
CMP/CMPB	2	D – A	✓	✓	✓	✓	↑	–	
MUL/MULU	2	D,D + 2 ← D × A	–	–	–	–	–	–	2
MUL/MULU	3	D,D + 2 ← B × A	–	–	–	–	–	–	2
MULB/MULUB	2	D,D + 1 ← D × A	–	–	–	–	–	–	3
MULB/MULUB	3	D,D + 1 ← B × A	–	–	–	–	–	–	3
DIVU	2	D ← (D,D + 2) /A,D + 2 ← remainder	–	–	–	✓	↑	–	2
DIVUB	2	D ← (D,D + 1) /A,D + 1 ← remainder	–	–	–	✓	↑	–	3
DIV	2	D ← (D,D + 2) /A,D + 2 ← remainder	–	–	–	✓	↑	–	
DIVB	2	D ← (D,D + 1) /A,D + 1 ← remainder	–	–	–	✓	↑	–	
AND/ANDB	2	D ← D AND A	✓	✓	0	0	–	–	
AND/ANDB	3	D ← B AND A	✓	✓	0	0	–	–	
OR/ORB	2	D ← D OR A	✓	✓	0	0	–	–	
XOR/XORB	2	D ← D (ecxl. or) A	✓	✓	0	0	–	–	
LD/LDB	2	D ← A	–	–	–	–	–	–	
ST/STB	2	A ← D	–	–	–	–	–	–	
LDBSE	2	D ← A; D + 1 ← SIGN(A)	–	–	–	–	–	–	3,4
LDBZE	2	D ← A; D + 1 ← 0	–	–	–	–	–	–	3,4
PUSH	1	SP ← SP – 2; (SP) ← A	–	–	–	–	–	–	
POP	1	A ← (SP); SP + 2	–	–	–	–	–	–	
PUSHF	0	SP ← SP – 2; (SP) ← PSW; PSW ← 0000H; I ← 0	0	0	0	0	0	0	
POPF	0	PSW ← (SP); SP ← SP + 2; I ← ✓	✓	✓	✓	✓	✓	✓	
SJMP	1	PC ← PC + 11-bit offset	–	–	–	–	–	–	5
LJMP	1	PC ← PC + 16-bit offset	–	–	–	–	–	–	5
BR[indirect]	1	PC ← (A)	–	–	–	–	–	–	
SCALL	1	SP ← SP – 2; (SP) ← PC; PC ← PC + 11-bit offset	–	–	–	–	–	–	5
LCALL	1	SP ← SP – 2; (SP) ← PC; PC ← PC + 16-bit offset	–	–	–	–	–	–	5

Table 3-1B. Instruction Summary

Mnemonic	Operands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
RET	0	PC ← (SP); SP ← SP + 2	—	—	—	—	—	—	
J (conditional)	1	PC ← PC + 8-bit offset (if taken)	—	—	—	—	—	—	5
JC	1	Jump if C = 1	—	—	—	—	—	—	5
JNC	1	jump if C = 0	—	—	—	—	—	—	5
JE	1	jump if Z = 1	—	—	—	—	—	—	5
JNE	1	Jump if Z = 0	—	—	—	—	—	—	5
JGE	1	Jump if N = 0	—	—	—	—	—	—	5
JLT	1	Jump if N = 1	—	—	—	—	—	—	5
JGT	1	Jump if N = 0 and Z = 0	—	—	—	—	—	—	5
JLE	1	Jump if N = 1 or Z = 1	—	—	—	—	—	—	5
JH	1	Jump if C = 1 and Z = 0	—	—	—	—	—	—	5
JNH	1	Jump if C = 0 or Z = 1	—	—	—	—	—	—	5
JV	1	Jump if V = 0	—	—	—	—	—	—	5
JNV	1	Jump if V = 1	—	—	—	—	—	—	5
JVT	1	Jump if VT= 1; Clear VT	—	—	—	—	0	—	5
JNVT	1	Jump if VT = 0; Clear VT	—	—	—	—	0	—	5
JST	1	Jump if ST = 1	—	—	—	—	—	—	5
JNST	1	Jump if ST = 0	—	—	—	—	—	—	5
JBS	3	Jump if Specified Bit = 1	—	—	—	—	—	—	5,6
JBC	3	Jump if Specified Bit = 0	—	—	—	—	—	—	5,6
DJNZ/ DJNZW	1	D ← D − 1; If D ≠ 0 then PC ← PC + 8-bit offset	—	—	—	—	—	—	5 10
DEC/DECB	1	D ← D − 1	✓	✓	✓	✓	↑	—	
NEG/NEGB	1	D ← 0 − D	✓	✓	✓	✓	↑	—	
INC/INCB	1	D ← D + 1	✓	✓	✓	✓	↑	—	
EXT	1	D ← D; D + 2 ← Sign (D)	✓	✓	0	0	—	—	2
EXTB	1	D ← D; D + 1 ← Sign (D)	✓	✓	0	0	—	—	3
NOT/NOTB	1	D ← Logical Not (D)	✓	✓	0	0	—	—	
CLR/CLRB	1	D ← 0	1	0	0	0	—	—	
SHL/SHLB/SHLL	2	C ← msb - - - - - lsb ← 0	✓	✓	✓	✓	↑	—	7
SHR/SHRB/SHRL	2	0 → msb - - - - - lsb → C	✓	✓	✓	0	—	✓	7
SHRA/SHRAB/SHRAL	2	msb → msb - - - - - lsb → C	✓	✓	✓	0	—	✓	7
SETC	0	C ← 1	—	—	1	—	—	—	
CLRC	0	C ← 0	—	—	0	—	—	—	

Table 3-1C. Instruction Summary

Mnemonic	Operands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
CLRVT	0	VT ← 0	—	—	—	—	0	—	
RST	0	PC ← 2080H	0	0	0	0	0	0	8
DI	0	Disable All Interupts (I ← 0)	—	—	—	—	—	—	
EI	0	Enable All Interupts (I ← 1)	—	—	—	—	—	—	
NOP	0	PC ← PC + 1	—	—	—	—	—	—	
SKIP	0	PC ← PC + 2	—	—	—	—	—	—	
NORML	2	Left shift till msb = 1; D ← shift count	✓	✓	0	—	—	—	7
TRAP	0	SP ← SP − 2; (SP) ← PC; PC ← (2010H)	—	—	—	—	—	—	9

PUSHA	1	SP ← SP-2; (SP) ← PSW;
		PSW ← 0000H; SP ← SP-2;
		(SP) ← IMASK1/WSR; IMASK1 ← 00H
			0	0	0	0	0	0	

POPA	1	IMASK1/WSR ← (SP); SP ← SP+2 PSW ← (SP); SP ← SP+2	✓	✓	✓	✓	✓	✓	

IDLPD	1	IDLE MODE IF KEY=1;
		POWERDOWN MODE IF KEY =2;
		CHIP RESET OTHERWISE
			—	—	—	—	—	—	

CMPL	2	D-A	✓	✓	✓	✓	↑	—	
BMOV	2	[PTR_HI]+ ← [PTR_LOW]+ ; UNTIL COUNT=0	—	—	—	—	—	—	

NOTES:

1. If the mnemonic ends in "B" a byte operation is performed, otherwise a word operation is done. Operands D, B, and A must conform to the alignment rules for the required operand type. D and B are locations in the Register File; A can be located anywhere in memory.
2. D,D + 2 are consecutive WORDS in memory; D is DOUBLE-WORD aligned.
3. D,D + 1 are consecutive BYTES in memory; D is WORD aligned.
4. Changes a byte to word.
5. Offset is a 2's complement number.
6. Specified bit is one of the 2048 bits in the register file.
7. The "L" (Long) suffix indicates double-word operation.
8. Initiates a Reset by pulling $\overline{\text{RESET}}$ low. Software should re-initialize all the necessary registers with code starting at 2080H.
9. The assembler will not accept this mnemonic.
10. The DJNZW instruction is not guaranteed to work. See Functional Deviations section.

Table 3-2A. Instruction Length (in Bytes)/Opcode

MNEMONIC	DIRECT	IMMED	INDIRECT		INDEXED	
			NORMAL*(1)	A-INC*(1)	SHORT*(1)	LONG*(1)
ADD (3-op)	4/44	5/45	4/46	4/46	5/47	6/47
SUB (3-op)	4/48	5/49	4/4A	4/4A	5/4B	6/4B
ADD (2-op)	3/64	4/65	3/66	3/66	4/67	5/67
SUB (2-op)	3/68	4/69	3/6A	3/6A	4/6B	5/6B
ADDC	3/A4	4/A5	3/A6	3/A6	4/A7	5/A7
SUBC	3/A8	4/A9	3/AA	3/AA	4/AB	5/AB
CMP	3/88	4/89	3/AB	3/AB	4/8B	5/8B
ADDB (3-op)	4/54	4/55	4/56	4/56	5/57	6/57
SUBB (3-op)	4/58	4/59	4/5A	4/5A	5/5B	6/5B
ADDB (2-op)	3/74	3/75	3/76	3/76	4/77	5/77
SUBB (2-op)	3/78	3/79	3/7A	3/7A	4/7B	5/7B
ADDCB	3/B4	3/B5	3/B6	3/B6	4/B7	5/B7
SUBCB	3/B8	3/B9	3/BA	3/BA	4/BB	5/BB
CMPB	3/98	3/99	3/9A	3/9A	4/9B	5/9B
MUL (3-op)	5/(2)	6/(2)	5/(2)	5/(2)	6/(2)	7/(2)
MULU (3-op)	4/4C	5/4D	4/4E	4/4E	5/4F	6/4F
MUL (2-op)	4/(2)	5/(2)	4/(2)	4/(2)	5/(2)	6/(2)
MULU (2-op)	3/6C	4/6D	3/6E	3/6E	4/6F	5/6F
DIV	4/(2)	5/(2)	4/(2)	4/(2)	5/(2)	6/(2)
DIVU	3/8C	4/8D	3/8E	3/8E	4/8F	5/8F
MULB (3-op)	5/(2)	5/(2)	5/(2)	5/(2)	6/(2)	7/(2)
MULUB (3-op)	4/5C	4/5D	4/5E	4/5E	5/5F	6/5F
MULB (2-op)	4/(2)	4/(2)	4/(2)	4/(2)	5/(2)	6/(2)
MULUB (2-op)	3/7C	3/7D	3/7E	3/7E	4/7F	5/7F
DIVB	4/(2)	4/(2)	4/(2)	4/(2)	5/(2)	6/(2)
DIVUB	3/9C	3/9D	3/9E	3/9E	4/9F	5/9F
AND (3-op)	4/40	5/41	4/42	4/42	5/43	6/43
AND (2-op)	3/60	4/61	3/62	3/62	4/63	5/63
OR (2-op)	3/80	4/81	3/82	3/82	4/83	5/83
XOR	3/84	4/85	3/86	3/86	4/87	5/87
ANDB (3-op)	4/50	4/51	4/52	4/52	5/53	5/53
ANDB (2-op)	3/70	3/71	3/72	3/72	4/73	4/73
ORB (2-op)	3/90	3/91	3/92	3/92	4/93	5/93
XORB	3/94	3/95	3/96	3/96	4/97	5/97
PUSH	2/C8	3/C9	2/CA	2/CA	3/CB	4/CB
POP	2/CC	—	2/CE	2/CE	3/CF	4/CF

NOTES:

1. Indirect and indirect + share the same opcodes, as do short and long indexed opcodes. If the second byte is even, use indirect or short indexed. If odd, use indirect or long indexed.
2. The opcodes for signed multiply and divide are the unsigned opcode with an "FE" prefix.

Table 3-2B. Instruction Length (in Bytes)/Opcode

MNEMONIC	DIRECT	IMMED	INDIRECT		INDEXED	
			NORMAL	A-INC	SHORT	LONG
LD	3/A0	4/A1	3/A2	3/A2	4/A3	5/A3
LDB	3/B0	3/B1	3/B2	3/B2	4/B3	5/B3
ST	3/C0	—	3/C2	3/C2	4/C3	5/C3
STB	3/C4	—	3/C6	3/C6	4/C7	5/C7
LDBSE	3/BC	3/BD	3/BE	3/BE	4/BF	5/BF
LBSZE	3/AC	3/AD	3/AE	3/AE	4/AF	5/AF

Mnemonic	Length/Opcode
PUSHF	1/F2
POPF	1/F3
PUSHA	1/F4
POPA	1/F5
TRAP	1/F7
LCALL	3/EF
SCALL	2/28-2F(3)
RET	1/F0
LJMP	3/E7
SJMP	2/20-27(3)
BR[ ]	2/E3
JNST	1/D0
JST	1/D8
JNH	1/D1
JH	1/D9
JGT	1/D2
JLE	1/DA
JNC	1/B3
JC	1/D8
JNVT	1/D4
JVT	1/DC
JNV	1/D5
JV	1/DD
JGE	1/D6
JLT	1/DE
JNE	1/D7
JE	1/DF
JBC	3/30-37
JBS	3/38-3F

Mnemonic	Length/Opcode
DJNZ	3/E0
DJNZW	3/E1(4)
NORML	3/0F
SHRL	3/0C
SHLL	3/0D
SHRAL	3/0E
SHR	3/08
SHRB	3/18
SHL	3/09
SHLB	3/19
SHRA	3/0A
SHRAB	3/1A
CLRC	1/F8
SETC	1/F9
DI	1/FA
EI	1/FB
CLRVT	1/FC
NOP	1/FD
RST	1/FF
SKIP	2/00
IDLPD	1/F6
BMOV	3/C1

NOTES:

3. The 3 least significant bits of the opcode are concatenated with the 8 bits to form an 11-bit, 2's complement offset.
4. The DJNZW instruction is not guaranteed to work. See Functional Deviations section.

Table 3.3A. Instruction Execution State Times (1)

MNEMONIC	DIRECT	IMMED	INDIRECT		INDEXED	
			NORMAL*	A-INC*	SHORT*	LONG*
ADD (3-op)	5	6	7/10	8/11	7/10	8/11
SUB (3-op)	5	6	7/10	8/11	7/10	8/11
ADD (2-op)	4	5	6/8	7/9	6/8	7/9
SUB (2-op)	4	5	6/8	7/9	6/8	7/9
ADDC	4	5	6/8	7/9	6/8	7/9
SUBC	4	5	6/8	7/9	6/8	7/9
CMP	4	5	6/8	7/9	6/8	7/9
ADDB (3-op)	5	5	7/10	8/11	7/10	8/11
SUBB (3-op)	5	5	7/10	8/11	7/10	8/11
ADDB (2-op)	4	4	6/8	7/9	6/8	7/9
SUBB (2-op)	4	4	6/8	7/9	6/8	7/9
ADDCB	4	4	6/8	7/9	6/8	7/9
SUBCB	4	4	6/8	7/9	6/8	7/9
CMPB	4	4	6/8	7/9	6/8	7/9
MUL (3-op)	16	17	18/21	19/22	19/22	20/23
MULU (3-op)	14	15	16/19	17/19	17/20	18/21
MUL (2-op)	16	17	18/21	19/22	19/22	20/23
MULU (2-op)	14	15	16/19	17/19	17/20	18/21
DIV	26	27	28/31	29/32	29/32	30/33
DIVU	24	25	26/29	27/30	27/30	28/31
MULB (3-op)	12	12	14/17	13/15	15/18	16/19
MULUB (3-op)	10	10	12/15	12/16	12/16	14/17
MULB (2-op)	12	12	14/17	15/18	15/18	16/19
MULUB (2-op)	10	10	12/15	13/15	12/16	14/17
DIVB	18	18	20/23	21/24	21/24	22/25
DIVUB	16	16	18/21	19/22	19/22	20/23
AND (3-op)	5	6	7/10	8/11	7/10	8/11
AND (2-op)	4	5	6/8	7/9	6/8	7/9
OR (2-op)	4	5	6/8	7/9	6/8	7/9
XOR	4	5	6/8	7/9	6/8	7/9
ANDB (3-op)	5	5	7/10	8/11	7/10	8/11
ANDB (2-op)	4	4	6/8	7/9	6/8	7/9
ORB (2-op)	4	4	6/8	7/9	6/8	7/9
XORB	4	4	6/8	7/9	6/8	7/9
LD/LDB	4	5	5/8	6/8	6/9	7/10
ST/STB	4	5	5/8	6/9	6/9	7/10
LDBSE	4	4	5/8	6/8	6/9	7/10
LDBZE	4	4	5/8	6/8	6/9	7/10
BMOV	internal/internal: 6 + 8 per word external/internal: 6 + 11 per word external/external: 6 + 14 per word					
PUSH (int stack)	6	7	9/12	10/13	10/13	11/14
POP (int stack)	8	—	10/12	11/13	11/13	12/14
PUSH (ext stack)	8	9	11/14	12/15	12/15	13/16
POP (ext stack)	11	—	13/15	14/16	14/16	15/17

*Times for operands as: SFRs and internal RAM (0–1FFH)/memory controller (200H–0FFFFH)

NOTE:

1. Execution times for memory controller references may be one to two states higher depending on the number of bytes in the prefetch queue. Internal stack is 200H–1FFH and external stack is 200H–0FFFFH.

Table 3.3B. Instruction Execution State Times

MNEMONIC		MNEMONIC	
PUSHF (int stack)	6	PUSHF (ext stack)	8
POPF (int stack)	7	POPF (ext stack)	10
PUSHA (int stack)	12	PUSHA (ext stack)	18
POPA (int stack)	12	POPA (ext stack)	18
TRAP (int stack)	16	TRAP (ext stack)	18
LCALL (int stack)	11	LCALL (ext stack)	13
SCALL (int stack)	11	SCALL (ext stack)	13
RET (int stack)	11	RET (ext stack)	14
CMPL	7	DEC/DECB	3
CLR/CLRB	3	EXT/EXTB	4
NOT/NOTB	3	INC/INCB	3
NEG/NEGB	3		
LJMP	7		
SJMP	7		
BR [indirect]	7		
JNST, JST	4/8 jump not taken/jump taken		
JNH, JH	4/8 jump not taken/jump taken		
JGT, JLE	4/8 jump not taken/jump taken		
JNC, JC	4/8 jump not taken/jump taken		
JNVT, JVT	4/8 jump not taken/jump taken		
JNV, JV	4/8 jump not taken/jump taken		
JGE, JLT	4/8 jump not taken/jump taken		
JNE, JE	4/8 jump not taken/jump taken		
JBC, JBS	5/9 jump not taken/jump taken		
DJNZ	5/9 jump not taken/jump taken		
DJNZW (Note 1)	5/9 jump not taken/jump taken		
NORML	8 + 1 per shift (9 for 0 shift)		
SHRL	7+ 1 per shift (8 for 0 shift)		
SHLL	7 + 1 per shift (8 for 0 shift)		
SHRAL	7 + 1 per shift (8 for 0 shift)		
SHR/SHRB	6 + 1 per shift (7 for 0 shift)		
SHL/SHLB	6 + 1 per shift (7 for 0 shift)		
SHRA/SHRAB	6 + 1 per shift (7 for 0 shift)		
CLRC	2		
SETC	2		
DI	2		
EI	2		
CLRVT	2		
NOP	2		
RST	15 (includes fetch of configuration byte)		
SKIP	3		
IDLPD	8/25 (proper key/improper key)		

NOTE:
1. The DJNZW instruction is not guaranteed to work. See Functional Deviations section.

3.5 80C196KB Instruction Set Additions and Differences

For users already familiar with the 8096BH, there are six instructions added to the standard MCS-96 instruction set to form the 80C196KB instruction set. All of the former instructions perform the same function, except as indicated in the next section. The new instructions and their descriptions are listed below:

PUSHA — PUSHes the PSW, INT_MASK, IMASK1, and WSR

POPA — POPs the PSW, INT_MASK, IMASK1, and WSR

IDLPD — Sets the part into IDLE or Powerdown mode

CMPL — Compare 2 long direct values

BMOV — Block move using 2 auto-incrementing pointers and a counter

DJNZW — Decrement Jump Not Zero using a Word counter (Not functional on current stepping.)

INSTRUCTION DIFFERENCES

Instruction times on the 80C196KB are shorter than those on the 8096 for many instructions. For example a 16×16 unsigned multiply has been reduced from 25 to 14 states. In addition, many zero and one operand instructions and most instructions using external data take one or two fewer state times.

Indexed and indirect operations relative to the stack pointer (SP) work differently on the 80C196KB than on the 8096BH. On the 8096BH, the address is calculated based on the un-updated version of the stack pointer. The 80C196KB uses the updated version. The offset for POP[SP] and POP nn[SP] instructions may need to be changed by a count of 2.

3.6 Software Standards and Conventions

For a software project of any size it is a good idea to modularize the program and to establish standards which control the communication between these modules. The nature of these standards will vary with the needs of the final application. A common component of all of these standards, however, must be the mechanism for passing parameters to procedures and returning results from procedures. In the absence of some overriding consideration which prevents their use, it is suggested that the user conform to the conventions adopted by the PLM-96 programming language for procedure linkage. It is a very usable standard for both the assembly language and PLM-96 environment and it offers compatibility between these environments. Another advantage is that it allows the user access to the same floating point arithmetics library that PLM-96 uses to operate on REAL variables.

REGISTER UTILIZATION

The MCS-96 architecture provides a 256 byte register file. Some of these registers are used to control register-mapped I/O devices and for other special functions such as the ZERO register and the stack pointer. The remaining bytes in the register file, some 230 of them, are available for allocation by the programmer. If these registers are to be used effectively, some overall strategy for their allocation must be adopted. PLM-96 adopts the simple and effective strategy of allocating the eight bytes between addresses 1CH and 23H as temporary storage. The starting address of this region is called PLMREG. The remaining area in the register file is treated as a segment of memory which is allocated as required.

ADDRESSING 32-BIT OPERANDS

These operands are formed from two adjacent 16-bit words in memory. The least significant word of the double word is always in lower address, even when the data is in the stack (which means that the most significant word must be pushed into the stack first). A double word is addressed by the address of its least significant byte. Note that the hardware supports some operations on double words. For these operations the double word must be in the internal register file and must have an address which is evenly divisible by four.

SUBROUTINE LINKAGE

Parameters are passed to subroutines in the stack. Parameters are pushed into the stack in the order that they are encountered in the scanning of the source text. Eight-bit parameters (BYTES or SHORT-INTEGERS) are pushed into the stack with the high order byte undefined. Thirty-two bit parameters (LONG-INTEGERS, DOUBLE-WORDS, and REALS) are pushed onto the stack as two 16-bit values; the most significant half of the parameter is pushed into the stack first.

As an example, consider the following PLM-96 procedure:

```
example_procedure: PROCEDURE
(param1,param2,param3);
  DECLARE param1 BYTE,
          param2 DWORD,
          param3 WORD;
```

When this procedure is entered at run time the stack will contain the parameters in the following order:

?????? : param1	
high word of param2	
low word of param2	
param3	
return address	← Stack_pointer

Figure 3-5. Stack Image

If a procedure returns a value to the calling code (as opposed to modifying more global variables) then the result is returned in the variable PLMREG. PLMREG is viewed as either an 8-, 16- or 32-bit variable depending on the type of the procedure.

The standard calling convention adopted by PLM-96 has several key features:

a) Procedures can always assume that the eight bytes of register file memory starting at PLMREG can be used as temporaries within the body of the procedure.

b) Code which calls a procedure must assume that the eight bytes of register file memory starting at PLMREG are modified by the procedure.

c) The Program Status Word (PSW—see Section 3.3) is not saved and restored by procedures so the calling code must assume that the condition flags (Z, N, V, VT, C, and ST) are modified by the procedure.

d) Function results from procedures are always returned in the variable PLMREG.

PLM-96 allows the definition of INTERRUPT procedures which are executed when a predefined interrupt occurs. These procedures do not conform to the rules of a normal procedure. Parameters cannot be passed to these procedures and they cannot return results. Since they can execute essentially at any time (hence the term interrupt), these procedures must save the PSW and PLMREG when they are entered and restore these values before they exit.

3.7 Software Protection Hints

Several features to assist in recovery from hardware and software errors are available on the 80C196KB. Protection is also provided against executing unimplemented opcodes by the unimplemented opcode interrupt. In addition, the hardware reset instruction (RST) can cause a reset if the program counter goes out of bounds. This instruction has an opcode of 0FFH, so if the processor reads in bus lines which have been pulled high it will reset itself.

It is recommended that unused areas of code be filled with NOPs and periodic jumps to an error routine or RST (reset chip) instructions. This is particularly important in the code around lookup tables, since if lookup tables are executed undesired results will occur. Wherever space allows, each table should be surrounded by 7 NOPs (the longest 80C196KB instruction has 7 bytes) and a RST or jump to error routine instruction. Since RST is a one-byte instruction, the NOPs are not needed if RSTs are used instead of jumps to an error routine. This will help to ensure a speedy recovery should the processor have a glitch in the program flow.

The Watchdog Timer (WDT) further protects against software and hardware errors. When using the WDT to protect software it is desirable to reset it from only one place in code, lessening the chance of an undesired WDT reset. The section of code that resets the WDT should monitor the other code sections for proper operation. This can be done by checking variables to make sure they are within reasonable values. Simply using a software timer to reset the WDT every 10 milliseconds will provide protection only for catastrophic failures.

4.0 PERIPHERAL OVERVIEW

There are five major peripherals on the 80C196KB: the pulse-width-modulated output (PWM), Timer1 and Timer2, High Speed I/O Unit, Serial Port and A/D Converter. With the exception of the high speed I/O unit (HSIO), each of the peripherals is a single unit that can be discussed without further separation.

Four individual sections make up the HSIO and work together to form a very flexible timer/counter based I/O system. Included in the HSIO are a 16-bit timer (Timer1), a 16-bit up/down counter (Timer2), a programmable high speed input unit (HSI), and a programmable high speed output unit (HSO). With very little CPU overhead the HSIO can measure pulse widths, generate waveforms, and create periodic interrupts. Depending on the application, it can perform the work of up to 18 timer/counters and capture/compare registers.

A brief description of the peripheral functions and interractions is included in this section. It provides overview information prior to the detailed discussions in the following sections. All of the details on control bits and precautions are in the individual sections for each peripheral starting with Section 5.

4.1 Pulse Width Modulation Output (D/A)

Digital to analog conversion can be done with the Pulse Width Modulation output. The output waveform is a variable duty cycle pulse which repeats every 256 state times or 512 state times if the prescaler is enabled. Changes in the duty cycle are made by writing to the PWM register. There are several types of motors which require a PWM waveform for most efficient operation. Additionally, if this waveform is integrated it will produce a DC level which can be changed in 256 steps by varying the duty cycle. Details on the PWM are in Section 6.

4.2 Timers

Two 16-bit timers are available for use on the 80C196KB. The first is designated "Timer1", the second "Timer2". Timer1 is used to synchronize events to real time, while Timer2 is clocked externally and synchronizes events to external occurrences. The timers are the time bases for the High Speed Input (HSI) and High Speed Output (HSO) units and can be considered an integral part of the HSI/O. Details on the timers are in Section 7.

Timer1 is a free-running timer which is incremented every eight state times, just as it is on the 8096BH. Timer1 can cause an interrupt when it overflows.

Timer2 counts transitions, both positive and negative, on its input which can be either the T2CLK pin or the HSI.1 pin. Timer2 can be read and written and can be reset by hardware, software or the HSO unit. It can be used as an up/down counter based on Port 2.6 and it's value can be captured into the T2CAPture register. Interrupts can be generated on capture events and if Timer2 crosses the 0FFFFH/0000H boundary or the 7FFFH/8000H boundary in either direction.

4.3 High Speed Inputs (HSI)

The High Speed Input (HSI) unit can capture the value of Timer1 when an event takes place on one of four input pins (HSI.0-HSI.3). Four types of events can trigger a capture: rising edges only, falling edges only, rising or falling edges, or every eighth rising edge. A block diagram of this unit is shown in Figure 4-3. Details on the HSI unit are in Section 8.

When events occur, the Timer1 value gets stored in the FIFO along with 4 status bits which indicate the input line(s) that caused the event. The next event ready to be unloaded from the FIFO is placed in the HSI Holding Register, so a total of 8 pieces of data can be stored in the FIFO. Data is taken off the FIFO by reading the HSI_STATUS register, followed by reading the HSI_TIME register. When the time register is read the next FIFO location is loaded into the holding register.

Three forms of HSI interrupts can be generated: when a value moves from the FIFO into the holding register; when the FIFO (independent of the holding register) has 4 or more events stored; and when the FIFO has 6 or more events stored. This flexibility allows optimization of the HSI for the expected frequency of interrupts.

Independent of the HSI operation, the state of the HSI pins is indicated by 4 bits of the HSI_STATUS register. Also independent of the HSI operation is the HSI.0 pin interrupt, which can be used as an extra external interrupt even if the pin is not enabled to the HSI unit.

4.4 High Speed Outputs (HSO)

The High Speed Output (HSO) unit can generate events at specified times or counts based on Timer1 or Timer2 with minimal CPU overhead. A block diagram of the HSO unit is shown in Figure 4-4. Up to 8 pending events can be stored in the CAM (Content Addressable Memory) of the HSO unit at one time. Commands are placed into the HSO unit by first writing to HSO_COMMAND with the event to occur, and then to HSO_TIME with the timer match value.

Fourteen different types of events can be triggered by the HSO: 8 external and 6 internal. There are two interrupt vectors associated with the HSO, one for external events, and one for internal events. External events consist of switching one or more of the 6 HSO pins (HSO.0-HSO.5). Internal events include setting up 4 Software Timers, resetting Timer2, and starting an A/D conversion. The software timers are flags that can be set by the HSO and optionally cause interrupts. Details on the HSO Unit are in Section 9.

4.5 Serial Port

The serial port on the 80C196KB is functionally compatible with the serial port on the MCS-51 and MCS-96 families of microcontrollers. One synchronous and three asynchronous modes are available. The asynchronous modes are full duplex, meaning they can transmit and receive at the same time. Double buffering is provided for the receiver so that a second byte can be received before the first byte has been read. The transmitter is also double buffered, allowing bytes to be written while transmission is still in progress.

The Serial Port STATus (SP_STAT) register contains bits to indicate receive overrun, parity, and framing errors, and transmit and receive interrupts. Details on the Serial Port are in Section 10.

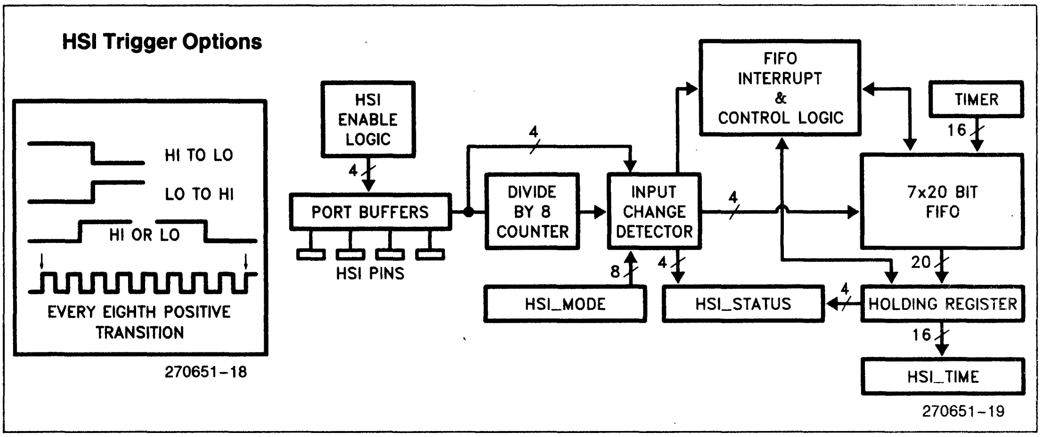


Figure 4-3. HSI Block Diagram

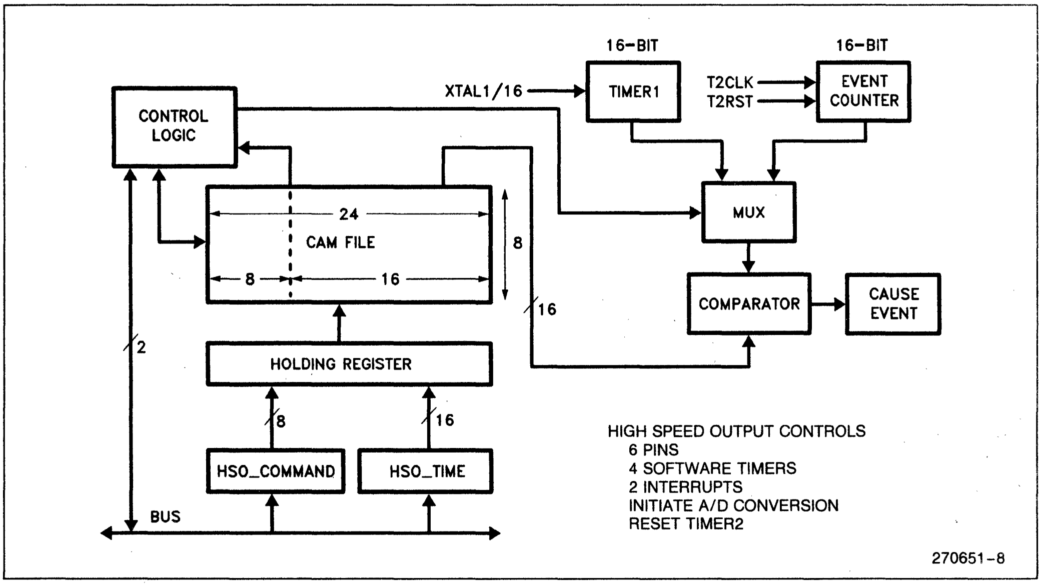


Figure 4-4. HSO Block Diagram

MODES OF OPERATION

Mode 0 is a synchronous mode which is commonly used for shift register based I/O expansion. Sets of 8 bits are shifted in or out of the 80C196KB with a data signal and a clock signal.

Mode 1 is the standard asynchronous communications mode: the data frame used in this mode consists of 10 bits: a start bit (0), 8 data bits (LSB first), and a stop bit (1). Parity can be enabled to send an even parity bit instead of the 8th data bit and to check parity on reception.

Modes 2 and 3 are 9-bit modes commonly used for multi-processor communications. The data frame used in these modes consist of a start bit (0), 9 data bits (LSB first), and a stop bit (1). When transmitting, the 9th data bit can be set to a one to indicate an address or other global transmission. Devices in Mode 2 will be interrupted only if this bit is set. Devices in Mode 3 will be interrupted upon any reception. This provides an easy way to have selective reception on a data link. Mode 3 can also be used to send and receive 8 bits of data plus even parity.

BAUD RATES

Baud rates are generated in an independent 15-bit counter based on either the T2CLK pin or XTAL1 pin. Common baud rates can be easily generated with standard crystal frequencies. A maximum baud rate of 750 Kbaud is available in the asynchronous modes with 12MHz on XTAL1. The synchronous mode has a maximum rate of 3.0 Mbaud with a 12 MHz clock.

4.6 A/D Converter

The 80C196KB's Analog interface consists of a sample-and-hold, an 8-channel multiplexer, and a 10-bit successive approximation analog-to-digital converter.

Analog signals can be sampled by any of the 8 analog input pins (ACH0 through ACH7) which are shared with Port 0. An A/D conversion is performed on one input at a time using successive approximation with a result equal to the ratio of the input voltage divided by the analog supply voltage. If the ratio is 1.00, then the result will be all ones. A conversion can be started by writing to the A/D_Command register or by an HSO Command. Details on the A/D converter are in Section 11.

4.7 I/O Ports

There are five 8-bit I/O ports on the 80C196KB. Some of these ports are input only, some are output only, some are bidirectional and some have multiple functions. In addition to these ports, the HSI/O pins can be used as standard I/O pins if their timer related features are not needed.

Port 0 is an input port which is also the analog input for the A/D converter. Port 1 is a quasi-bidirectional port and the 3MSBs of Port 1 are multiplexed with the $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ functions. Port 2 contains three types of port lines: quasi-bidirectional, input and output. Its input and output lines are shared with other functions such as serial port receive and transmit and Timer2 clock and reset. Ports 3 and 4 are open-drain bidirectional ports which share their pins with the address/data bus.

Quasi-bidirectional pins can be used as input and output pins without the need for a data direction register. They output a strong low value and a weak high value. The weak high value can be externally pulled low providing an input function. A detailed explanation of these ports can be found in Section 12.

4.8 Watchdog Timer

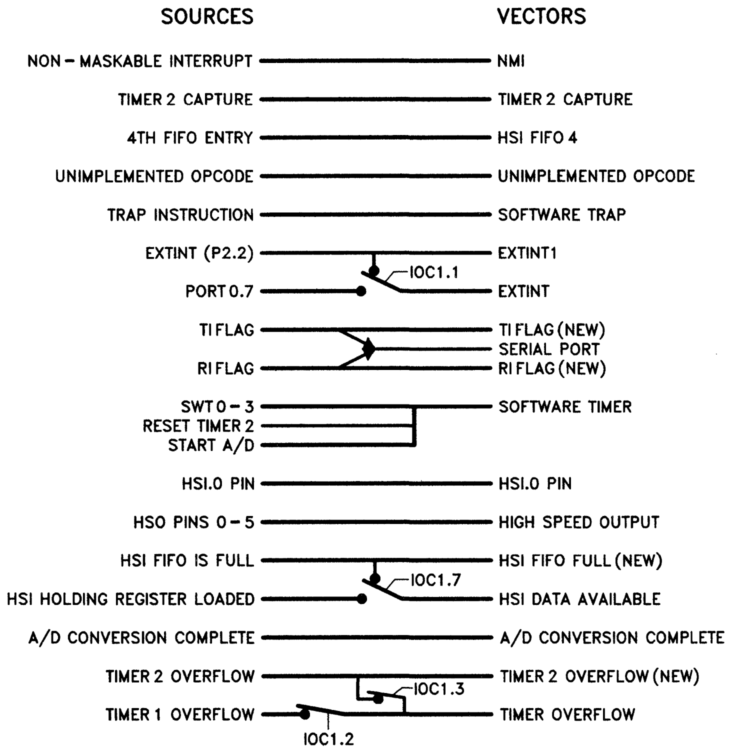
The Watchdog Timer (WDT) provides a means to recover gracefully from a software upset. When the watchdog is enabled it will initiate a hardware reset unless the software clears it every 64K state times. Hardware resets on the 80C196KB cause the $\overline{\text{RESET}}$ input pin to be pulled low, providing a reset signal to other components on the board. The WDT is independent of the other timers on the 80C196KB.

5.0 INTERRUPTS

Twenty-eight (28) sources of interrupts are available on the 80C196KB. These sources are gathered into 15 vectors plus special vectors for NMI, the TRAP instruction, and Unimplemented Opcodes. Figure 5-1 shows the routing of the interrupt sources into their vectors as well as the control bits which enable some of the sources.

Special Interrupts

Three special interrupts are available on the 80C196KB: NMI, TRAP and Unimplemented opcode. The external NMI pin generates an unmaskable interrupt for implementation of critical interrupt routines. The TRAP instruction is useful in the development of custom software debuggers or generation of software interrupts. The unimplemented opcode interrupt generates an interrupt when unimplemented opcodes are exe-



270651-9

Figure 5-1. 80C196KB Interrupt Sources

cuted. This provides software recovery from random execution during hardware and software failures. Although available for customer use, these interrupts may be used in Intel development tools or evaluation boards.

NMI

NMI, the external Non-Maskable Interrupt, is the highest priority interrupt. It vectors indirectly through location 203EH. For design symmetry, a mask bit exists in INT_MASK1 for the NMI. To prevent accidental masking of an NMI, the bit does not function and will not stop an NMI from occurring. For future compatibility, the NMI mask bit must be set to zero.

NMI on the 8096 vectored directly to location 0000H, so for the 80C196KB to be compatible with 8096 software, which uses the NMI, location 203EH must be loaded with 0000H. The NMI interrupt vector and interrupt vector location is used by some Intel development tools. For example, the EV80C196KB evaluation board uses the NMI to process serial communication interrupts from the host. The NMI interrupt routine executes monitor commands passed from the host.

The NMI interrupt is sampled during PH1 or CLKOUT low and is latched internally. If the pin is held high, multiple interrupts will not occur.

TRAP

Opcode 0F7H, the TRAP instruction, causes an indirect vector through location 2010H. The TRAP instruction provides a single instruction interrupt useful in designing software debuggers. The TRAP instruction prevents the acknowledgement of interrupts until after execution of the next instruction.

Unimplemented Opcode

Opcodes which are not implemented on the 80C196KB will cause an indirect vector through location 2012H. User code or hardware which may have failed and run into an unimplemented opcode can software recover through this interrupt. The DJNZW instruction is not supported on the 80C196KB but remains a valid opcode, therefore, no interrupt will occur.

The programmer must initialize the interrupt vector table with the starting addresses of the appropriate interrupt service routines. It is suggested that any unused interrupts be vectored to an error handling routine. In a debug environment, it may be desirable to have the routine lock into a jump to self loop which would be easily traceable with emulation tools. More sophisticated routines may be appropriate for production code recoveries.

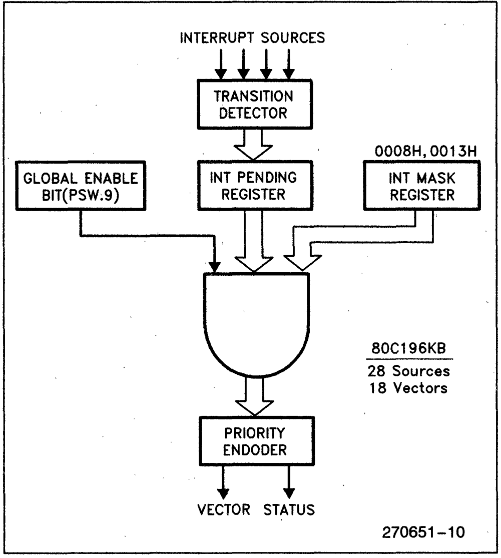


Figure 5-2. 80C196KB Interrupt Structure Block Diagram

Five registers control the operation of the interrupt system: INT_PEND, INT_PEND1, INT_MASK and INT_MASK1 and the PSW which contains a global disable bit. A block diagram of the system is shown in Figure 5-2. The transition detector looks for 0 to 1 transitions on any of the sources. External sources have a maximum transition speed of one edge every state time. Sampling will be guaranteed if the level on the interrupt line is held for at least one state time. If the interrupt line is not held for at least one state time, the interrupt may not be detected.

5.1 Interrupt Control

Interrupt Pending Register

When the hardware detects one of the sixteen interrupts it sets the corresponding bit in one of two pending interrupt registers (INT__PEND-09H and INT__PEND1-12H). When the interrupt vector is taken, the pending bit is cleared. These registers, the formats of which are shown in Figure 5-3, can be read or modified as byte registers. They can be read to determine which of the interrupts are pending at any given time or modified to either clear pending interrupts or generate interrupts under software control. Any software which modifies the INT__PEND registers should ensure that the entire operation is inseparable. The easiest way to do this is to use the logical instructions in the two or three operand format, for example:

```
ANDB   INT_PEND,#11111101B
           ; Clears the A/D Interrupt
ORB    INT_PEND,#00000010B
           ; Sets the A/D Interrupt
```

Caution must be used when writing to the pending register to clear interrupts. If the interrupt has already been acknowledged when the bit is cleared, a 5 state time "partial" interrupt cycle will occur. This is because the 80C196KB will have to fetch the next instruction of the normal instruction flow, instead of proceeding with the interrupt processing. The effect on the program will be essentially that of an extra two NOPs. This can be prevented by clearing the bits using a 2 operand immediate logical, as the 80C196KB holds off acknowledging interrupts during these "read/modify/write" instructions.

Interrupt Mask Register

Individual interrupts can be enabled or disabled by setting or clearing bits in the interrupt mask registers (INT__MASK-08H and INT__MASK1-13H). The format of these registers is the same as that of the Interrupt Pending Register shown in Figure 5-3.

The INT__MASK and INT__MASK1 registers can be read or written as byte registers. A one in any bit position will enable the corresponding interrupt source and a zero will disable the source. The hardware will save any interrupts that occur by setting bits in the pending register, even if the interrupt mask bit is cleared. The INT__MASK register is the lower eight bits of the PSW so the PUSHF and POPF instructions save and restore the INT__MASK register as well as the global interrupt lockout and the arithmetic flags. Both the INT__MASK and INT__MASK1 registers can be saved with the PUSHA and POPA Instructions.

Global Disable

The processing of all interrupts except the NMI, TRAP and unimplemented opcode interrupts can be disabled by clearing the I bit in the PSW. Setting the I bit will enable interrupts that have mask register bits which are set. The I bit is controlled by the EI (Enable Interrupts) and DI (Disable Interrupts) instructions. Note that the I bit only controls the actual servicing of interrupts. Interrupts that occur during periods of lockout will be held in the pending register and serviced on a prioritized basis when the lockout period ends.

5.2 Interrupt Priorities

The priority encoder looks at all of the interrupts which are both pending and enabled, and selects the one with the highest priority. The priorities are shown in Figure 5-4 (15 is highest, 0 is lowest). The interrupt generator then forces a call to the location in the indicated vector location. This location would be the starting location of the Interrupt Service Routine (ISR).

	7	6	5	4	3	2	1	0
12H IPEND1: 13H IMASK1:	NMI	FIFO FULL	EXT INT1	T2 OVF	T2 CAP	HSI4	RI	TI

	7	6	5	4	3	2	1	0
09H IPEND: 08H IMASK:	EXT INT	SER PORT	SOFT TIMER	HSI.0 PIN	HSO PIN	HSI DATA	A/D DONE	TIMER OVF

Figure 5-3. Interrupt Mask and Pending Registers

Number	Source	Vector Location	Priority
INT15	NMI	203EH	15
INT14	HSI FIFO Full	203CH	14
INT13	EXTINT1	203AH	13
INT12	TIMER2 Overflow	2038H	12
INT11	TIMER2 Capture	2036H	11
INT10	4th Entry into HSI FIFO	2034H	10
INT09	RI	2032H	9
INT08	TI	2030H	8
SPECIAL	Unimplemented Opcode	2012H	N/A
SPECIAL	Trap	2010H	N/A
INT07	EXTINT	200EH	7
INT06	Serial Port	200CH	6
INT05	Software Timer	200AH	5
INT04	HSI.0 Pin	2008H	4
INT03	High Speed Outputs	2006H	3
INT02	HSI Data Available	2004H	2
INT01	A/D Conversion Complete	2002H	1
INT00	Timer Overflow	2000H	0

Figure 5-4. 80C196KB Interrupt Priorities

This priority selection controls the order in which pending interrupts are passed to the software via interrupt calls. The software can then implement its own priority structure by controlling the mask registers (INT_MASK and INT_MASK1). To see how this is done, consider the case of a serial I/O service routine which must run at a priority level which is lower than the HSI data available interrupt but higher than any other source. The "preamble" and exit code for this interrupt service routine would look like this:

```
serial_io_isr:
 PUSHA          ; Save the PSW, INT_MASK
                ; INT_MASK1, and WSR
 LDB   INT_MASK,#00000100B
 EI             ; Enable interrupts again
 ;
 ;
 ;
 ;   }  Service the interrupt
 ;
 ;
 ;
 POPA           ; Restore
 RET
```

Note that location 200CH in the interrupt vector table would have to be loaded with the label serial_io_isr and the interrupt be enabled for this routine to execute.

There is an interesting chain of instruction side-effects which makes this (or any other) 80C196KB interrupt service routine execute properly:

A) After the interrupt controller decides to process an interrupt, it executes a "CALL", using the location from the corresponding interrupt vector table entry as the destination. The return address is pushed onto the stack. Another interrupt cannot be serviced until after the first instruction following the interrupt call is executed.

B) The PUSHA instruction, which is now guaranteed to execute, saves the PSW, INT_MASK, INT_MASK1, and the WSR on the stack as two words, and clears them. An interrupt cannot be serviced immediately following a PUSHA instruction. (If INT_MASK1 and the WSR register are not used, or 8096BH code is being executed, PUSHF, which saves only the PSW and INT_MASK, can be used in place of PUSHA).

C) LD INT_MASK, which is guaranteed to execute, enables those interrupts that are allowed to interrupt this ISR. This allows the software to establish its own priorities independent of the hardware.

D) The EI instruction reenables the processing of interrupts with the new priorities.

E) At the end of the ISR, the POPA instruction restores the PSW, INT_MASK, INT_MASK1, and WSR to their original state when the interrupt occurred. Interrupts cannot occur immediately following a POPA instruction so the RET instruction is guaranteed to execute. This prevents the stack from overflowing if interrupts are occurring at high frequency. (If INT_MASK1 and the WSR are not being used, or 8096BH code is being executed, POPF, which restores only the PSW and INT_MASK, can be used in place of POPA.)

Notice that the "preamble" and exit code for the interrupt service routine does not include any code for saving or restoring registers. This is because it has been assumed that the interrupt service routine has been allocated its own private set of registers from the onboard register file. The availability of some 230 bytes of register storage makes this quite practical.

5.3 Critical Regions

Interrupt service routines must sometimes share data with other routines. Whenever the programmer is coding those sections of code which access these shared pieces of data, great care must be taken to ensure that the integrity of the data is maintained. Consider clearing a bit in the interrupt pending register as part of a non-interrupt routine:

```
LDB      AL,INT_PEND
ANDB     AL,#bit_mask
STB      AL,INT_PEND
```

This code works if no other routines are operating concurrently, but will cause occasional but serious problems if used in a concurrent environment. (All programs which make use of interrupts must be considered to be part of a concurrent environment.) To demonstrate this problem, assume that the INT__PEND register contains 00001111B and bit 3 (HSO event interrupt pending) is to be reset. The code does work for this data pattern but what happens if an HSI interrupt occurs somewhere between the LDB and the STB instructions? Before the LDB instruction INT__PEND contains 00001111B and after the LDB instruction so does AL. If the HSI interrupt service routine executes at this point then INT__PEND will change to 00001011B. The ANDB changes AL to 00000111B and the STB changes INT__PEND to 00000111B. It should be 00000011B. This code sequence has managed to generate a false HSI interrupt The same basic process can generate an amazing assortment of problems and headaches. These problems can be avoided by assuring mutual exclusion which basically means that if more than one routine can change a variable, then the programmer must ensure exclusive access to the variable during the entire operation on the variable.

In many cases the instruction set of the 80C196KB allows the variable to be modified with a single instruction. The code in the above example can be implemented with a single instruction.

```
ANDB     INT_PEND,#bit_mask
```

Instructions are indivisible so mutual exclusion is ensured in this case. Changes to the INT__PEND or INT__PEND1 register must be made as a single instruction, since bits can be changed in this register even if interrupts are disabled. Depending on system configurations, several other SFRs might also need to be changed in a single instruction for the same reason.

When variables must be modified without interruption, and a single instruction can not be used, the programmer must create what is termed a critical region in which it is safe to modify the variable. One way to do this is to simply disable interrupts with a DI instruction, perform the modification, and then re-enable interrupts with an EI instruction. The problem with this approach is that it leaves the interrupts enabled even if they were not enabled at the start. A better solution is to enter the critical region with a PUSHF instruction which saves the PSW and also clears the interrupt enable flags. The region can then be terminated with a POPF instruction which returns the interrupt enable to the state it was in before the code sequence. It should be noted that some system configurations might require more protection to form a critical region. An example is a system in which more than one processor has access to a common resource such as memory or external I/O devices.

5.4 Interrupt Timing

The 80C196KB can be interrupted from four different external sources; NMI, P2.2, HSI.0 and P0.7. All external interrupts are sampled during PH1 or CLKOUT low and are latched internally. Holding levels on external interrupts for at least one state time will ensure recognition of the interrupts.

The external interrupts on the 80C196KB, although sampled during PH1, are edge triggered interrupts as opposed to level triggered. Edge triggered interrupts will generate only one interrupt if the input is held high. On the other hand, level triggered interrupts will generate multiple interrupts when held high.

Interrupts are not always acknowledged immediately. If the interrupt signal does not occur prior to 4 statetimes before the end of an instruction, the interrupt may not be acknowledged until after the next instruction has been executed. This is because an instruction is fetched and prepared for execution a few state times before it is actually executed.

There are 6 instructions which always inhibit interrupts from being acknowledged until after the next instruction has been executed. These instructions are:

EI, DI — Enable and disable all interrupts by toggling the global disable bit (PSW.9).

PUSHF — PUSH Flags pushes the PSW/INT__MASK pair then clears it, leaving both INT__MASK and PSW.9 clear.

POPF — POP Flags pops the PSW/INT_MASK pair off the stack

PUSHA — PUSH All does a PUSHF, then pushes the INT_MASK1/WSR pair and clears INT_MASK1

POPA — POP All pops the INT_MASK1/WSR pair and then does a POPF

Interrupts can also not occur immediately after execution of:

Unimplemented Opcodes

TRAP — The software trap instruction

SIGND — The signed prefix for multiply and divide instructions

When an interrupt is acknowledged the interrupt pending bit is cleared, and a call is forced to the location indicated by the specified interrupt vector. This call occurs after the completion of the instruction in process, except as noted above. The procedure of getting the vector and forcing the call requires 16 state times. If the stack is in external RAM an additional 2 state times are required.

The maximum number of state times required from the time an interrupt is generated (not acknowledged) until the 80C196KB begins executing code at the desired location is the time of the longest instruction, NORML (Normalize — 39 state times), plus the 4 state times prior to the end of the previous instruction, plus the response time (16(internal stack) or 18(external stack) state times). Therefore, the maximum response time is 61 (39 + 4 + 18) state times. This does not include the 10 state times required for PUSHF if it is used as the first instruction in the interrupt routine or additional latency caused by having the interrupt masked or disabled. Refer to Figure 5-5, Interrupt Response Time, to visualize an example of worst case scenario.

Interrupt latency time can be reduced by careful selection of instructions in areas of code where interrupts are expected. Using 'EI' followed immediately by a long instruction (e.g. MUL, NORML, etc.) will increase the maximum latency by 4 state times, as an interrupt cannot occur between EI and the instruction following EI. The DI, PUSHF, POPF, PUSHA, POPA and TRAP instructions will also cause the same situation. Typically these instructions would only effect latency when one interrupt routine is already in process, as these instructions are seldom used at other times.

5.5 Interrupt Summary

Many of the interrupt vectors on the 8096BH were shared by multiple interrupts. The interrupts which were shared on the 8096BH are: Transmit Interrupt, Receive Interrupt, HSI FIFO Full, Timer2 Overflow and EXTINT. On the 80C196KB, each of these interrupts have their own interrupt vectors. The source of the interrupt vectors are typically programmed through control registers. These registers can be read in Window 15 to determine the source of any interrupt. Interrupt sources with two possible interrupt vectors, serial receive interrupt sharing serial port and receive interrupt vectors for example, should be configured for only one interrupt vector.

Interrupts with separate vectors include: NMI, TRAP, Unimplemented Opcode, Timer2 Capture, 4th Entry into HSI FIFO, Software timer, HSI.0 Pin, High Speed Outputs, and A/D conversion Complete. The NMI, TRAP and Unimplemented Opcode interrupts were covered in section 5.0.

EXTINT and P0.7

The 80C196KB has two external interrupt vectors; EXTINT (200EH) and EXTINT1 (203AH). The EXTINT vector has two alternate sources selectable by IOC1.1, the external interrupt pin (Port 2.2) and Port 0.7. The external interrupt pin is the only source for the EXTINT1 interrupt vector. The external interrupt pin should not be programmed to interrupt through both vectors. Both external interrupt sources are rising edge triggered.

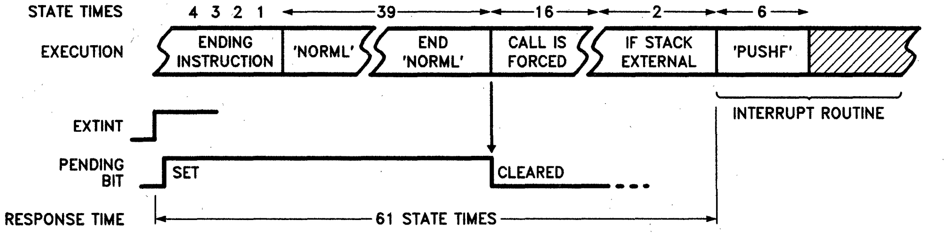


Figure 5-5. Interrupt Response Time

Serial Port Interrupts

The serial port generates one of three possible interrupts: Transmit interrupt TI(2030H), Receive Interrupt RI(2032H) and SERIAL(200CH). Refer to section 10 for information on the serial port interrupts. The 8096BH shared the TI and RI interrupts on the SERIAL interrupt vector. On the 80C196KB, these interrupts share both the serial interrupt vector and have their own interrupt vectors. Ideally, the transmit and receive interrupts should be programmed as separate interrupt vectors while disabling the SERIAL interrupt. For 8096BH compatibility, the interrupts can still use the SERIAL interrupt vector.

HSI FIFO FULL and HSI DATA AVAILABLE

HSI FIFO FULL and HSI DATA AVAILABLE interrupts shared the HSI DATA AVAILABLE interrupt vector on the 8096BH. The source of the HSI DATA AVAILABLE interrupt is controlled by the setting of I/O Control Register 1,(IOC1.7). Setting IOC1.7 to zero will generate an interrupt when a time value is loaded into the holding register. Setting the bit to one generates an interrupt when the FIFO, independent of the holding register, has six entries in it.

On the 80C196KB, separate interrupt vectors are available for the HSI FIFO FULL(203CH) and HSI DATA AVAILABLE(2004H) interrupts. The interrupts should be programmed for separate interrupt vector locations. Refer to Section 8 for more information on the High Speed Inputs.

HSI FIFO_4

The HSI FIFO can generate an interrupt when the HSI has four or more entries in the FIFO. The HSI FIFO_4 interrupt vectors through location 2034H. Refer to Section 8 for more information on the High Speed Inputs.

HSI.0 External Interrupt

The rising edge on HSI.0 pin can be used as an external interrupt. The HSI.0 pin is sampled during PH1 or CLKOUT low. Sampling is guaranteed if the pin is held for at least one state time. The interrupt vectors through location 2008H. The pin does not need to be enabled to the HSI FIFO in order to generate the interrupt.

Timer2 and Timer1 overflow

Timer2 and Timer1 can interrupt on overflow. These interrupts shared the same interrupt vector TIMER OVERFLOW(2000H) on the 8096BH. The interrupts are individually enabled by setting bits 2 and 3 of IOC1: bit 2 for Timer1, and bit 3 for Timer2. Which timer actually caused the interrupt can be determined by bits 4 and 5 of IOS1: bit 4 for Timer2 and 5 for Timer1. On the 80C196KB Timer2 overflow(0H or 8000H) has a separate interrupt vector through location 2038H.

Timer2 Capture

The 80C196KB can generate an interrupt in response to a Timer2 capture triggered by a rising edge on P2.7. Timer2 Capture vectors through location 2036H.

High Speed Outputs

The High Speed Outputs interrupt can be generated in response to a programmed HSO command which causes an external event. HSO commands which set or clear the High Speed Output pins are considered external events. Status Register IOS2 indicates which HSO events have occured and can be used to arbitrate which HSO command caused the interrupt. The High Speed Output interrupt vectors indirectly through location 2006H. For more information on High Speed Outputs, refer to Section 9.

Software Timers

HSO commands which create internal events can interrupt through the Software Timer interrupt vector. Internal events include triggering an A/D conversion, resetting Timer2 and software timers. Status registers IOS2 and IOS1 can be used to determine which internal HSO event has occured. Location 200AH is the interrupt vector for the Software Timer interrupt. Refer to Section 9 for more information on software timers and the HSO.

A/D Conversion Complete

The A/D Conversion Complete interrupt can generate an interrupt in response to a completed A/D conversion. The interrupt vectors indirectly through location 2002H. Refer to section 11 for more information on the A/D Converter.

6.0 Pulse Width Modulation Output (D/A)

Digital to analog conversion can be done with the Pulse Width Modulation output; a block diagram of the circuit is shown in Figure 6-1. The 8-bit counter is incremented every state time. When it equals 0, the PWM output is set to a one. When the counter matches the value in the PWM register, the output is switched low. When the counter overflows, the output is once again switched high. A typical output waveform is shown in

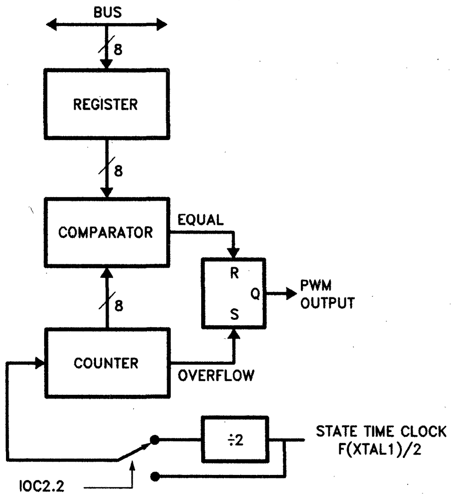
Figure 6-2. Note that when the PWM register equals 00, the output is always low. Additionally, the PWM register will only be reloaded from the temporary latch when the counter overflows. This means the compare circuit will not recognize a new value until the counter has expired preventing missed PWM edges.

The 80C196KB PWM unit has a prescaler bit (divide by 2) which is enabled by setting IOC2.2 = 1. The PWM frequencies are shown in Figure 6-3. The output waveform is a variable duty cycle pulse which repeats every 256 or 512 state times (42.75 μs or 85.5 μs at 12 MHz). Changes in the duty cycle are made by writing to the PWM register at location 17H. The value programmed into the PWM register can be read in Window 15 (WSR = 15). There are several types of motors which require a PWM waveform for more efficient operation. Additionally, if this waveform is integrated it will produce a DC level which can be changed in 256 steps by varying the duty cycle. as described in the next section.

XTAL1 =	8 MHz	10 MHz	12 MHz
IOC2.2 = 0	15.6 KHz	19.6 KHz	23.6 KHz
IOC2.2 = 1	7.8 KHz	9.8 KHz	11.8 KHz

Figure 6-3. PWM Frequencies

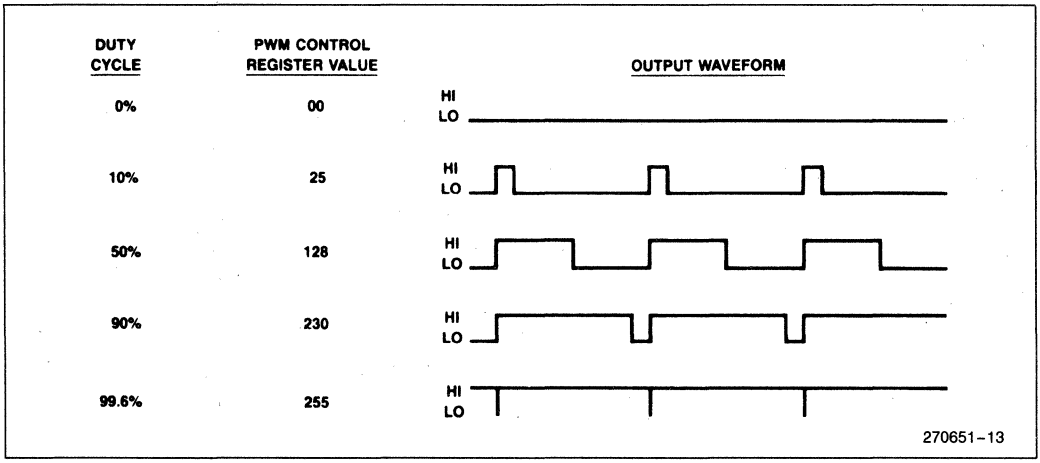
The PWM output shares a pin with Port 2, pin 5 so that these two features cannot be used at the same time. IOC1.0 equal to 1 selects the PWM function instead of the standard port function.



270651-12

- Duty Cycle Programmable in 256 Steps

Figure 6-1. PWM Block Diagram



270651-13

Figure 6-2. Typical PWM Outputs

6.1 Analog Outputs

Analog outputs can be generated by two methods, either by using the PWM output or the HSO. See Section 9.7 for information on generating a PWM with the High Speed Output Unit. Either device will generate a rectangular pulse train that varies in duty cycle and period. If a smooth analog signal is desired as an output, the rectangular waveform must be filtered.

In most cases this filtering is best done after the signal is buffered to make it swing from 0 to 5 volts since both of the outputs are guaranteed only to low current levels. A block diagram of the type of circuit needed is shown in Figure 6-4. By proper selection of components, accounting for temperature and power supply drift, a highly accurate 8-bit D to A converter can be made using either the HSO or the PWM output. Figure 6-5 shows two typical circuits. If the HSO is used the accuracy could be theoretically extended to 16-bits, however the temperature and noise related problems would be extremely hard to handle.

When driving some circuits it may be desirable to use unfiltered Pulse Width Modulation. This is particularly true for motor drive circuits. The PWM output can generate these waveforms if a fixed period on the order of 64 μs is acceptable. If this is not the case then the HSO unit can be used. The HSO can generate a variable waveform with a duty cycle variable in up to 65536 steps and a period of up to 87.5 milliseconds. Both of these outputs produce CHMOS levels.

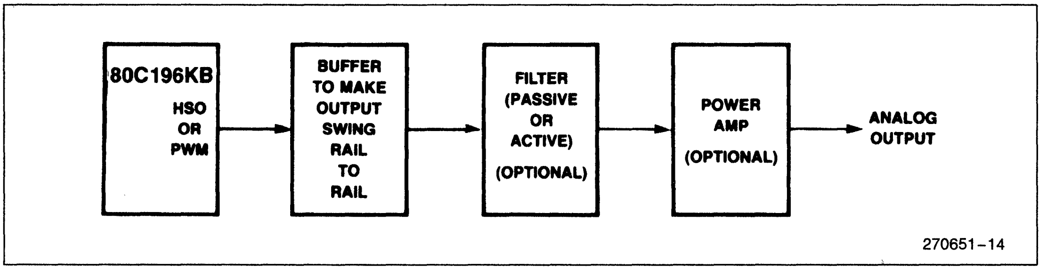


Figure 6-4. D/A Buffer Block Diagram

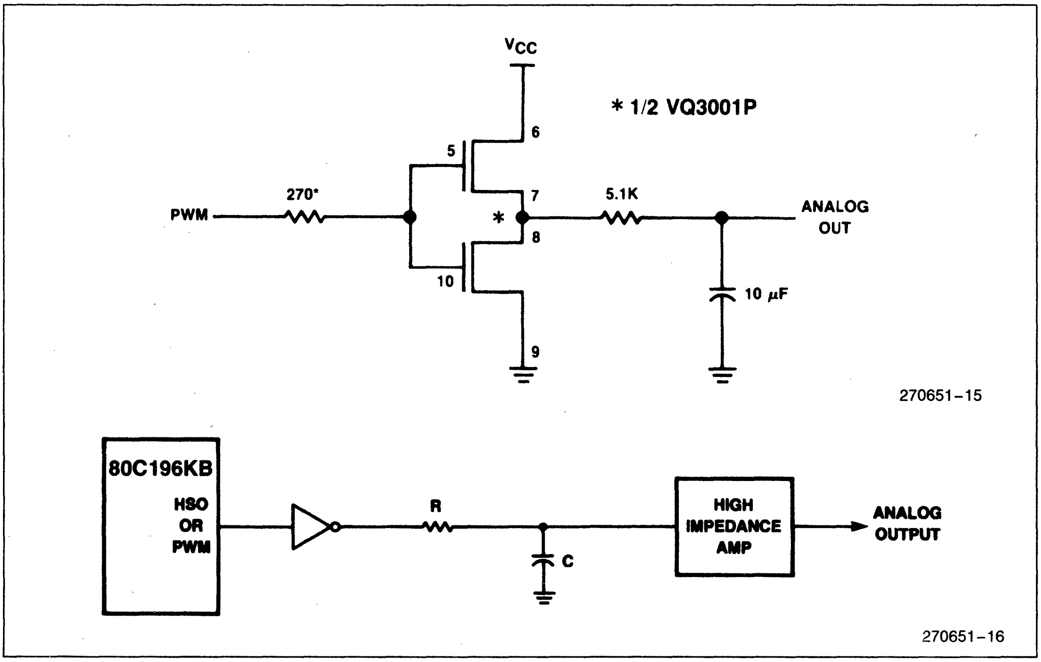


Figure 6-5. Buffer Circuits for D/A

7.0 TIMERS

7.1 Timer1

Timer1 is a 16-bit free-running timer which is incremented every eight state times. An interrupt can be generated in response to an overflow. It is read through location 0AH in Window 0 and written in Window 15. Figure 7-1 shows a block diagram of the timers.

Care must be taken when writing to it if the High Speed I/O (HSIO) Subsystem is being used. HSO time entries in the CAM depend on exact matches with Timer1. Writes to Timer1 should be taken into account in software to ensure events in the HSO CAM are not missed or occur in an order which may be unexpected. Changing Timer1 with incoming events on the High Speed Input lines may corrupt relative references between captured inputs. Further information on the High Speed Outputs and High Speed Inputs can be found in Sections 8 and 9 respectively.

7.2 Timer2

Timer2 on the 80C196KB can be used as an external reference for the HSO unit, an up/down counter, an external event capture or as an extra counter. Timer2 is clocked externally using either the T2CLK pin or the HSI.1 pin depending on the state of IOC0.7. Timer 2 counts both positive and negative transitions. The maximum transition speed is once per state time in the Fast Increment mode, and once every 8 states otherwise. Timer2 can be read and written through location 0CH in Window 0. Figure 7-1 shows a block diagram of the timers.

Timer2 can be reset by hardware, software or the HSO unit. Either T2RST or HSI.0 can reset Timer2 externally depending on the setting of IOC0.5. Figure 7-2 shows the configuration and input pins of Timer2. Figure 7-3 shows the reset and clocking options for Timer2. The appropriate control registers can be read in Window 15 to determine the programmed modes. However, IOC0.1(T2RST) is not latched and will read a 1.

Caution should be used when writing to the timers if they are used as a reference to the High Speed Output Unit. Programmed HSO commands could be missed if the timers do not count continuously in one direction. High Speed Output events based on Timer2 must be carefully programmed when using Timer2 as an up/down counter or is reset externally. Programmed events could be missed or occur in the wrong order. Refer to section 9 for more information on using the timers with the High Speed Output Unit.

Capture Register

The value in Timer2 can be captured into the T2CAPture register by a rising edge on P2.7. The edge must be held for at least one state time as discussed in the next section. T2CAP is located at 0CH in Window 15. The interrupt generated by a capture vectors through location 2036H.

Fast Increment Mode

Timer2 can be programmed to run in fast increment mode to count transitions every state time. Setting IOC2.0 programs Timer2 in the Fast Increment mode. In this mode, the events programmed on the HSO unit with Timer2 as a reference will not execute properly since the HSO requires eight state times to compare every location in the HSO CAM. With Timer2 as a reference for the HSO unit, Timer2 transitioning every state time may cause programmed HSO events to be missed. For this reason, Timer2 should not be used as a reference for the HSO if transitions occur faster than once every eight state times.

Timer2 should not be RESET in the fast increment mode. All Timer2 resets are synchronized to an eight state time clock. If Timer2 is reset when clocking faster than once every 8 states, it may reset on a different count.

Up/Down Counter Mode

Timer2 can be made to count up or down based on the Port 2.6 pin if IOC2.1 = 1. However, caution must be used when this feature is working in conjunction with the HSO. If Timer2 does not complete a full cycle it is possible to have events in the CAM which never match the timer. These events would stay in the CAM until the CAM is cleared or the chip is reset.

7.3 Sampling on External Timer Pins

The T2UP/DN, T2CLK, T2RST, and T2CAP pins are sampled during PH1. PH1 roughly corresponds to CLKOUT low externally. For valid sampling, the inputs should be present 30 nsec prior to the rising edge of CLKOUT or it may not be sampled until the next CLKOUT. To synchronize the inputs, the rising edge of CLKOUT should latch the inputs and hold them until the next rising edge of CLKOUT. T2UP/DN and T2CLK needs to be synchronized unless they never transition within one state time of each other. Otherwise, Timer2 may count in the wrong direction.

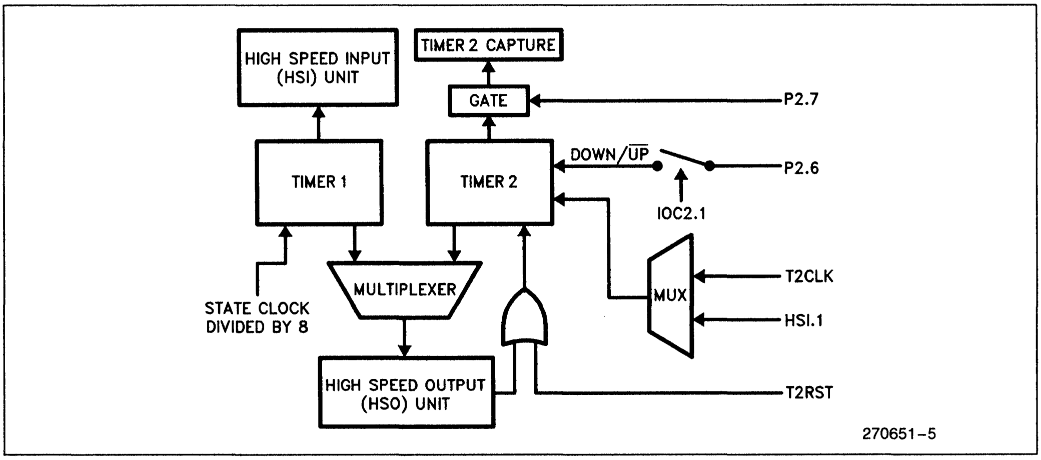


Figure 7-1. Timer Block Diagram

	Bit = 1	Bit = 0
IOC0.1	Reset Timer2 each write	No action
IOC0.3	Enable external reset	Disable
IOC0.5	HSI.0 is ext. reset source	T2RST is reset source
IOC0.7	HSI.1 is T2 clock source	T2CLK is clock source
IOC1.3	Enable Timer2 overflow int.	Disable overflow interrupt
IOC2.0	Enable fast increment	Disable fast increment
IOC2.1	Enable downcount feature	Disable downcount
P2.6	Count down if IOC2.1 = 1	Count up
IOC2.5	Interrupt on 7FFFH/8000H	Interrupt on 0FFFFH/0000H
P2.7	Capture Timer2 into T2CAPture on rising edge	

Figure 7-2. Timer2 Configuration and Control Pins

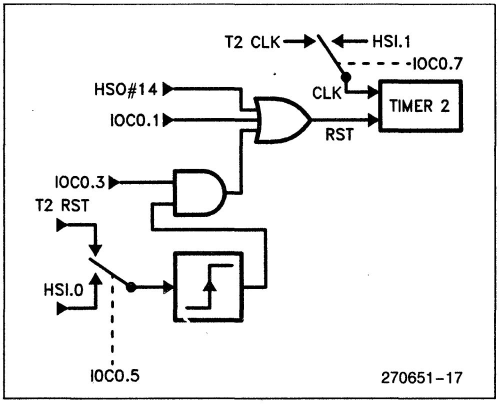


Figure 7-3. Timer2 Clock and Reset Options

7.4 Timer Interrupts

Both Timer1 and Timer2 can trigger a timer overflow interrupt and set a flag in the I/O Status Register 1 (IOS1). Timer1 overflow is controlled by setting IOC1.2 and the interrupt status is indicated in IOS1.5. The TIMER OVERFLOW interrupt is enabled by setting INT_MASK.0.

A Timer2 overflow condition interrupts through location 2000H by setting IOC1.3 and setting INT_MASK.0. Alternatively, Timer2 overflow can interrupt through location 2038H by setting INT_MASK1.3. The status of the Timer2 overflow interrupt is indicated in IOS1.4.

Interrupts can be generated if Timer2 crosses the 0FFFFH/0000H boundary or the 7FFFH/8000H boundary in either direction. By having two interrupt points it is possible to have interrupts enabled even if

Timer2 is counting up and down centered around one of the interrupt points. The boundaries used to control the Timer2 interrupt is determined by the setting of IOC2.5. When set, Timer2 will interrupt on the 7FFFH/8000H boundary, otherwise, the 0FFFFH/0000H boundary interrupts.

A T2CAPTURE interrupt is enabled by setting INT_MASK1.3. The interrupt will vector through location 2036H.

Caution must be used when examining the flags, as any access (including Compare and Jump on Bit) of IOS1 clears bits 0 through 5 including the software timer flags. It is, therefore, recommended to copy the byte to a temporary register before testing bits. Writing to IOS1 in Window 15 will set the status bits but not cause interrupts. The general enabling and disabling of the timer interrupts are controlled by the Interrupt Mask Register bit 0. In all cases, setting a bit enables a function, while clearing a bit disables it.

8.0 HIGH SPEED INPUTS

The High Speed Input Unit (HSI) can record the time an event occurs with respect to Timer1. There are 4 lines (HSI.0 through HSI.3) which can be used in this mode and up to a total of 8 events can be recorded. HSI.2 and HSI.3 are bidirectional pins which can also be used as HSO.4 and HSO.5. The I/O Control Registers (IOC0 and IOC1) determine the functions of these pins. The values programmed into IOC0 and IOC1 can be read in Window 15. A block diagram of the HSI unit is shown in Figure 8-1.

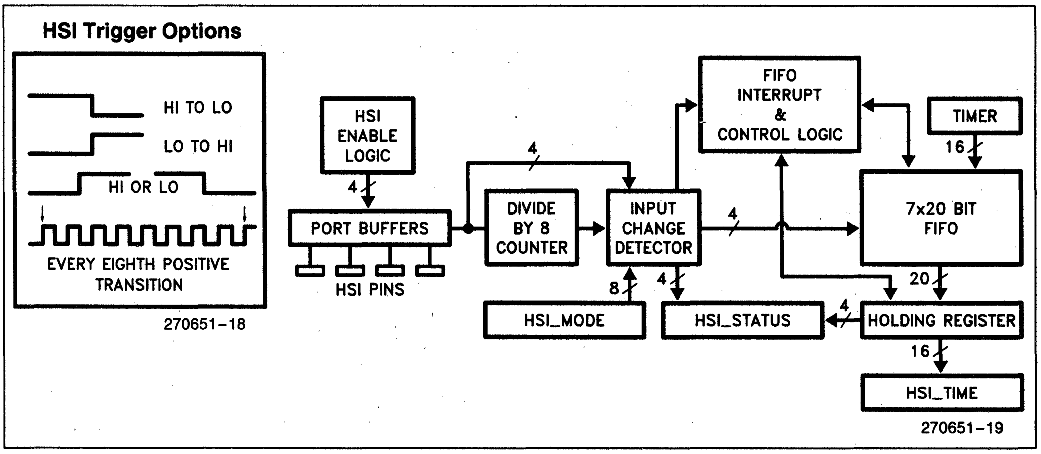
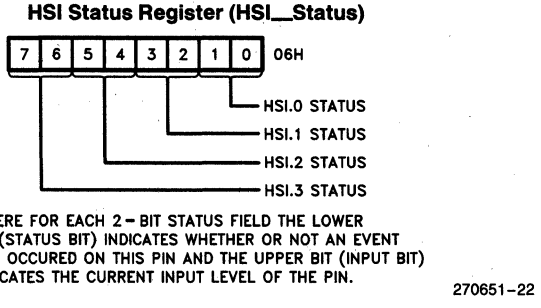


Figure 8-1. High Speed Input Unit



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT (STATUS BIT) INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT (INPUT BIT) INDICATES THE CURRENT INPUT LEVEL OF THE PIN.

Figure 8-2. HSI Status Register Diagram

When an HSI event occurs, a 7×20 FIFO stores the 16 bits of Timer1, and the 4 bits indicating which pins recorded events associated with that time tag. Therefore, if multiple pins are being used as HSI inputs, software must check each status bits when processing on HSI event. Multiple pins can recognize events with the same time tag. It can take up to 8 state times for this information to reach the holding register. For this reason, 8 state times must elapse between consecutive reads of HSI__TIME. When the FIFO is full, one additional event, for a total of 8 events, can be stored by considering the holding register part of the FIFO. If the FIFO and holding register are full, any additional events will not be recorded.

8.1 HSI Modes

There are 4 possible modes of operation for each of the HSI pins. The HSI__MODE register at location 03H controls which pins will look for what type of events. In Window 15, reading the register will read back the programmed HSI mode. The 8-bit register is set up as shown in Figure 8-3.

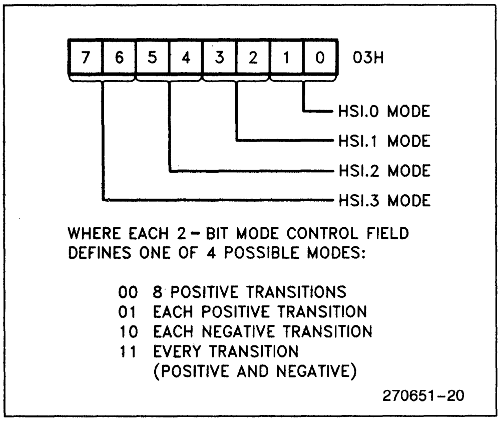

7
6
5
4
3
2
1
0
03H
HSI.0 MODE
HSI.1 MODE
HSI.2 MODE
HSI.3 MODE
WHERE EACH 2-BIT MODE CONTROL FIELD
DEFINES ONE OF 4 POSSIBLE MODES:
00 8 POSITIVE TRANSITIONS
01 EACH POSITIVE TRANSITION
10 EACH NEGATIVE TRANSITION
11 EVERY TRANSITION
(POSITIVE AND NEGATIVE)
270651-20

Figure 8-3. HSI Mode Register 1

The maximum input speed is 1 event every 8 state times except when the 8 transition mode is used, in which case it is 1 transition per state time.

The HSI pins can be individually enabled and disabled using bits in IOC0 as shown in Figure 8-4. If the pin is disabled, transitions are not entered in the FIFO. However, the input bits of the HSI__STATUS register (Figure 8-2) are always valid regardless of whether the pin is enabled to the FIFO. This allows the HSI pins to be used as general purpose input pins.

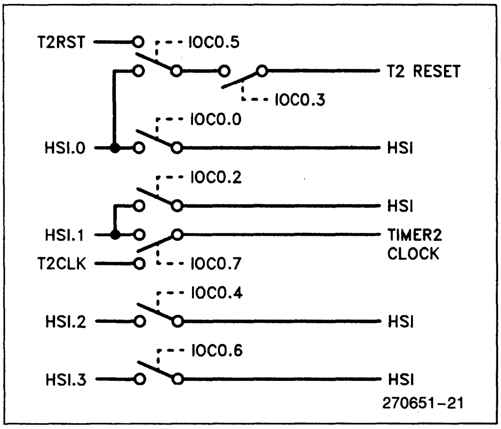


Figure 8-4. IOC0 Control of HSI Pin Functions

8.2 HSI Status

Bits 6 and 7 of the I/O Status Register 1 (IOS1—see Figure 8-5) indicate the status of the HSI FIFO. If bit 7 is set, the HSI holding register is loaded. The FIFO may or may not contain 1-5 events. If bit 6 is set, the FIFO contains 6 entries. If the FIFO fills, future events will not be recorded. Reading IOS1 clears bits 0-5, so keep an image of the register and test the image to retain all 6 bits.

Reading the HSI holding register must be done in a certain order. The HSI__STATUS Register (Figure 8-2) is read first to obtain the status and input bits. Second, the HSI__TIME Register (04H) is read to obtain the time tag. Reading HSI__TIME unloads one level of the FIFO. If the HSI__TIME is read before HSI__STATUS, the contents of HSI__STATUS associated with that HSI__TIME tag are lost.

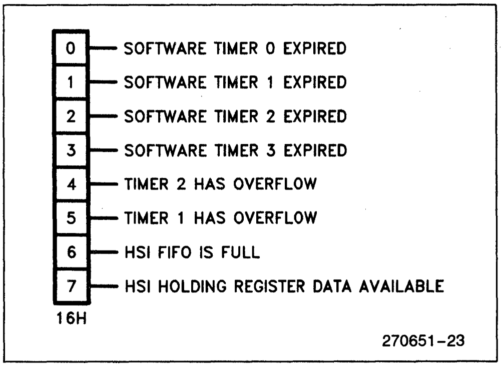

0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE
16H
270651-23

Figure 8-5. I/O Status Register 1

If the HSI_TIME register is read without the holding register being loaded, the returned value will be indeterminate. Under the same conditions, the four bits in HSI_STATUS indicating which events have occurred will also be indeterminate. The four HSI_STATUS bits which indicate the current state of the pins will always return the correct value.

It should be noted that many of the Status register conditions are changed by a reset, see section 13. Writing to HSI_TIME in window 15 will write to the HSI FIFO holding register. Writing to HSI_STATUS in Window 15 will set the status bits but will not affect the input bits.

8.3 HSI Interrupts

Interrupts can be generated by the HSI unit in three ways: when a value moves from the FIFO into the holding register; when the FIFO (independent of the holding register) has 4 or more event stored; when the FIFO has 6 or more events.

The HSI DATA AVAILABLE and HSI FIFO FULL interrupts are shared on the 8096BH. The source for the HSI DATA AVAILABLE interrupt is controlled by IOC1.7. When IOC1.7 is cleared, the HSI will generate an interrupt when the holding register is loaded. The interrupt indicates at least one HSI event has occurred and is ready to be processed. The interrupt vectors through location 2004H. The interrupt is enabled by setting INT_MASK.2. The generation of a HSI DATA AVAILABLE interrupt will set IOS1.7. The HSI FIFO FULL interrupt will vector through HSI DATA AVAILABLE if IOC1.7 is set. On the 80C196KB, the HSI FIFO FULL has a separate interrupt vector at location 203CH.

A HSI FIFO FULL interrupt occurs when the HSI FIFO has six or more entries loaded independent of the holding register. Since all interrupts are rising edge triggered, the processor will not be reinterrupted until the FIFO first contains 5 or less records, then contains six or more. The HSI FIFO FULL interrupt mask bit is INT_MASK1.6. The occurrence of a HSI FIFO FULL interrupt is indicated by IOS1.6. Earlier warning of a impending FIFO full condition can be achieved by the HSI FIFO 4th Entry interrupt.

The HSI_FIFO_4 interrupt generates an interrupt when four or more events are stored in the HSI FIFO independent of the holding register. The interrupt is enabled by setting INT_MASK1.2. The HSI_FIFO_4 vectors indirectly through location 2034H. There is no status flag associated with the HSI_FIFO_4 interrupt since it has its own independent interrupt vector.

The HSI.0 pin can generate an interrupt on the rising edge even if its not enabled to the HSI FIFO. An interrupt generated by this pin vectors through location 2008H.

8.4 HSI Input Sampling

The HSI pins are sampled internally once each state time. Any value on these pins must remain stable for at least 1 full state time to guarantee that it is recognized. The actual sampling occurs during PH1 or during CLKOUT low. The HSI inputs should be valid at least 30 nsec before the rising of CLKOUT. Otherwise, the HSI input may be sampled in the next CLKOUT. Therefore, if information is to be synchronized to the HSI it should be latched on the rising edge of CLKOUT.

8.5 Initializing the HSI

When starting the HSI, two things need to be done. The FIFO should first be flushed and the HSI initialized. The FIFO should be flushed to clear out any pending events. The following section of code can be used to flush the FIFO:

```
reflush: ld 0, HSI_TIME ;clear an event
         skip           ;wait 8 state times
         skip
         jbs IOS1, 7, reflush
```

When initializing the HSI, interrupt(s) need to be enabled and the HSI pins need to be individually enabled to the FIFO through IOC0. It is very important to initialize the interrupts before the HSI pins or a FIFO lockout condition could occur. For example, if the HSI pins were enabled first, an event could get loaded into the holding register before the HSI_DATA_AVAILABLE interrupt is enabled. If this happens, no HSI_DATA_AVAILABLE interrupts will ever occur.

9.0 HIGH SPEED OUTPUTS

The High Speed Output unit (HSO) trigger events at specific times with minimal CPU overhead. Events are generated by writing commands to the HSO_COMMAND register and the relative time at which the events are to occur into the HSO_TIME register. In Window 15, these registers will read the last value programmed in the holding register. The programmable events include: starting an A/D conversion, resetting Timer2, setting 4 software flags, and switching 6 output lines (HSO.0 through HSO.5). The format of the HSO_COMMAND register is shown in Figure 9-1. Commands 0CH and 0DH are reserved for use on future products. Up to eight events can be pending at one time and interrupts can be generated whenever any of these events are triggered. HSO.4 and HSO.5 are bi-

	7	6	5	4	3	2	1	0	
HSO_ COMMAND	CAM LOCK	TMR2/ $\overline{\text{TMR1}}$	SET/ CLEAR	INT/ $\overline{\text{INT}}$	CHANNEL				06H

CAM Lock — Locks event in CAM if this is enabled by IOC2.6 (ENA_LOCK)

TMR/$\overline{\text{TMR1}}$ — Events Based on Timer2/Based on Timer1 if 0

SET/$\overline{\text{CLEAR}}$ — Set HSO pin/Clear HSO pin if 0

INT/$\overline{\text{INT}}$ — Cause interrupt/No interrupt if 0

CHANNEL: (in Hex)		
	0–5:	HSO pins 0–5 separately
	6:	HSO pins 0 and 1 together
	7:	HSO pins 2 and 3 together
	8–B:	Software Timers 0–3
	C–D:	Unflagged Events (Do not use for future compatibility)
	E:	Reset Timer2
	F:	Start A to D Conversion

Figure 9-1. HSO Command Register

directional pins which are multiplexed with HSI.2 and HSI.3 respectively. Bits 4 and 6 of I/O Control Register 1 (IOC1.4, IOC1.6) enable HSO.4 and HSO.5 as outputs. The Control Registers can be read in Window 15 to determine the programmed modes for the HSO. However, the IOC2.7(CAM CLEAR) bit is not latched and will read as a one. Entries can be locked in the CAM to generate periodic events or waveforms.

9.1 HSO Interrupts and Software Timers

The HSO unit can generate two types of interrupts. The High Speed Output execution interrupt can be generated (if enabled) for HSO commands which change one or more of the six output pins. The other HSO interrupt is the interrupt which can be generated by any other HSO command, (e.g. triggering the A/D, resetting Timer2 or generating a software time delay).

HSO Interrupt Status

Register IOS2 at location 17H displays the HSO events which have occurred. IOS2 is shown in Figure 9-2. The events displayed are HSO.0 through HSO.5, Timer2 Reset and start of an A/D conversion. IOS2 is cleared when accessed, therefore, the register should be saved in an image register if more than one bit is being tested. The status register is useful in determining which events have caused an HSO generated interrupt. Writing to this register in Window 15 will set the status bits but not cause interrupts. In Window 15, writing to IOS2 can set the High Speed Output lines to an initial value. Refer to Section 15.2 for more information on Window 15.

IOS2:	7	6	5	4	3	2	1	0
	START A/D	T2 RESET	HSO.5	HSO.4	HSO.3	HSO.2	HSO.1	HSO.0

17H
read

Indicates which HSO event occcured

START A/D: HSO_CMD 15, start A/D

T2RESET: HSO_CMD 14, Timer2 Reset

HSO.0–5: Output pins HSO.0 through HSO.5

Figure 9-2. I/O Status Register 2

SOFTWARE TIMERS

The HSO can be programmed to generate interrupts at preset times. Up to four such "Software Timers" can be in operation at a time. As each preprogrammed time is reached, the HSO unit sets a Software Timer Flag. If the interrupt bit in the HSO command register was set then a Software Timer Interrupt will also be generated. The interrupt service routine can then examine I/O Status register 1 (IOS1) to determine which software timer expired and caused the interrupt. When the HSO resets Timer2 or starts an A/D conversion, it can also be programmed to generate a software timer interrupt.

If more than one software timer interrupt occurs in the same time frame, multiple status bits will be set. Each read or test of any bit in IOS1 (see Figure 9-5) will clear bits 0 through 5. Be certain to save the byte before testing it unless you are only concerned with 1 bit. See also Section 11.5.

9.2 HSO CAM

A block diagram of the HSO unit is shown in Figure 9-3. The Content Addressable Memory (CAM) file is the center of control. One CAM register is compared with the timer values every state time, taking 8 state times to compare all CAM registers with the timers. This defines the time resolution of the HSO to be 8 state times (1.33 microseconds at an oscillator frequency of 12 MHz).

Each CAM register is 24 bits wide. Sixteen bits specify the time at which the action is to be carried out, one bit for the lock bit and 7 bits specify both the nature of the action and whether Timer1 or Timer2 is the reference. The format of the command to the HSO unit is shown in Figure 9-1. Note that bit 5 is ignored for command channels 8 through 0FH.

To enter a command into the CAM file, write the 8-bit "Command Tag" into location 0006H followed by the time the action is to be carried out into word address 0004H. The typical code would be:

```
LDB HSO_COMMAND,#what_to_do
ADD HSO_TIME,Timer1,#when_to_do_it
```

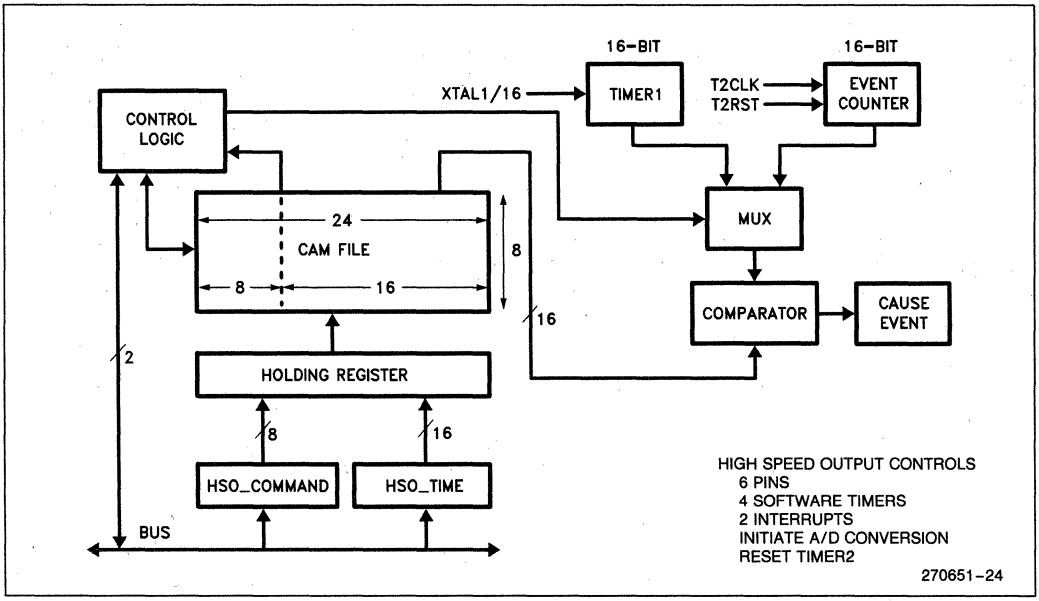


Figure 9-3. High Speed Output Unit

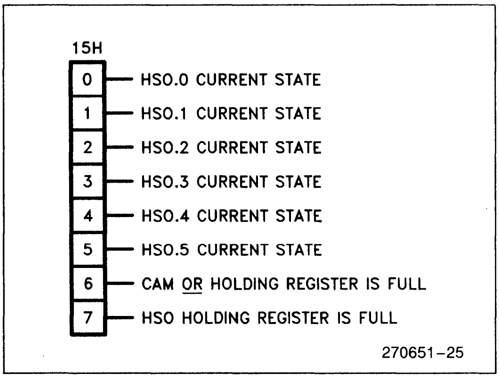


Figure 9-4. I/O Status Register 0

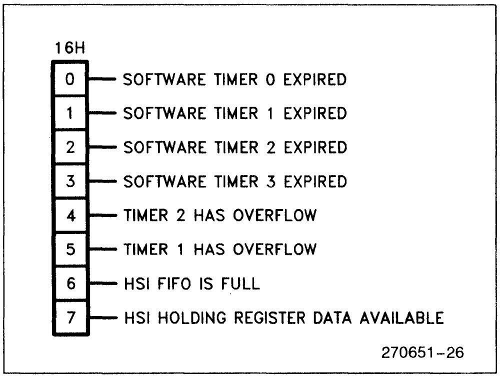

16H
0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE
270651-26

Figure 9-5. I/O Status Register 1 (IOS1)

Writing the time value loads the HSO Holding Register with both the time and the last written command tag. The command does not actually enter the CAM file until an empty CAM register becomes available.

Commands in the holding register will not execute even if their time tag is reached. Commands must be in the CAM to execute. Commands in the holding register can also be overwritten. Since it can take up to 8 state times for a command to move from the holding register to the CAM, 8 states must be allowed between successive writes to the CAM.

To provide proper synchronization, the minimum time that should be loaded to Timer1 is Timer1 + 2. Smaller values may cause the Timer match to occur 65,636 counts later than expected. A similar restriction applies if Timer2 is used.

Care must be taken when writing the command tag for the HSO, because an interrupt can occur between writing the command tag and loading the time value. If the interrupt service routine writes to the HSO, the command tag used in the interrupt routine will overwrite the command tag from the main routine. One way of avoiding this problem would be to disable interrupts when writing to the HSO unit.

9.3 HSO Status

Before writing to the HSO, it is desirable to ensure that the Holding Register is empty. If it is not, writing to the HSO will overwrite the value in the Holding Register. I/O Status Register 0 (IOS0) bits 6 and 7 indicate the status of the HSO unit. If IOS0.6 equals 0, the holding register is empty and at least one CAM register is empty. If IOS0.7 equals 0, the holding register is empty. The programmer should carefully decide which of these two flags is the best to use for each application. This register also shows the current status of the HSO.0 through HSO.5. The HSO pins can be set by writing to this register in Window 15. The format for I/O Status Register 0 is shown in Figure 9-4.

The expiration of software timer 0 through 4, and the overflow of Timer1 and Timer2 are indicated in IOS1. The status bits can be set in Window 15 but not cause interrupts. The register is shown in Figure 9-5.

Whenever the processor reads this register all of the time-related flags (bits 5 through 0) are cleared. This applies not only to explicit reads such as:

```
LDB   AL,IOS1
```

but also to implicit reads such as:

```
JBS    IOS1,3,somewhere_else
```

which jumps to somewhere_else if bit 3 of IOS1 is set. In most cases this situation can best be handled by having a byte in the register file which maintains an image of the register. Any time a hardware timer interrupt or a HSO software timer interrupt occurs the byte can be updated:

```
ORB   IOS1_image,IOS1
```

leaving IOS1_image containing all the flags that were set before plus all the new flags that were read and cleared from IOS1. Any other routine which needs to sample the flags can safely check IOS1_image. Note that if these routines need to clear the flags that they have acted on, then the modification of IOS1_image must be done from inside a critical region.

9.4 Clearing the HSO and Locked Entries

All 8 CAM locations of the HSO are compared before any action is taken. This allows a pending external event to be cancelled by simply writing the opposite event to the CAM. However, once an entry is placed in the CAM, it cannot be removed until either the specified timer matches the written value , a chip reset occurs or IOC2.7 is set. IOC2.7 is the CAM clear bit which clears all entries in the CAM.

Internal events cannot be cleared by writing an opposite event. This includes events on HSO channels 8 through F. The only method for clearing these events are by a reset or setting IOC2.7.

HSO LOCKED ENTRIES

The CAM Lock bit (HSO_Command.7) can be set to keep commands in the CAM, otherwise the commands will clear from the CAM as soon as they cause an event. This feature allows for generation periodic events based on Timer2 and must be enabled by setting IOC2.6. To clear locked events from the CAM, the entire CAM can be cleared by writing a one to the CAM clear bit IOC2.7. A chip reset will also clear the CAM.

Locked entries are useful in applications requiring periodic or repetitive events to occur. Timer2 used as an HSO reference can generate periodic events with the use of the HSO T2RST command. HSO events programmed with a HSO time less then the Timer2 reset time will occur repeatedly as Timer2 resets. Recurrent software tasks can be scheduled by locking software timers commands into the High Speed Output Unit. Continuous sampling of the A/D converter can be accompished by programming a locked HSO A/D conversion command. One of the most useful features is the generation of multiple PWM's on the High Speed Output lines. Locked entries provide the ability to program periodic events while minimizing the software overhead. Section 9.6 describes the generation of four PWMs using locked entries.

Individual external events setting or clearing an HSO pin can by cancelled by writing the opposite event to the CAM. The HSO events do not occur until the timer reference has changed state. An event programmed to set and clear an HSO event at the same time will cancel each other out. Locked entries can correspondingly be cancelled using this method. However, the entries remain in the HSO CAM and can quickly fill up the available eight locations. As an alternative, all entries in the HSO CAM can be cleared by setting IOC2.7.

9.5 HSO Precautions

Timer1 is incremented once every 8 state-times. When it is being used as the reference timer for an HSO command, the comparator has a chance to look at all 8 CAM registers before Timer1 changes its value. Writing to Timer1, which is allowed in Window 15, should be carefully done. The user should ensure writing to Timer1 will not cause programmed HSO events to be missed or occur in the wrong order. The same precaution applies to Timer2.

The HSO requires at least eight state times to compare each entry in the CAM. Therefore, the fast increment mode for Timer2 cannot be used as a reference for the HSO if transitions occur faster then once every eight state times.

Referencing events when Timer2 is being used as an up/down counter could cause events to occur in opposite order or be missed entirely. Additionally, locked entries could possibly occur several times if Timer2 is oscillating around the time tag for an entry.

When using Timer2 as the HSO reference, caution must be taken that Timer2 is not reset prior to the highest value for a Timer2 match in the CAM. If that match is never reached, the event will remain pending in the CAM until the part is reset or CAM is cleared.

9.6 PWM Using the HSO

The HSO unit can generate PWM waveforms with very little CPU overhead using Timer2 as a reference. A PWM is generated by programming an HSO line to a high and a T2RST to occur at the same time. An HSO low time is programmed on the CAM to generate the duty cycle of the PWM. A repetitive PWM waveform is generated by locking the commands into the CAM. Reprogramming of the duty cycle or PWM frequency can be accomplished by generating a software interrupt and reprogramming the HSO high, HSO low and T2RST commands.

Multiple PWMs can be programmed using Timer2 as a reference and locked CAM entries. Up to four PWM's can be generated by locking a PWM(High) and PWM(low) into the CAM for each HSO.0 through HSO.3. Timer2 is used as a reference and set to zero by programming a T2RST command at the same time an HSO command sets all the lines high. Two CAM entries program the four PWM (high) times by setting HSO.0/HSO.1 and HSO.2/HSO.3 high with the same command. Four entries in the CAM set each of the HSO lines low. One entry is used to reset Timer2. This method uses a total of seven CAM entries with little or no software overhead. The PWMs can change their duty cycle by reprogramming the CAM with different HSO levels.

Changing the duty cycle for each PWM requires the flushing of the CAM and reprogramming of all seven entries in the CAM. The 80C196KB can flush the entire CAM by setting bit 7 in the IOC2 register (location 16H). Each HSO(high) and HSO(low) times should be reprogrammed in addition to the Timer2 reset command. This method provides for up to four PWM's with no software overhead except when reprogramming the duty cycle of any particular PWM. The code to generate these PWMs is shown in Figure 9-6.

9.7 HSO Output Timing

Changes in the HSO lines are synchronized to either Timer1 or Timer2. All of the external HSO lines due to change at a certain value of a timer will change just after the incrementing of the timer. Internally, the timer changes every eight state times during Phase1. From an external perspective the HSO pin should change just prior to the falling edge of CLKOUT and be stable by its rising edge. Information from the HSO can be latched on the CLKOUT rising edge. Internal events also occur when the reference timer increments.

10.0 SERIAL PORT

The serial port on the 80C196KB has one synchronous and 3 asynchronous modes. The asynchronous modes

```
$include (reg196.inc)
; *********************************************************
;                                                         *
; *    GENERATION OF FOUR PWM'S USING LOCKED ENTRIES      *
;                                                         *
; *    Timer2 is used as a reference and is clocked       *
; *    externally by T2CLK.  The High Speed outputs are   *
; *    used as PWMs by programming each individual        *
; *    PWM(low) and PWM(High) time as a locked entry.     *
; *    The period of the PWM is programmed by resetting   *
; *    timer2 and setting all the HSO lines high at the   *
; *    same time.  The PWMs are reprogrammed by           *
; *    clearing the HSO CAM and reloading new values      *
; *    for the PWM period and duty cycle.                 *
;                                                         *
; *********************************************************

RSEG at 60h
pwm0timl: dsw 1
pwm1timl: dsw 1
pwm2timl: dsw 1
pwm3timl: dsw 1
PWM_period: dsw 1
temp: dsw 1

cseg at 2080h
     ld sp,#0d0h            ; initialize stack pointer
     ld PWM_period,#0f000h  ; intialize pwm period
     ld pwm0timl,#2000h     ; initialize pwm 0-3 duty cycle
     ld pwm1timl,#4000h
     ld pwm2timl,#6000h
     ld pwm3timl,#8000h
     ldb ioc2,#40h          ; Enable locked entries
     ldb ioc0,#0h           ; Enable t2clk for timer2 clock
                            ; source
     call pwm_program       ; program pwm's on CAM
here:      sjmp here        ; loop forever
```

Figure 9-6. Generating Four PWMs Using Locked Entries

```
pwm_program:
     ldb ioc2,#0c0h           ; flush entire cam
     ldb hso_command,#0ceh    ; program timer2 reset time
     ld hso_time,PWM_period
     nop                      ; delay eight state times before
     nop                      ; next load
     nop
     nop
     ldb hso_command,#0e6h    ; HSO 0/1 high, locked, timer2 as
                              ; reference
     ld hso_time,PWM_period   ; set hso_high on t2rst
     nop
     nop
     nop
     nop
     ldb hso_command,#0e7h    ; HSO 2/3 high, locked, timer2
                              ; as reference
     ld hso_time,PWM_period   ; set hso_high on t2rst
     nop
     nop
     nop
     nop
     ldb hso_command,#0c0h    ; set HSO.0 low, locked, timer2
                              ; as reference
     ld hso_time,pwm0timl     ; HSO.0 time low
     nop
     nop
     nop
     nop
     ldb hso_command,#0c1h    ; set HSO.1 low, locked, timer2
                              ; reference
     ld hso_time,pwm1timl     ; HSO.1 time low
     nop
     nop
     nop
     nop
     ldb hso_command,#0c2h    ; set HSO.2 low, locked,timer2
                              ; as reference
     ld hso_time,pwm2timl     ; HSO.2 time low
     nop
     nop
     nop
     nop
     ldb hso_command,#0c3h    ; set HSO.3 low, locked,timer2
                              ; as reference
     ld hso_time,pwm3timl     ; HSO.3 time low
     ret
     end
```

Figure 9-6. Generating Four PWMs Using Locked Entries (Continued)

are full duplex, meaning they can transmit and receive at the same time. The receiver is double buffered so that the reception of a second byte can begin before the first byte has been read. The transmitter on the 80C196KB is also double buffered allowing continuous transmissions. The port is functionally compatible with the serial port on the MCS-51 family of microcontrollers, although the software controlling the ports is different.

Data to and from the serial port is transferred through SBUF(RX) and SBUF(TX), both located at 07H. SBUF(TX) holds data ready for transmission and SBUF(RX) contains data received by the serial port. SBUF(TX) and SBUF(RX) can be read and can be written in Window 15.

Mode 0, the synchronous shift register mode, is designed to expand I/O over a serial line. Mode 1 is the standard 8 bit data asynchronous mode used for normal serial communications. Modes 2 and 3 are 9 bit data asynchronous modes typically used for interprocessor communications. Mode 2 provides monitoring of a communication line for a 1 in the 9th bit position before causing an interrupt. Mode 3 causes interrupts independant of the 9th bit value.

10.1 Serial Port Status and Control

Control of the serial port is done through the Serial Port Control (SP_CON) register shown in Figure 10-1. Writing to location 11H accesses SP_CON while reading it accesses SP_STAT. The upper 3 bits of SP_CON must be written as 0s for future compatibility. On the 80C196KB the SP_STAT register contains new bits to indicate receive Overrun Error (OE), Framing Error (FE), and Transmitter Empty (TXE). The bits which were also present on the 8096BH are the Transmit Interrupt (TI) bit, the Receive Interrupt (RI) bit, and the Received Bit 8 (RB8) or Receive Parity Error (RPE) bit. SP_STAT is read-only in Window 0 and is shown in Figure 10-1.

In all modes, the RI flag is set after the last data bit is sampled, approximately in the middle of a bit time. Data is held in the receive shift register until the last data bit is received, then the data byte is loaded into SBUF (RX). The receiver on the 80C196KB also checks for a valid stop bit. If a stop bit is not found within the appropriate time, the Framing Error (FE) bit is set.

Since the receiver is double-buffered, reception on a second data byte can begin before the first byte is read. However, if data in the shift register is loaded into SBUF (RX) before the previous byte is read, the Overflow Error (OE) bit is set. Regardless, the data in SBUF (RX) will always be the latest byte received; it will never be a combination of the two bytes. The RI, FE, and OE flags are cleared when SP_STAT is read. However, RI does not have to be cleared for the serial port to receive data.

SP_CON:	7	6	5	4	3	2	1	0	
	X	X	X	TB8	REN	PEN	M2	M1	11H

TB8 — Sets the ninth data bit for transmission. Cleared after each transmission. Not valid if parity is enabled.

REN — Enables the receiver

PEN — Enables the Parity function (even parity)

M2, M1 — Sets the mode. Mode0 = 00, Mode1 = 01, Mode2 = 10, Mode3 = 11

SP_STAT	7	6	5	4	3	2	1	0	
	RB8/ RPE	RI	TI	FE	TXE	OE	X	X	11H

RB8 — Set if the 9th data bit is high on reception (parity disabled)

RPE — Set if parity is enabled and a parity error occurred

RI — Set after the last data bit is sampled

TI — Set at the beginning of the STOP bit transmission

FE — Set if no STOP bit is found at the end of a reception

TXE — Set if two bytes can be transmitted

OE — Set if the receiver buffer is overwritten

Figure 10-1. Serial Port Control and Status Registers

The Transmitter Empty (TXE) bit is set if the transmit buffer is empty and ready to take up to two characters. TXE gets cleared as soon as a byte is written to SBUF. Two bytes may be written consecutively to SBUF if TXE is set. One byte may be written if TI alone is set. By definition, if TXE has just been set, a transmission has completed and TI will be set. The TI bit is reset when the CPU reads the SP_STAT registers.

The TB8 bit is cleared after each transmission and both TI and RI are cleared when SP_STAT read. The RI and TI status bits can be set by writing to SP_STAT in window 15 but they will not cause an interrupt. Reading of SP_CON in Window 15 will read the last value written. Whenever the TXD pin is used for the serial port it must be enabled by setting IOC1.5 to a 1. I/O control register 1 can be read in window 15 to determine the setting.

STARTING TRANSMISSIONS AND RECEPTIONS

In Mode 0, if REN = 0, writing to SBUF (TX) will start a transmission. Causing a rising edge on REN, or clearing RI with REN = 1, will start a reception. Setting REN = 0 will stop a reception in progress and inhibit further receptions. To avoid a partial or complete undesired reception, REN must be set to zero before RI is cleared. This can be handled in an interrupt environment by using software flags or in straight-line code by using the Interrupt Pending register to signal the completion of a reception.

In the asynchronous modes, writing to SBUF (TX) starts a transmission. A falling edge on RXD will begin a reception if REN is set to 1. New data placed in SBUF (TX) is held and will not be transmitted until the end of the stop bit has been sent.

In all modes, the RI flag is set after the last data bit is sampled approximately in the middle of the bit time. Also for all modes, the TI flag is set after the last data bit (either 8th or 9th) is sent, also in the middle of the bit time. The flags clear when SP_STAT is read, but do not have to be clear for the port to receive or transmit. The serial port interrupt bit is set as a logical OR of the RI and TI bits. Note that changing modes will reset the Serial Port and abort any transmission or reception in progress on the channel.

BAUD RATES

Baud rates are generated based on either the T2CLK pin or XTAL1 pin. The values used are different than those used for the 8096BH because the 80C196KB uses a divide-by-2 clock instead of a divide-by-3 clock to generate the internal timings. Baud rates are calculated using the following formulas where BAUD_REG is the value loaded into the baud rate register:

Asynchronous Modes 1, 2 and 3:

$$\text{BAUD\_REG} = \frac{\text{XTAL1}}{\text{Baud Rate} * 16} - 1 \text{ OR } \frac{\text{T2CLK}}{\text{Baud Rate} * 8}$$

Synchronous Mode 0:

$$\text{BAUD\_REG} = \frac{\text{XTAL1}}{\text{Baud Rate} * 2} - 1 \text{ OR } \frac{\text{T2CLK}}{\text{Baud Rate}}$$

The most significant bit in the baud register value is set to a one to select XTAL1 as the source. If it is a zero the T2CLK pin becomes the source. The following table shows some typical baud rate values.

BAUD RATES AND BAUD REGISTER VALUES

Baud Rate	XTAL1 Frequency		
	8.0 MHz	10.0 MHz	12.0 MHz
300	1666 / −0.02	2082 / 0.02	2499 / 0.00
1200	416 / −0.08	520 / −0.03	624 / 0.00
2400	207 / 0.16	259 / 0.16	312 / −0.16
4800	103 / 0.16	129 / 0.16	155 / 0.16
9600	51 / 0.16	64 / 0.16	77 / 0.16
19.2K	25 / 0.16	32 / 1.40	38 / 0.16

Baud Register Value / % Error

A maximum baud rate of 750 Kbaud is available in the asynchronous modes with 12 MHz on XTAL1. The synchronous mode has a maximum rate of 3.0 Mbaud with a 12 MHz clock. Location 0EH is the Baud Register. It is loaded sequentially in two bytes, with the low byte being loaded first. This register may not be loaded with zero in serial port Mode 0.

10.2 Serial Port Interrupts

The serial port generates one of three possible interrupts: Transmit Interrupt TI(2030H), Receive Interrupt RI(2032H) and SERIAL(200CH). When the RI bit gets set an interrupt is generated through either 200CH or 2032H depending on which interrupt is enabled. INT__MASK1.1 controls the serial port receive interrupt through location 2032H and INT__MASK.6 controls serial port interrupts through location 200CH. The 8096BH shared the TI and RI interrupts on the SERIAL interrupt vector. On the 80C196KB, these interrupts share both the serial interrupt vector and have their own interrupt vectors.

When the TI bit is set it can cause an interrupt through the vectors at locations 200CH or 2030. Interrupt through location 2030 is determined by INT__MASK1.0. Interrupts through the serial interrupt is controlled by the same bit as the RI interrupt(INT__MASK.6). The user should not mask off the serial port interrupt when using the double-buffered feature of the transmitter, as it could cause a missed count in the number of bytes being transmitted.

10.3 Serial Port Modes

MODE 0

Mode 0 is a synchronous mode which is commonly used for shift register based I/O expansion. In this mode the TXD pin outputs a set of 8 pulses while the RXD pin either transmits or receives data. Data is transferred 8 bits at a time with the LSB first. A diagram of the relative timing of these signals is shown in Figure 10-2. Note that this is the only mode which uses RXD as an output.

Mode 0 Timings

In Mode 0, the TXD pin sends out a clock train, while the RXD pin transmits or receives the data. Figure 10-2 shows the waveforms and timing.

In this mode the serial port expands the I/O capability of the 80C196KB by simply adding shift registers. A schematic of a typical circuit is shown in Figure 10-3. This circuit inverts the data coming in, so it must be reinverted in software.

MODE 1

Mode 1 is the standard asynchronous communications mode. The data frame used in this mode is shown in Figure 10-4. It consists of 10 bits; a start bit (0), 8 data bits (LSB first), and a stop bit (1). If parity is enabled by setting SPCON.2, an even parity bit is sent instead of the 8th data bit and parity is checked on reception.

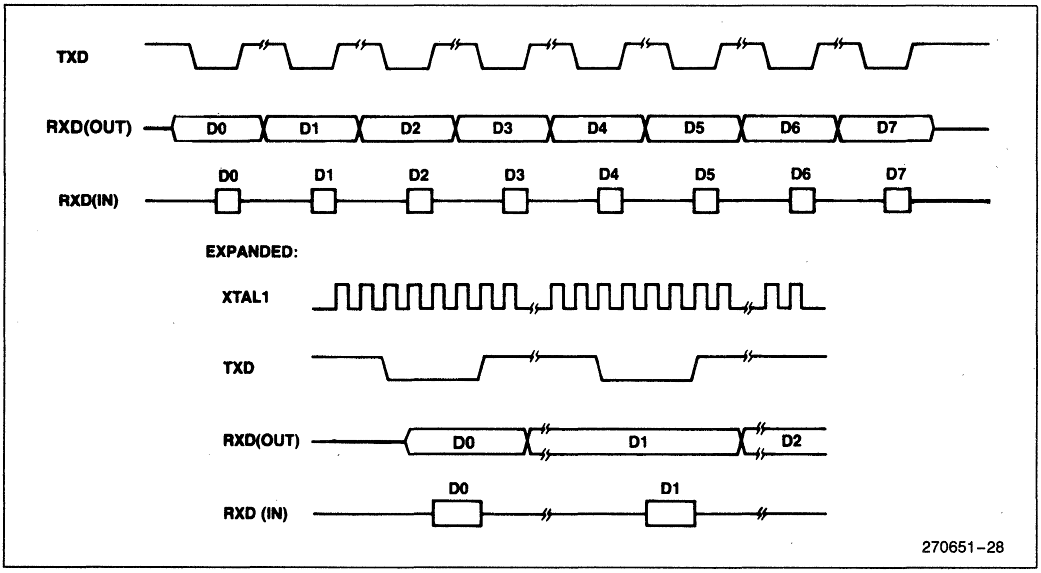


Figure 10-2. Mode 0 Timing

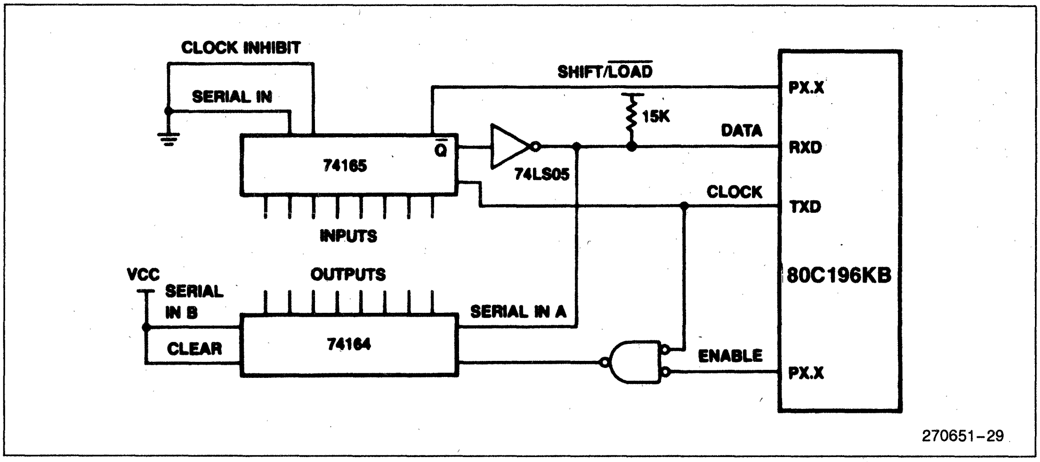


Figure 10-3. Typical Shift Register Circuit

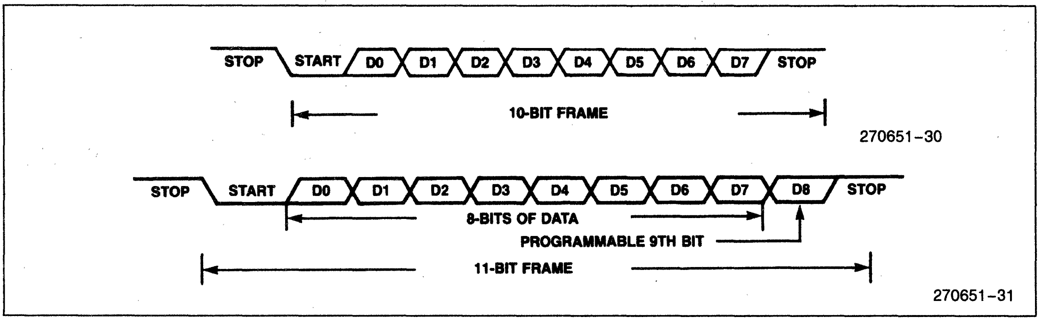


Figure 10-4. Serial Port Frames, Mode 1, 2, and 3

The transmit and receive functions are controlled by separate shift clocks. The transmit shift clock starts when the baud rate generator is initialized, the receive shift clock is reset when a '1 to 0' transition (start bit) is received. The transmit clock may therefore not be in sync with the receive clock, although they will both be at the same frequency.

The TI (Transmit Interrupt) and RI (Receive Interrupt) flags are set to indicate when operations are complete. TI is set when the last data bit of the message has been sent, not when the stop bit is sent. If an attempt to send another byte is made before the stop bit is sent the port will hold off transmission until the stop bit is complete. RI is set when 8 data bits are received, not when the stop bit is received. Note that when the serial port status register is read both TI and RI are cleared.

Caution should be used when using the serial port to connect more than two devices in half-duplex, (i.e. one wire for transmit *and* receive). If the receiving processor does not wait for one bit time after RI is set before starting to transmit, the stop bit on the link could be corrupted. This could cause a problem for other devices listening on the link.

MODE 2

Mode 2 is the asynchronous 9th bit recognition mode. This mode is commonly used with Mode 3 for multiprocessor communications. Figure 10-4 shows the data frame used in this mode. It consists of a start bit (0), 9 data bits (LSB first), and a stop bit (1). When transmitting, the 9th bit can be set to a one by setting the TB8 bit in the control register before writing to SBUF (TX). The TB8 bit is cleared on every transmission, so it must be set prior to writing to SBUF (TX). During reception, the serial port interrupt and the Receive Interrupt will not occur unless the 9th bit being received is set. This provides an easy way to have selective reception on a data link. Parity cannot be enabled in this mode.

MODE 3

Mode 3 is the asynchronous 9th bit mode. The data frame for this mode is identical to that of Mode 2. The transmission differences between Mode 3 and Mode 2 are that parity can be enabled (PEN = 1) and cause the 9th data bit to take the even parity value. The TB8 bit can still be used if parity is not enabled (PEN = 0). When in Mode 3, a reception always causes an interrupt, regardless of the state of the 9th bit. The 9th bit is stored if PEN = 0 and can be read in bit RB8. If PEN = 1 then RB8 becomes the Receive Parity Error (RPE) flag.

Mode 2 and 3 Timings

Modes 2 and 3 operate in a manner similar to that of Mode 1. The only difference is that the data is now made up of 9 bits, so 11-bit packages are transmitted and received. This means that TI and RI will be set on the 9th data bit rather than the 8th. The 9th bit can be used for parity or multiple processor communications.

10.4 Multiprocessor Communications

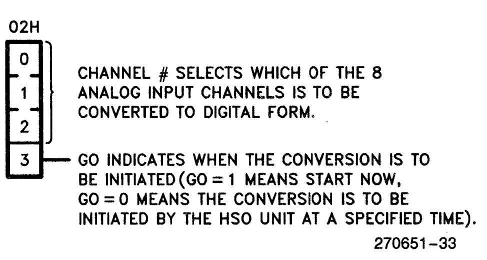
Mode 2 and 3 are provided for multiprocessor communications. In Mode 2 if the received 9th data bit is zero, the RI bit is not set and will not cause an interrupt. In Mode 3, the RI bit is set and always causes an interrupt regardless of the value in the 9th bit. The way to use this feature in multiprocessor systems is described below.

The master processor is set to Mode 3 so it always gets interrupts from serial receptions. The slaves are set in Mode 2 so they only have receive interrupts if the 9th bit is set. Two types of frames are used: address frames which have the 9th bit set and data frames which have the 9th bit cleared. When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address frame which identifies the target slave. Slaves in Mode 2 will not be interrupted by a data frame, but an address frame will interrupt all slaves. Each slave can examine the received byte and see if it is being addressed. The addressed slave switches to Mode 3 to receive the coming data frames, while the slaves that were not addressed stay in Mode 2 continue executing.

11.0 A/D CONVERTER

Analog Inputs to the 80C196KB System are handled by the A/D converter System. As shown in Figure 11-4, the converter system has an 8 channel multiplexer, a sample-and-hold, and a 10 bit successive approximation A/D converter. Conversions can be performed on one of eight channels, the inputs of which share pins with port 0. A conversion can be done in as little as 91 state times.

Conversions are started by loading the AD_COMMAND register at location 02H with the channel number. The conversion can be started immediately by setting the GO bit to a one. If it is cleared the conversion will start when the HSO unit triggers it. The result of the conversion is read in the AD_RESULT(High) and AD_RESULT(Low) registers. The AD_RESULT(High) contains the most significant eight bits of the conversion. The AD_RESULT(Low) register contains the remaining two bits and the A/D channel number and A/D status. The format for the AD_COMMAND register is shown in Figure 11-1. In Window 15, reading the AD_COMMAND register will read the last command written. Writing to the AD_RESULT register will write a value into the result register.



CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.

GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).

270651-33

Figure 11-1. A/D Command Register

The A/D converter can cause an interrupt through the vector at location 2002H when it completes a conversion. It is also possible to use a polling method by checking the Status (S) bit in the lower byte of the AD_RESULT register, also at location 02H. The status bit will be a 1 while a conversion is in progress. It takes 8 state times to set this bit after a conversion is started. The upper byte of the result register contains the most significant 8 bits of the conversion. The lower byte format is shown in Figure 11-2.

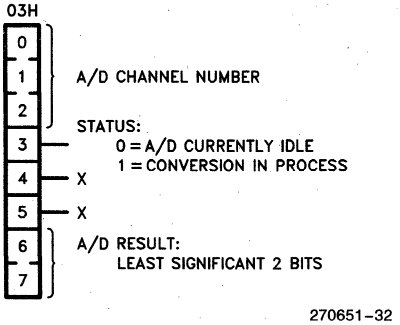


Figure 11-2. A/D Result Lo Register

At high crystal frequencies, more time is needed to allow the comparator to settle. For this reason IOC2.4 is provided to adjust the speed of the A/D conversion by disabling/enabling a clock prescaler.

A summary of the conversion time for the two options is shown below. The numbers represent the number of state times required for conversion, e.g., 91 states is 22.7 μs with an 8 MHz XTAL1 (providing a 250 ns state time.)

Clock Prescaler On IOC2.4 = 0	Clock Prescaler Off IOC2.4 = 1
158 States 26.33 μs @ 12 MHz	91 States 22.75 μs @ 8 MHz.

Figure 11-3. A/D Conversion Times

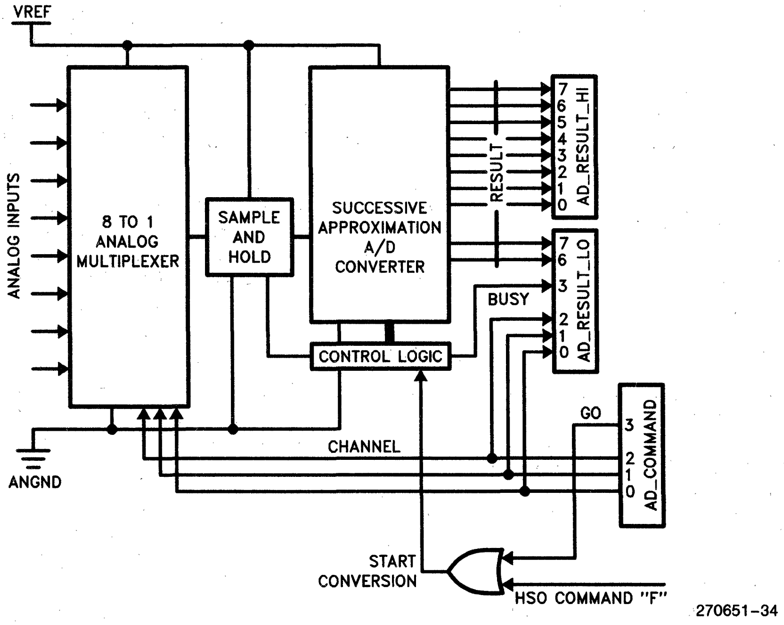


Figure 11-4. A/D Converter Block Diagram

11.1 A/D Conversion Process

The conversion process is initiated by the execution of HSO command 0FH, or by writing a one to the GO Bit in the A/D Control Register. Either activity causes a start conversion signal to be sent to the A/D converter control logic. If an HSO command was used, the conversion process will begin when Timer1 increments. This aids applications attempting to approach spectrally pure sampling, since successive samples spaced by equal Timer1 delays will occur with a variance of about ±50 ns (assuming a stable clock on XTAL1). However, conversions initiated by writing a one to the ADCON register GO Bit will start within three state times after the instruction has completed execution resulting in a variance of about 0.50 μs (XTAL1 = 12 MHz).

Once the A/D unit receives a start conversion signal, there is a one state time delay before sampling (Sample Delay) while the successive approximation register is reset and the proper multiplexer channel is selected. After the sample delay, the multiplexer output is connected to the sample capacitor and remains connected for 8 state times in fast mode or 15 state times for slow mode (Sample Time). After this 8/15 state time "sample window" closes, the input to the sample capacitor is disconnected from the multiplexer so that changes on the input pin will not alter the stored charge while the conversion is in progress. The comparator is then auto-zeroed and the conversion begins. The sample delay and sample time uncertainties are each approximately ±50 ns, independent of clock speed.

To perform the actual analog-to-digital conversion the 80C196KB implements a successive approximation algorithm. The converter hardware consists of a 256-resistor ladder, a comparator, coupling capacitors and a 10-bit successive approximation register (SAR) with logic that guides the process. The resistor ladder provides 20 mV steps (V_{REF} = 5.12V), while capacitive coupling creates 5 mV steps within the 20 mV ladder voltages. Therefore, 1024 internal reference voltages are available for comparison against the analog input to generate a 10-bit conversion result.

A successive approximation conversion is performed by comparing a sequence of reference voltages, to the analog input, in a binary search for the reference voltage that most closely matches the input. The ½ full scale reference voltage is the first tested. This corresponds to a 10-bit result where the most significant bit is zero, and all other bits are ones (0111.1111.11b). If the analog input was less than the test voltage, bit 10 of the SAR is left a zero, and a new test voltage of ¼ full scale (0011.1111.11b) is tried. If this test voltage was lower than the analog input, bit 9 of the SAR is set and bit 8 is cleared for the next test (0101.1111.11b). This binary search continues until 10 tests have occurred, at which time the valid 10-bit conversion result resides in the SAR where it can be read by software.

The total number of state times required for a conversion is determined by the setting of IOC2.4 clock prescaler bit. With the bit set the conversion time is 91 states and 158 states when the bit is cleared.

11.2 A/D Interface Suggestions

The external interface circuitry to an analog input is highly dependent upon the application, and can impact converter characteristics. In the external circuit's design, important factors such as input pin leakage, sample capacitor size and multiplexer series resistance from the input pin to the sample capacitor must be considered.

For the 80C196KB, these factors are idealized in Figure 11-5. The external input circuit must be able to charge a sample capacitor (C_S) through a series resistance (R_I) to an accurate voltage given a D.C. leakage (I_L). On the 80C196KB, C_S is around 2 pF, R_I is around 5 KΩ and I_L is specified as 3 μA maximum. In determining the necessary source impedance R_S, the value of V_{BIAS} is not important.

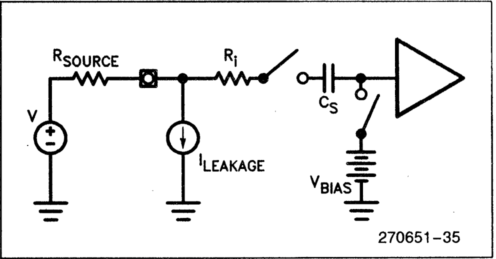


Figure 11-5. Idealized A/D Sampling Circuitry

External circuits with source impedances of 1 KΩ or less will be able to maintain an input voltage within a tolerance of about ±0.61 LSB (1.0 KΩ × 3.0 μA = 3.0 mV) given the D.C. leakage. Source impedances above 2 KΩ can result in an external error of at least one LSB due to the voltage drop caused by the 3 μA leakage. In addition, source impedances above 25 KΩ may degrade converter accuracy as a result of the internal sample capacitor not being fully charged during the 1 μs (12 MHz clock) sample window.

If large source impedances degrade converter accuracy because the sample capacitor is not charged during the sample time, an external capacitor connected to the pin compensates for this. Since the sample capacitor is 2 pF, a 0.005 μF capacitor (2048 * 2 pF) will charge the sample capacitor to an accurate input voltage of ±0.5 LSB. An external capacitor does not compensate for the voltage drop across the source resistance, but charges the sample capacitor fully during the sample time.

Placing an external capacitor on each analog input will also reduce the sensitivity to noise, as the capacitor combines with series resistance in the external circuit to form a low-pass filter. In practice, one should include a small series resistance prior to the external capacitor on the analog input pin and choose the largest capacitor value practical, given the frequency of the signal being converted. This provides a low-pass filter on the input, while the resistor will also limit input current during over-voltage conditions.

Figure 11-6 shows a simple analog interface circuit based upon the discussion above. The circuit in the figure also provides limited protection against over-voltage conditions on the analog input. Should the input voltage inappropriately drop significantly below ground, diode D2 will forward bias at about 0.8 DCV. The equivalent circuit to Figure 11-6 essentially sets up an RC charge circuit with one time constant equal to 1.35 μs (270Ω × 0.005 μF). The charge on this circuit represents the amount of voltage that will appear at the input pin of the MCS-96 converter. Momentary transitions below ground are not steady state conditions. The instantaneous charge will be 0V followed in time by increasingly negative voltage. Given the time constant discussed earlier, and assuming a transition below ground of −0.8V, the MCS-96 specification of −0.3V will not be exceeded for a duration of approximately 700 ns.

This will effectively limit the voltage, and therefore the current through the pin, to a safe level for brief periods of time. *However, before any circuit is used in an actual application, it should be thoroughly analyzed for applicability to the specific problem at hand.*

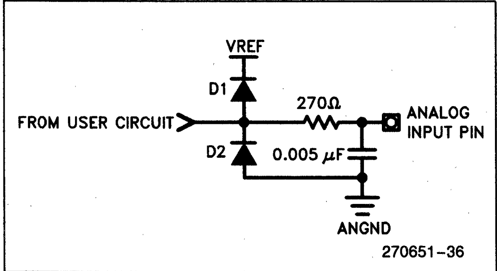


Figure 11-6. Suggested A/D Input Circuit

ANALOG REFERENCES

Reference supply levels strongly influence the absolute accuracy of the conversion. For this reason, it is recommended that the ANGND pin be tied to the two V_{SS} pins at the power supply. Bypass capacitors should also be used between V_{REF} and ANGND. ANGND should be within about a tenth of a volt of V_{SS}. V_{REF} should be well regulated and used only for the A/D converter. The V_{REF} supply can be between 4.5V and 5.5V and needs to be able to source around 5 mA. See Section 13 for the minimum hardware connections.

Note that if only ratiometric information is desired, V_{REF} can be connected to V_{CC}. In addition, V_{REF} and ANGND must be connected even if the A/D converter is not being used. Remember that Port 0 receives its power from the V_{REF} and ANGND pins even when it is used as digital I/O.

11.3 The A/D Transfer Function

The conversion result is a 10-bit ratiometric representation of the input voltage, so the numerical value obtained from the conversion will be:

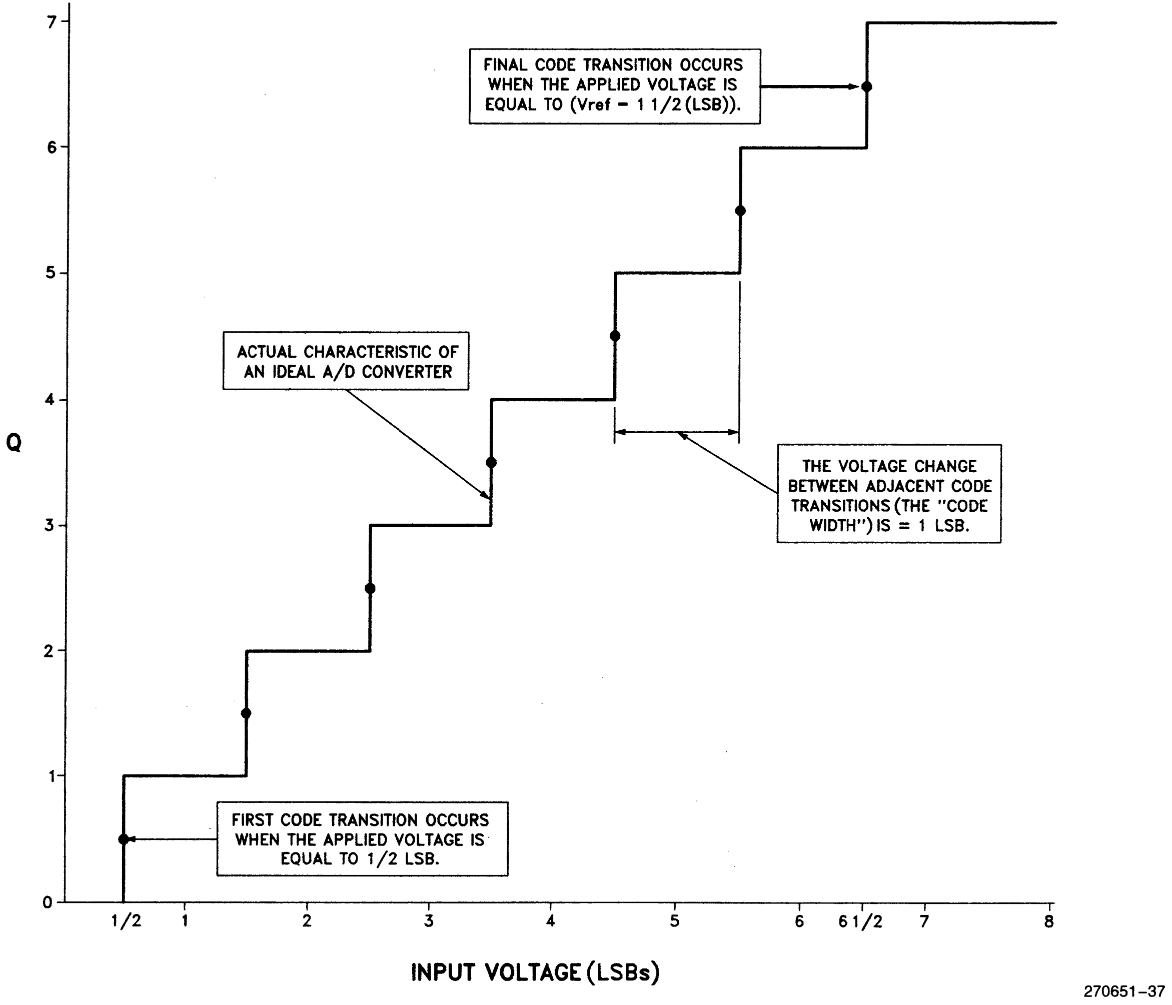
$$\text{INT } [1023 \times (V_{IN} - \text{ANGND})/(V_{REF} - \text{ANGND})].$$

This produces a stair-stepped transfer function when the output code is plotted versus input voltage (see Figure 11-7). The resulting digital codes can be taken as simple ratiometric information, or they provide information about absolute voltages or relative voltage changes on the inputs. The more demanding the application is on the A/D converter, the more important it is to fully understand the converter's operation. For simple applications, knowing the absolute error of the converter is sufficient. However, closing a servo-loop with analog inputs necessitates a detailed understanding of an A/D converter's operation and errors.

The errors inherent in an analog-to-digital conversion process are many: quantizing error, zero offset, full-scale error, differential non-linearity, and non-linearity. These are "transfer function" errors related to the A/D converter. In addition, converter temperature drift, V_{CC} rejection, sample-hold feedthrough, multiplexer off-isolation, channel-to-channel matching and random noise should be considered. Fortunately, one "Absolute Error" specification is available which describes the sum total of all deviations between the actual conversion process and an ideal converter. However, the various sub-components of error are important in many applications. These error components are described in Section 11.5 and in the text below where ideal and actual converters are compared.

An unavoidable error simply results from the conversion of a continuous voltage to an integer digital representation. This error is called quantizing error, and is always ±0.5 LSB. Quantizing error is the only error seen in a perfect A/D converter, and is obviously present in actual converters. Figure 11-7 shows the transfer function for an ideal 3-bit A/D converter (i.e. the Ideal Characteristic).

Note that in Figure 11-7 the Ideal Characteristic possesses unique qualities: it's first code transition occurs when the input voltage is 0.5 LSB; it's full-scale code transition occurs when the input voltage equals the full-



FINAL CODE TRANSITION OCCURS WHEN THE APPLIED VOLTAGE IS EQUAL TO (Vref − 1 1/2 (LSB)).

ACTUAL CHARACTERISTIC OF AN IDEAL A/D CONVERTER

THE VOLTAGE CHANGE BETWEEN ADJACENT CODE TRANSITIONS (THE "CODE WIDTH") IS = 1 LSB.

FIRST CODE TRANSITION OCCURS WHEN THE APPLIED VOLTAGE IS EQUAL TO 1/2 LSB.

Figure 11-7. Ideal A/D Characteristic

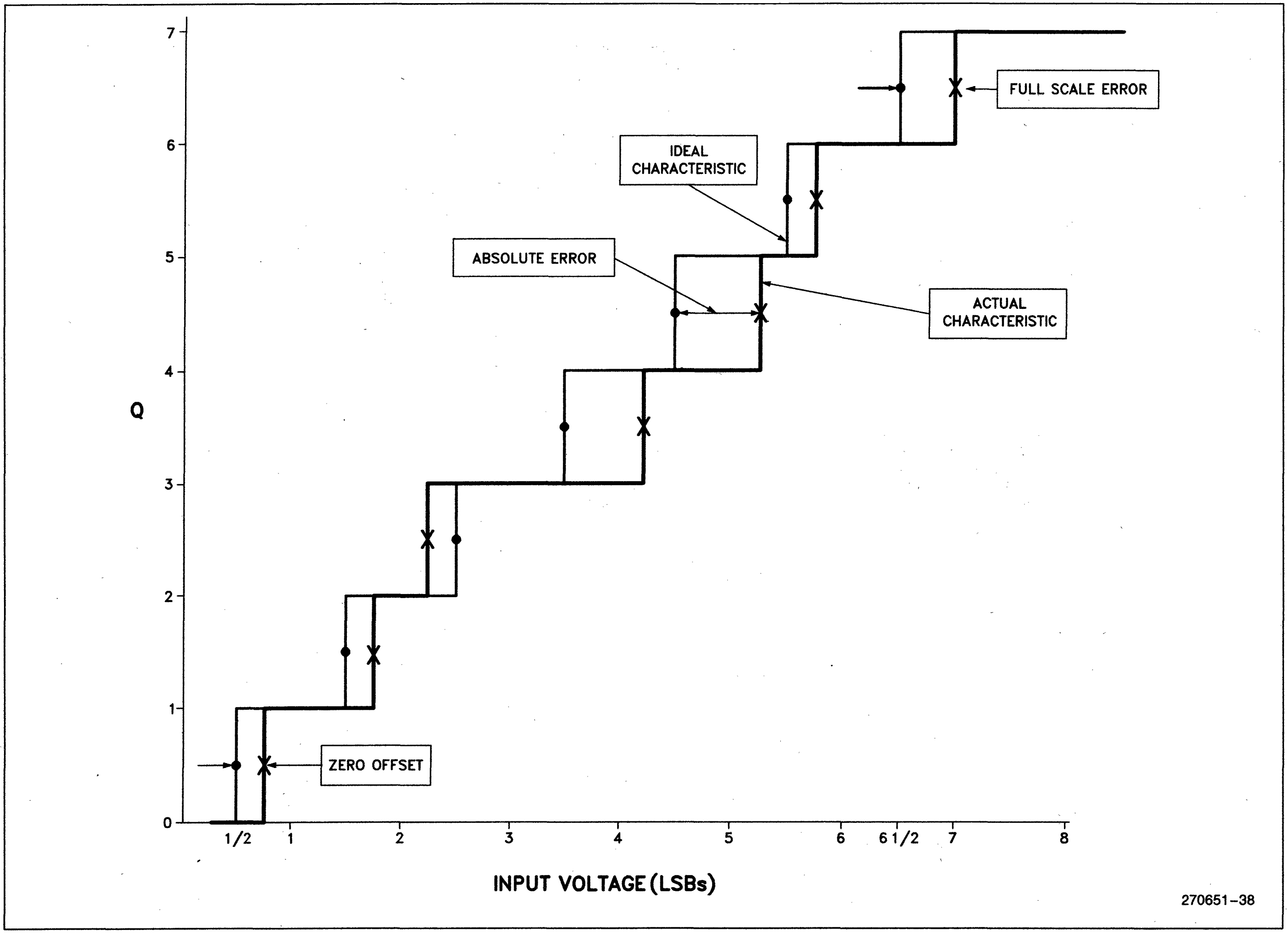


Figure 11-8. Actual and Ideal Characteristics

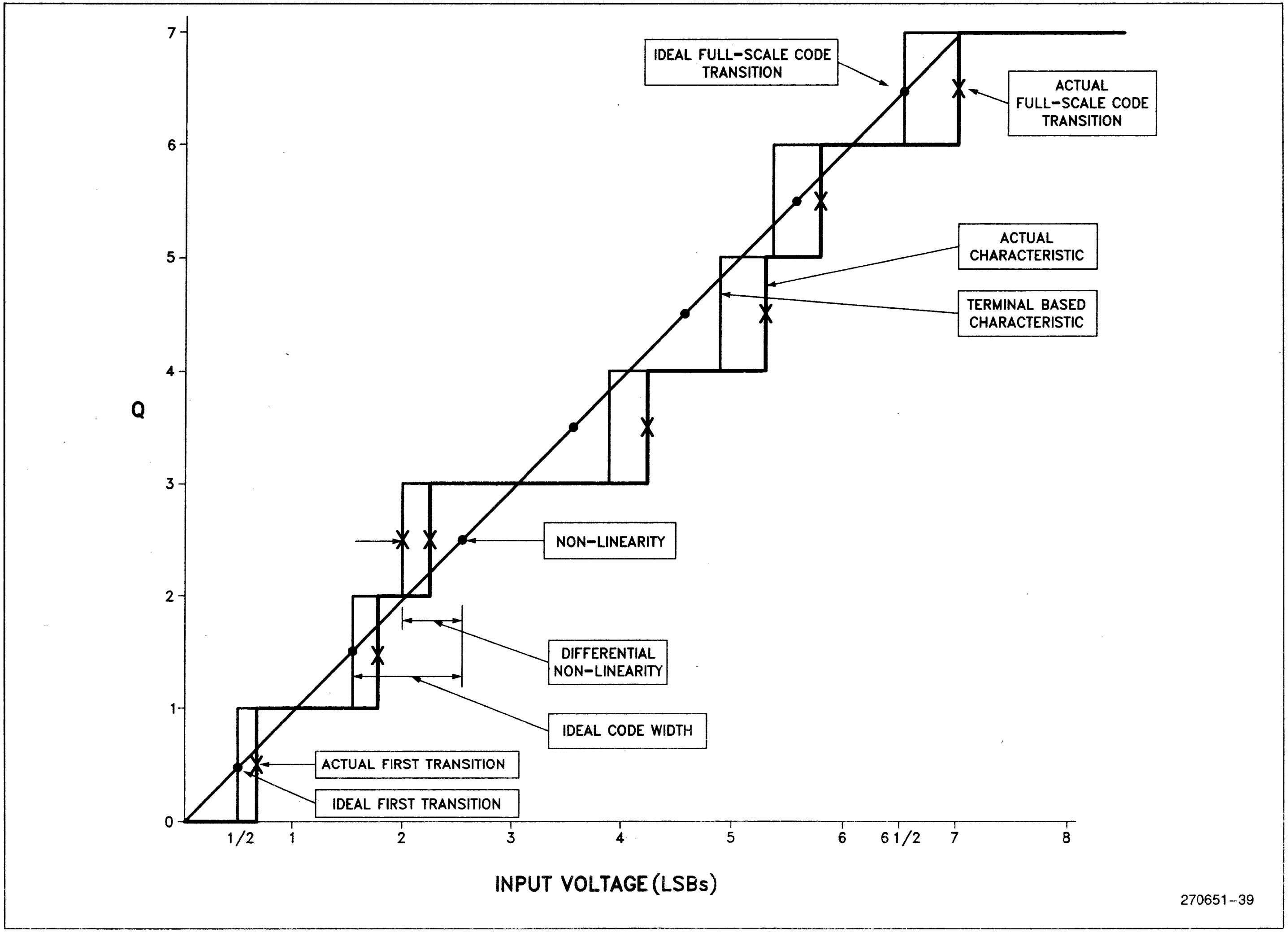


Figure 11-9. Terminal Based Characteristic

scale reference minus 1.5 LSB; and it's code widths are all exactly one LSB. These qualities result in a digitization without offset, full-scale or linearity errors. In other words, a perfect conversion.

Figure 11-8 shows an Actual Characteristic of a hypothetical 3-bit converter, which is not perfect. When the Ideal Characteristic is overlaid with the imperfect characteristic, the actual converter is seen to exhibit errors in the location of the first and final code transitions and code widths. The deviation of the first code transition from ideal is called "zero offset", and the deviation of the final code transition from ideal is "full-scale error". The deviation of the code widths from ideal causes two types of errors. Differential Non-Linearity and Non-Linearity. Differential Non-Linearity is a local linearity error measurement, whereas Non-Linearity is an overall linearity error measure.

Differential Non-Linearity is the degree to which actual code widths differ from the ideal one LSB width. It gives the user a measure of how much the input voltage may have changed in order to produce a one count change in the conversion result. Non-Linearity is the worst case deviation of code transitions from the corresponding code transitions of the Ideal Characteristic. Non-Linearity describes how much Differential Non-Linearities could add up to produce an overall maximum departure from a linear characteristic. If the Differential Non-Linearity errors are too large, it is possible for an A/D converter to miss codes or exhibit non-monotonicity. Neither behavior is desirable in a closed-loop system. A converter has no missed codes if there exists for each output code a unique input voltage range that produces that code only. A converter is monotonic if every subsequent code change represents an input voltage change in the same direction.

Differential Non-Linearity and Non-Linearity are quantified by measuring the Terminal Based Linearity Errors. A Terminal Based Characteristic results when an Actual Characteristic is shifted and rotated to eliminate zero offset and full-scale error (see Figure 11-9). The Terminal Based Characteristic is similar to the Actual Characteristic that would be seen if zero offset and full-scale error were externally trimmed away. In practice, this is done by using input circuits which include gain and offset trimming. In addition, V_{REF} on the 80C196KB could also be closely regulated and trimmed within the specified range to affect full-scale error.

Other factors that affect a real A/D Converter system include sensitivity to temperature, failure to completely reject all unwanted signals, multiplexer channel dissimilarities and random noise. Fortunately these effects are small.

Temperature sensitivities are described by the rate at which typical specifications change with a change in temperature.

Undesired signals come from three main sources. First, noise on V_{CC}—V_{CC} Rejection. Second, input signal changes on the channel being converted after the sample window has closed—Feedthrough. Third, signals applied to channels not selected by the multiplexer—Off-Isolation.

Finally, multiplexer on-channel resistances differ slightly from one channel to the next causing Channel-to-Channel Matching errors, and random noise in general results in Repeatability errors.

11.4 A/D Glossary of Terms

Figures 11-7, 11-8, and 11-9 display many of these terms. Refer to AP-406 'MCS-96 Analog Acquisition Primer' for additional information on the A/D terms.

ABSOLUTE ERROR—The maximum difference between corresponding actual and ideal code transitions. Absolute Error accounts for all deviations of an actual converter from an ideal converter.

ACTUAL CHARACTERISTIC—The characteristic of an actual converter. The characteristic of a given converter may vary over temperature, supply voltage, and frequency conditions. An Actual Characteristic rarely has ideal first and last transition locations or ideal code widths. It may even vary over multiple conversion under the same conditions.

BREAK-BEFORE-MAKE—The property of a multiplexer which guarantees that a previously selected channel will be deselected before a new channel is selected. (e.g. the converter will not short inputs together.)

CHANNEL-TO-CHANNEL MATCHING—The difference between corresponding code transitions of actual characteristićs taken from different channels under the same temperature, voltage and frequency conditions.

CHARACTERISTIC—A graph of input voltage versus the resultant output code for an A/D converter. It describes the transfer function of the A/D converter.

CODE—The digital value output by the converter.

CODE CENTER—The voltage corresponding to the midpoint between two adjacent code transitions.

CODE TRANSITION—The point at which the converter changes from an output code of Q, to a code of Q+1. The input voltage corresponding to a code transition is defined to be that voltage which is equally likely to produce either of two adjacent codes.

CODE WIDTH—The voltage corresponding to the difference between two adjacent code transitions.

CROSSTALK—See "Off-Isolation".

D.C. INPUT LEAKAGE—Leakage current to ground from an analog input pin.

DIFFERENTIAL NON-LINEARITY—The difference between the ideal and actual code widths of the terminal based characteristic of a converter.

FEEDTHROUGH—Attenuation of a voltage applied on the selected channel of the A/D converter after the sample window closes.

FULL SCALE ERROR—The difference between the expected and actual input voltage corresponding to the full scale code transition.

IDEAL CHARACTERISTIC—A characteristic with its first code transition at $V_{IN} = 0.5$ LSB, its last code transition at $V_{IN} = (V_{REF} - 1.5$ LSB) and all code widths equal to one LSB.

INPUT RESISTANCE—The effective series resistance from the analog input pin to the sample capacitor.

LSB—LEAST SIGNIFICANT BIT: The voltage value corresponding to the full scale voltage divided by 2^n, where n is the number of bits of resolution of the converter. For a 10-bit converter with a reference voltage of 5.12 volts, one LSB is 5.0 mV. Note that this is different than digital LSBs, since an uncertainty of two LSBs, when referring to an A/D converter, equals 10 mV. (This has been confused with an uncertainty of two digital bits, which would mean four counts, or 20 mV.)

MONOTONIC—The property of successive approximation converters which guarantees that increasing input voltages produce adjacent codes of increasing value, and that decreasing input voltages produce adjacent codes of decreasing value.

NO MISSED CODES—For each and every output code, there exists a unique input voltage range which produces that code only.

NON-LINEARITY—The maximum deviation of code transitions of the terminal based characteristic from the corresponding code transitions of the ideal characteristics.

OFF-ISOLATION—Attenuation of a voltage applied on a deselected channel of the A/D converter. (Also referred to as Crosstalk.)

REPEATABILITY—The difference between corresponding code transitions from different actual characteristics taken from the same converter on the same channel at the same temperature, voltage and frequency conditions.

RESOLUTION—The number of input voltage levels that the converter can unambiguously distinguish between. Also defines the number of useful bits of information which the converter can return.

SAMPLE DELAY—The delay from receiving the start conversion signal to when the sample window opens.

SAMPLE DELAY UNCERTAINTY—The variation in the Sample Delay.

SAMPLE TIME—The time that the sample window is open.

SAMPLE TIME UNCERTAINTY—The variation in the sample time.

SAMPLE WINDOW—Begins when the sample capacitor is attached to a selected channel and ends when the sample capacitor is disconnected from the selected channel.

SUCCESSIVE APPROXIMATION—An A/D conversion method which uses a binary search to arrive at the best digital representation of an analog input.

TEMPERATURE COEFFICIENTS—Change in the stated variable per degree centigrade temperature change. Temperature coefficients are added to the typical values of a specification to see the effect of temperature drift.

TERMINAL BASED CHARACTERISTIC—An Actual Characteristic which as been rotated and translated to remove zero offset and full-scale error.

V_{CC} REJECTION—Attenuation of noise on the V_{CC} line to the A/D converter.

ZERO OFFSET—The difference between the expected and actual input voltage corresponding to the first code transition.

12.0 I/O PORTS

There are five 8-bit I/O ports on the 80C196KB. Some of these ports are input only, some are output only, some are bidirectional and some have alternate functions. In addition to these ports, the HSI/O unit provides extra I/O lines if the timer related features of these lines are not needed.

Port 0 is an input port which is also used as the analog input for the A/D converter. Port 0 is read at location 0EH. Port 1 is a quasi-bidirectional port and is read or written to through location 0FH. The three most significant bits of Port 1 are the control signals for the $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ bus port pins. Port 2 contains three types of port lines: quasi-bidirectional, input and output. Port2 is read or written from location 10H. The ports cannot be read or written in Window 15. The input and output lines are shared with other functions in the 80C196KB as shown in Figure 12-1. Ports 3 and 4 are open-drain bidirectional ports which share their pins with the address/data bus. On EPROM and ROM parts, Port 3 and 4 are read and written through location 1FFEH.

PIN	FUNC.	ALTERNATE FUNCTION	CONTROL REG.
2.0	Output	TXD (Serial Port Transmit)	IOC1.5
2.1	Input	RXD (Serial Port Receive)	SPCON.3
P2.2	Input	EXTINT	IOC1.1
2.3	Input	T2CLK (Timer2 Clock & Baud)	IOC0.7
2.4	Input	T2RST (Timer2 Reset)	IOC0.5
2.5	Output	PWM Output	IOC1.0
2.6	QBD*	Timer2 up/down select	IOC2.1
2.7	QBD*	Timer2 Capture	N/A

*QBD = Quasi-bidirectional

Figure 12-1. Port 2 Multiple Functions

While discussing the characteristics of the I/O pins some approximate current or voltage specifications will be given. The exact specifications are available in the latest version of the data sheet that corresponds to the part being used.

12.1 Input Ports

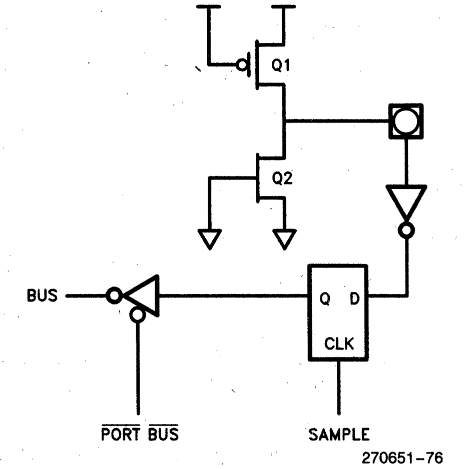
Input ports and pins can only be read. There are no output drivers on these pins. The input leakage of these pins is in the microamp range. The specific values can be found in the data sheet for the device being considered. Figure 12-2 shows the input port structure.

The high impedance input pins on the 80C196KB have an input leakage of a few microamps and are predominantly capacitive loads on the order of 10 pF.

In addition to acting as a digital input, each line of Port 0 can be selected to be the input of the A/D converter as discussed in Section 11. The capacitance on these pins is approximately 1 pF and will instantaneously increase by around 2 pF when the pin is being sampled by the A/D converter.

Port 0 pins are special in that they may individually be used as digital inputs and analog inputs at the same time. A Port 0 pin being used as a digital input acts as the high impedance input ports just described. However, Port 0 pins being used as analog inputs are required to provide current to the internal sample capacitor when a conversion begins. This means that the input characteristics of a pin will change if a conversion is being done on that pin. In either case, if Port 0 is to be used as analog or digital I/O, it will be necessary to provide power to this port through the V_{REF} pin and ANGND pins.

Port 0 is only sampled when the SFR is read to reduce the noise in the A/D converter. The data must be stable one state time before the SFR is read.

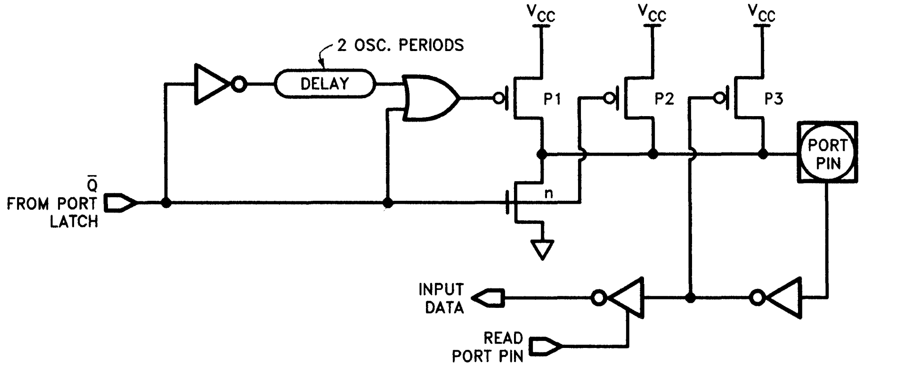


NOTE:
*Q1 and Q2 are ESD Protection Devices

Figure 12-2. Input Port Structure

12.2 Quasi-Bidirectional Ports

Port 1 and Port 2 have quasi-bidirectional I/O pins. When used as inputs the data on these pins must be stable one state time prior to reading the SFR. This timing is also valid for the input-only pins of Port 2 and is similar to the HSI in that the sample occurs during PH1 or during CLKOUT low. When used as outputs, the quasi-bidirectional pins will change state shortly after CLKOUT falls. If the change was from '0' to a '1'



CHMOS Configuration. pFET 1 is turned on for 2 osc. periods after Q̄ makes a 0-to-1 transition. During this time, pFET 1 also turns on pFET 3 through the inverter to form a latch which holds the 1. pFET 2 is also on.

Figure 12-3. CHMOS Quasi-Bidirectional Port Circuit

the low impedance pullup will remain on for one state time after the change.

Port 1, Port 2.6 and Port 2.7 are quasi-bidirectional ports. When the processor writes to the pins of a quasi-bidirectional port it actually writes into a register which in turn drives the port pin. When the processor reads these ports, it senses the status of the pin directly. If a port pin is to be used as an input then the software should write a one to its associated SFR bit, this will cause the low-impedance pull-down device to turn off and leave the pin pulled up with a relatively high impedance pullup device which can be easily driven down by the device driving the input.

If some pins of a port are to be used as inputs and some are to be used as outputs the programmer should be careful when writing to the port.

Particular care should be exercised when using XOR opcodes or any opcode which is a read-modify-write instruction. It is possible for a Quasi-Bidirectional Pin to be written as a one, but read back as a zero if an external device (i.e., a transistor base) is pulling the pin below V_{IH}.

Quasi-bidirectional pins can be used as input and output pins without the need for a data direction register. They output a strong low value and a weak high value. The weak high value can be externally pulled low providing an input function. Figure 12-3 shows the configuration of a CHMOS quasi-bidirectional port.

Outputting a 0 on a quasi-bidirectional pin turns on the strong pull-down and turns off all of the pull-ups. When a 1 is output the pull-down is turned off and 3 pull-ups (strong-P1, weak-P3, very weak-P2) are turned on. Each time a pin switches from 0 to 1 transistor P1 turns on for two oscillator periods. P2 remains on until a zero is written to the pin. P3 is used as a latch, so it is turned on whenever the pin is above the threshold value (around 2 volts).

To reduce the amount of current which flows when the pin is externally pulled low, P3 is turned off when the pin voltage drops below the threshold. The current required to pull the pin from a high to a low is at its maximum just prior to the pull-up turning off. An external driver can switch these pins easily. The maximum current required occurs at the threshold voltage and is approximately 700 microamps.

When the Port 1 pins are used as their alternate functions ($\overline{\text{HOLD}}$, $\overline{\text{HLDA}}$, and $\overline{\text{BREQ}}$), the pins act like a standard output port.

HARDWARE CONNECTION HINTS

When using the quasi-bidirectional ports as inputs tied to switches, series resistors may be needed if the ports will be written to internally after the part is initialized. The amount of current sourced to ground from each pin is typically 7 mA or more. Therefore, if all 8 pins are tied to ground, 56 mA will be sourced. This is equivalent to instantaneously doubling the power used by the chip and may cause noise in some applications.

This potential problem can be solved in hardware or software. In software, never write a zero to a pin being used as an input.

In hardware, a 1K resistor in series with each pin will limit current to a reasonable value without impeding the ability to override the high impedance pullup. If all 8 pins are tied together a 120Ω resistor would be reasonable. The problem is not quite as severe when the inputs are tied to electronic devices instead of switches, as most external pulldowns will not hold 20 mA to 0.0 volts.

Writing to a Quasi-Bidirectional Port with electronic devices attached to the pins requires special attention. Consider using P1.0 as an input and trying to toggle P1.1 as an output:

```
ORB  IOPORT1, #00000001B ; Set P1.0
                         ; for input
XORB IOPORT1, #00000010B ; Complement
                         ; P1.1
```

The first instruction will work as expected but two problems can occur when the second instruction executes. The first is that even though P1.1 is being driven high by the 80C196KB it is possible that it is being held low externally. This typically happens when the port pin drives the base of an NPN transistor which in turn drives whatever there is in the outside world which needs to be toggled. The base of the transistor will clamp the port pin to the transistor's Vbe above ground, typically 0.7V. The 80C196KB will input this value as a zero even if a one has been written to the port pin. When this happens the XORB instruction will always write a one to the port pin's SFR and the pin will not toggle.

The second problem, which is related to the first, is that if P1.0 happens to be driven to a zero when Port 1 is read by the XORB instruction, then the XORB will write a zero to P1.0 and it will no longer be useable as an input.

The first situation can best be solved by the external driver design. A series resistor between the port pin and the base of the transistor often works by bringing up the voltage present on the port pin. The second case can be taken care of in the software fairly easily:

```
LDB  AL, IOPORT1
XORB AL, #010B
ORB  AL, #001B
STB  AL, IOPORT1
```

A software solution to both cases is to keep a byte in RAM as an image of the data to be output to the port; any time the software wants to modify the data on the port it can then modify the image byte and copy it to the port.

If a switch is used on a long line connected to a quasi-bidirectional pin, a pullup resistor is recommended to reduce the possibility of noise glitches and to decrease the rise time of the line. On extremely long lines that are handling slow signals, a capacitor may be helpful in addition to the resistor to reduce noise.

12.3 Output Ports

Output pins include the bus control lines, the HSO lines, and some of Port 2. These pins can only be used as outputs as there are no input buffers connected to them. The output pins are output before the rising edge of PH1 and is valid some time during PH1. Externally, PH1 corresponds to CLKOUT low. It is not possible to use immediate logical instructions such as XOR to toggle these pins.

The control outputs and HSO pins have output buffers with the same output characteristics as those of the bus pins. Included in the category of control outputs are: $\overline{\text{TXD}}$, $\overline{\text{RXD}}$ (in Mode 0), PWM, CLKOUT, ALE, $\overline{\text{BHE}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$. The bus pins have 3 states: output high, output low, and high impedance. Figure 12-4 shows the internal configuration of an output pin.

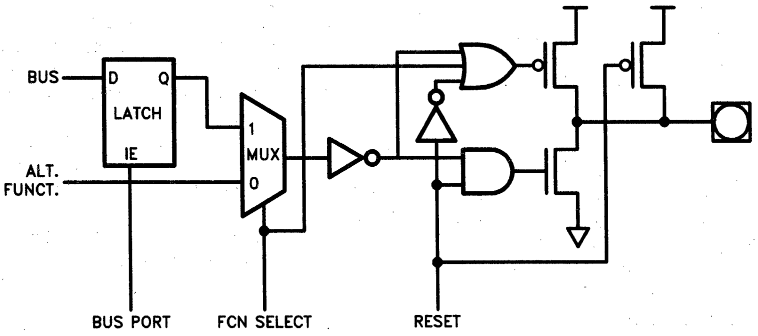


Figure 12-4. Output Port

12.4 Ports 3 and 4/AD0–15

These pins have two functions. They are either bidirectional ports with open-drain outputs or System Bus pins which the memory controller uses when it is accessing off-chip memory. If the $\overline{EA}$ line is low, the pins always act as the System Bus. Otherwise they act as bus pins only during a memory access. If these pins are being used as ports and bus pins, ones must be written to them prior to bus operations.

Accessing Port 3 and 4 as I/O is easily done from internal registers. Since the LD and ST instructions require the use of internal registers, it may be necessary to first move the port information into an internal location before utilizing the data. If the data is already internal, the LD is unnecessary. For instance, to write a word value to Port 3 and 4 . . .

```
LD intreg, portdata   ; register ←
                      ; data
                      ; not needed if
                      ; already
                      ; internal

ST intreg, 1FFEH      ; register →
                      ; Port 3 and 4
```

To read Port 3 and 4 requires that "ones" be written to the port registers to first setup the input port configuration circuit. Note that the ports are reset to this input condition, but if zeroes have been written to the port, then ones must be re-written to any pins which are to be used as inputs. Reading Port 3 and 4 from a previously written zero condition is as follows . . .

```
LD intregA, #0FFFFH   ; setup port
                      ; change mode
                      ; pattern

ST intregA, 1FFEH     ; register →
                      ; Port 3 and 4
                      ; LD & ST not
                      ; needed if
                      ; previously
                      ; written as ones

LD intregB, 1FFEH     ; register ←
                      ; Port 3 and 4
```

Note that while the format of the LD and ST instructions are similar, the source and destination directions change.

When acting as the system bus the pins have strong drivers to both V_{CC} and V_{SS}. These drivers are used whenever data is being output on the system bus and are not used when data is being output by Ports 3 and 4. The pins, external input buffers and pulldowns are shared between the bus and the ports. The ports use different output buffers which are configured as open-drain, and require external pullup resistors. (open-drain is the MOS version of open-collector.) The port pins and their system bus functions are shown in Figure 12-5.

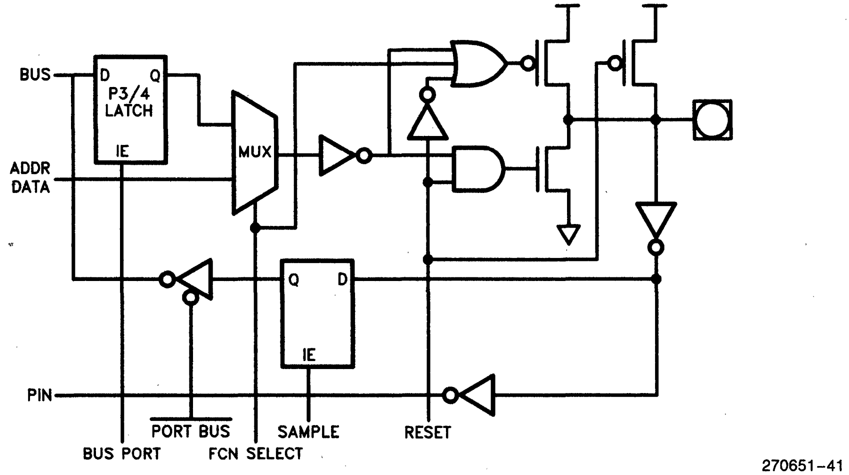


Figure 12-5. Port 3, 4/AD0-15 Pins

Ports 3 and 4 on the 80C196KB are open drain ports. There is no pullup when these pins are used as I/O ports. A diagram of the output buffers connected to Ports 3 and 4 and the bus pins is shown in Figure 12-5.

When Ports 3 and 4 are to be used as inputs, or as bus pins, they must first be written with a '1'. This will put the ports in a high impedance mode. When they are used as outputs, a pullup resistor must be used externally. A 15K pullup resistor will source a maximum of 0.33 milliamps, so it would be a reasonable value to choose if no other circuits with pullups were connected to the pin.

Ports 3 and 4 are addressed as off-chip memory-mapped I/O. The port pins will change state shortly after the falling edge of CLKOUT. When these pins are used as Ports 3 and 4 they are open drains, their structure is different when they are used as part of the bus.

Port 3 and 4 can be reconstructed as I/O ports from the Address/Data bus. Refer to Section 15.7 for details.

13.0 MINIMUM HARDWARE CONSIDERATIONS

The 80C196KB requires several external connections to operate correctly. Power and ground must be connected, a clock source must be generated, and a reset circuit must be present. We will look at each of these areas in detail.

13.1 Power supply

Power to the 80C196KB flows through 5 pins. V_{CC} supplies the positive voltage to the digital portion of the chip while V_{REF} supplies the A/D converter and Port0 with a positive voltage. These two pins need to be connected to a 5 volt power supply. When using the A/D converter, it is desirable to connect V_{REF} to a separate power supply, or at least a separate trace to minimize the noise in the A/D converter.

The four common return pins, $V_{SS}1$, $V_{SS}2$, $V_{SS}3$, and Angd, must all be nominally at 0 volts. Even if the A/D converter is not being used, V_{REF} and Angd must still be connected for Port0 to function.

13.2 Noise Protection Tips

Due to the fast rise and fall times of high speed CMOS logic, noise glitches on the power supply lines and outputs at the chip are not uncommon. The 80C196KB is no exception to this rule. So it is extremely important to follow good design and board layout techniques to keep noise to a minimum. Liberal use of decoupling caps, V_{CC} and ground planes, and transient absorbers can all be of great help. It is much easier to design a board with these features then to search for random noise on a poorly designed PC board. For more information on noise, refer to Applications Note AP-125, 'Designing Microcontroller Systems for Noisy Environments' in the *Embedded Control Application Handbook.*

13.3 Oscillator and Internal Timings

ON-CHIP OSCILLATOR

The on-chip oscillator circuitry for the 80C196KB, as shown in Figure 13.1, consists of a crystal-controlled, positive reactance oscillator. In this application, the crystal is operated in its fundamental response mode as an inductive reactance in parallel resonance with capacitance external to the crystal.

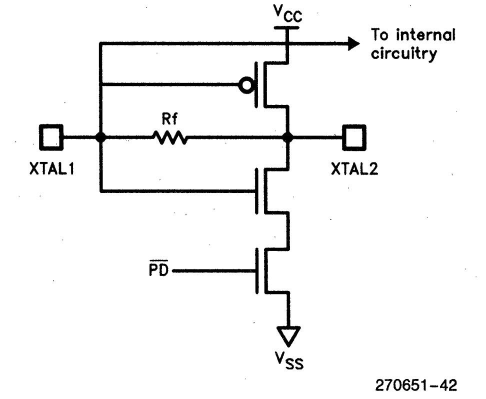


Figure 13-1. On-chip Oscillator Circuitry

The feedback resistor, Rf, consists of paralleled n-channel and p-channel FETs controlled by the PD (powerdown) bit. Rf acts as an open when in Powerdown Mode. Both XTAL1 and XTAL2 also have ESD protection on the pins which is not shown in the figure.

The crystal specifications and capacitance values in Figure 13-2 are not critical. 20 pF is adequate for any frequency above 1 MHz with good quality crystals. Ceramic resonators can be used instead of a crystal in cost sensitive applications. For ceramic resonators, the manufacturer should be contacted for values of the capacitors.

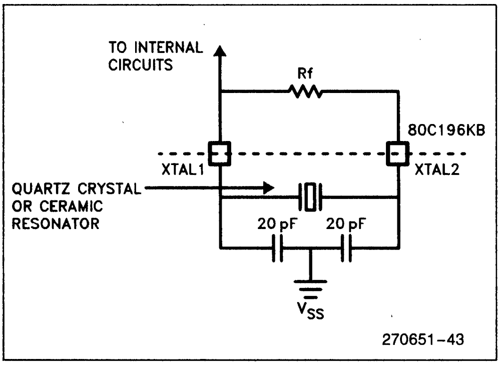


Figure 13-2. External Crystal Connections

To drive the 80C196KB with an external clock source, apply the external clock signal to XTAL1 and let XTAL2 float. An example of this circuit is shown in Figure 13-3. The required voltage levels on XTAL1 are specified in the data sheet. The signal on XTAL1 must be clean with good solid levels.

It is important that the minimum high and low times are met to avoid having the XTAL1 pin in the transition range for long periods of time. The longer the signal is in the transition region, the higher the probability that an external noise glitch could be seen by the clock generator circuitry. Noise glitches on the 80C196KB internal clock lines will cause unreliable operation.

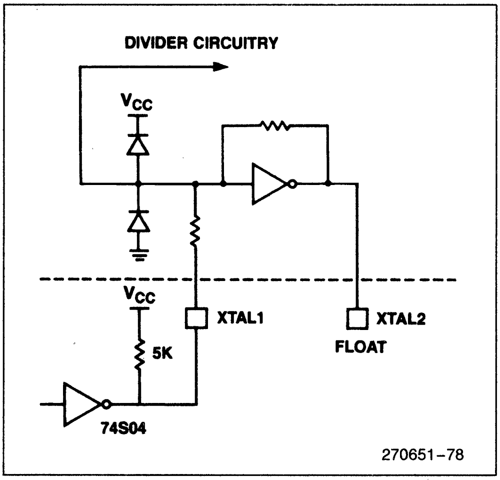


Figure 13-3. External Clock Drive

INTERNAL TIMINGS

Internal operation of the chip is based on the oscillator frequency divided by two, giving the basic time unit, known as a 'state time'. With a 12 Mhz crystal, a state time is 167 nS. Since the 80C196KB can operate at many frequencies, the times given throughout this overview will be in state times.

Two non-overlapping internal phases are created by the clock generator: phase 1 and phase 2 as shown in Figure 13-4. CLKOUT is generated by the rising edge of phase 1 and phase 2. This is not the same as the 8096BH, which uses a three phase clock. Changing from a three phase clock to a two phase one speeds up operation for a set oscillator frequency. Consult the latest data sheet for AC timing specifications.

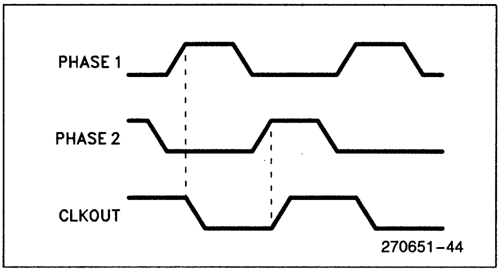


Figure 13-4. Internal Clock Phases

13.4 Reset and Reset Status

Reset starts the 80C196KB off in a known state. To reset the chip, the $\overline{\text{RESET}}$ pin must be held low for at least four state times after the power supply is within tolerance and the oscillator has stabilized. As soon as the $\overline{\text{RESET}}$ pin is pulled low, the I/O and control pins are asynchronously driven to their reset condition.

After the $\overline{\text{RESET}}$ pin is brought high, a ten state reset sequence occurs as shown in Figure 13-5. During this time the CCB (Chip Configuration Byte) is read from location 2018H and stored in the CCR (Chip Configuration Register). The $\overline{\text{EA}}$ (External Access) pin qualifies whether the CCB is read from external or internal memory. Figure 13-6 gives the reset status of all the pins and Special Function Registers.

80C196KB Reset Sequence

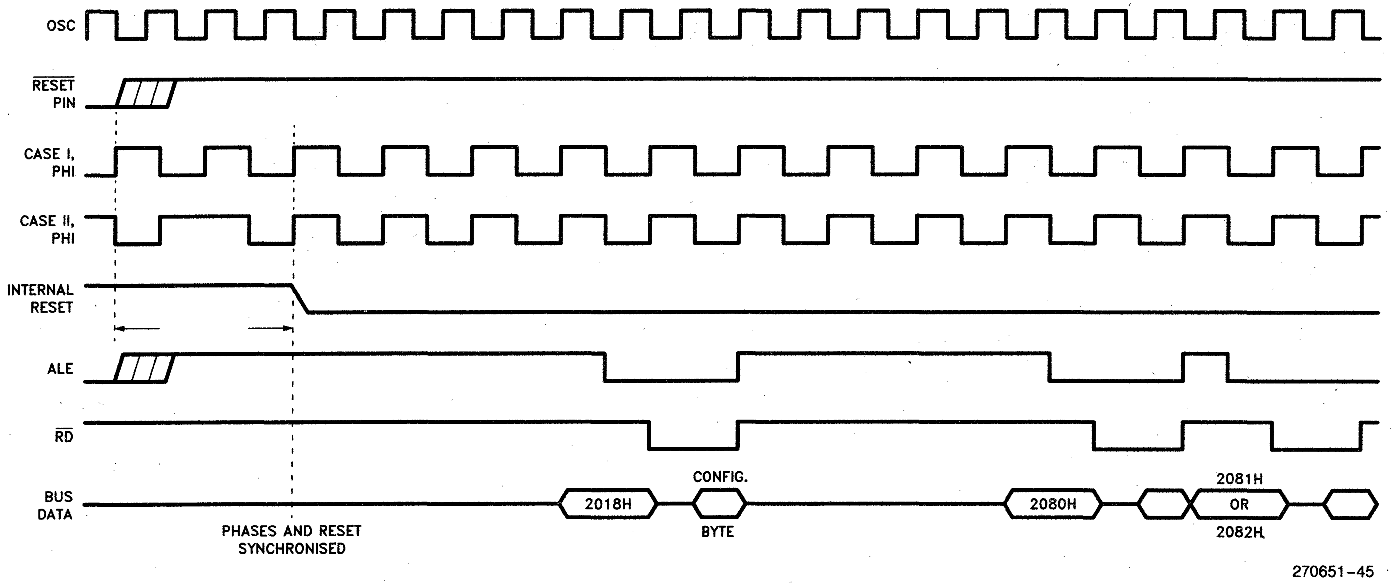


Figure 13-5. Reset Sequence

WATCHDOG TIMER

There are three ways in which the 80C196KB can reset itself. The watchdog timer will reset the 80C196KB if it is not cleared in 64K state times. The watchdog timer is enabled the first time it is cleared. To clear the watchdog, write a '1E' followed immediately by an 'E1' to location 0AH. Once enabled, the watchdog can only be disabled by a reset.

RST INSTRUCTION

Executing a RST instruction will also reset the 80C196KB. The opcode for the RST instruction is 0FFH. By putting pullups on the Addr/data bus, unimplemented areas of memory will read 0FFH and cause the 80C196KB to be reset.

Pin Name	Multiplexed Port Pins	Value of the Pin on Reset
RESET		Mid-sized Pullup
ALE		Weak Pullup
RD		Weak Pullup
BHE		Weak Pullup
WR		Weak Pullup
INST		Weak Pullup
EA		Undefined Input *
READY		Undefined Input *
NMI		Undefined Input *
BUSWIDTH		Undefined Input *
CLKOUT		Phase 2 of Clock
System Bus	P3.0-P4.7	Weak Pullups
ACH0-7	P0.0-P0.7	Undefined Input *
PORT1	P1.0-P1.7	Weak Pullups
TXD	P2.0	Weak Pullup
RXD	P2.1	Undefined Input *
EXTINT	P2.2	Undefined Input *
T2CLK	P2.3	Undefined Input *
T2RST	P2.4	Undefined Input *
PWM	P2.5	Weak Pulldown
—	P2.6-P2.7	Weak Pullups
HSI0-HSI1		Undefined Input *
HSI2/HSO4		Undefined Input *
HSI3/HSO5		Undefined Input *
HSO0-HSO3		Weak Pulldown

Register Name	Value
AD_RESULT	7FF0H
HSI_STATUS	x0x0x0x0B
SBUF(RX)	00H
INT_MASK	00000000B
INT_PENDING	00000000B
TIMER1	0000H
TIMER2	0000H
IOPORT1	11111111B
IOPORT2	11000001B
SP_STAT/SP_CON	00001011B
IMASK1	00000000B
IPEND1	00000000B
WSR	XXXX0000B
HSI_MODE	11111111B
IOC2	X0000000B
IOC0	000000X0B
IOC1	00100001B
PWM_CONTROL	00H
IOPORT3	11111111B
IOPORT4	11111111B
IOS0	00000000B
IOS1	00000000B
IOS2	00000000B

*These pins must be driven and not left floating.

Figure 13-6. Chip Reset Status

RESET CIRCUITS

The simplest way to reset an 80C196KB is to insert a capacitor between the $\overline{\text{RESET}}$ pin and V_{SS}. The 80C196KB has an internal pullup which has a value between 6K and 50K ohms. A 5 uF or greater capacitor should provide sufficient reset time as long as Vcc rises quickly.

Figure 13-7 shows what the $\overline{\text{RESET}}$ pin looks like internally. The $\overline{\text{RESET}}$ pin functions as an input and as an output to reset an entire system with a watchdog timer overflow, or by executing a RST instruction. For a system reset application, the reset circuit should be a one-shot with an open collector output. The reset pulse may have to be lengthened and buffered since $\overline{\text{RESET}}$ is only asserted for four state times. If this is done, it is possible for the 80C196KB to start running before other chips in the system are out of reset. Software must take this condition into account. A capacitor cannot be connected directly to $\overline{\text{RESET}}$ if it is to drive the reset pins of other chips in the circuit. The capacitor may keep the voltage on the pin from going below guaranteed V_{IL} for circuits connected to the $\overline{\text{RESET}}$ pin. Figure 13-8 shows an example of a system reset circuit.

13.5 Minimum Hardware Connections

Figure 13-9 shows the minimum connections needed to get the 80C196KB up and running. It is important to tie all unused inputs to V_{CC} or V_{SS}. If these pins are

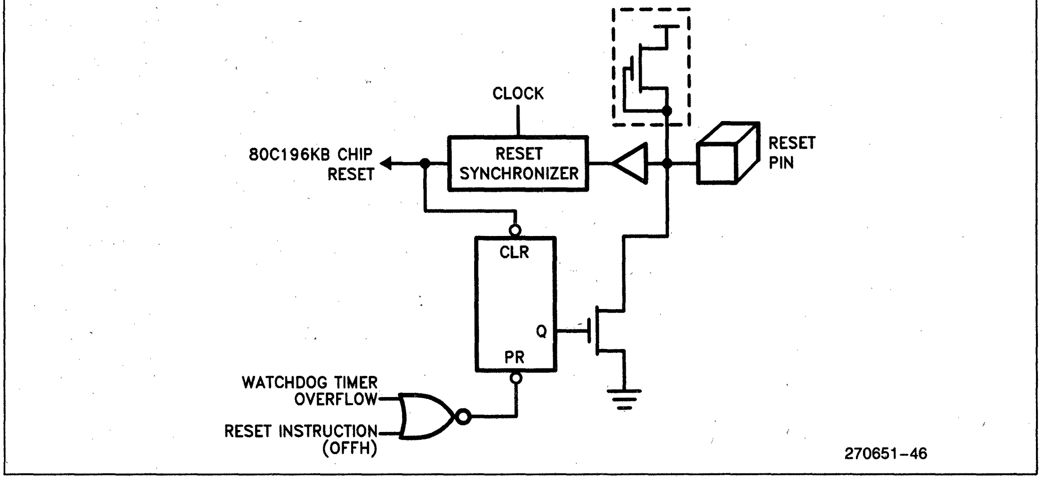
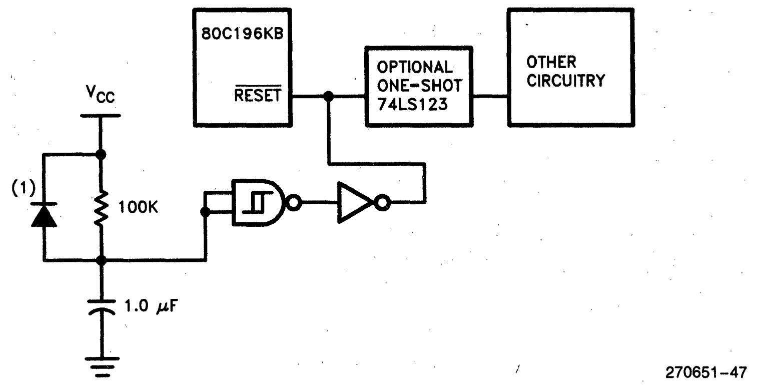
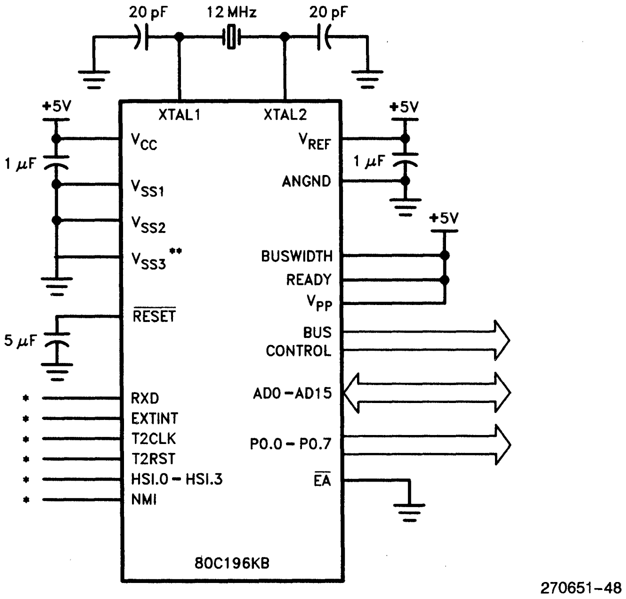


Figure 13-7. Reset Pin



NOTE:
1. The diode will provide a faster cycle time repetitive power-on-resets.

Figure 13-8. System Reset Circuit



NOTE:
*Must be driven high or low.
**V_{SS3} was formerly the CDE pin. The CDE function is no longer available. This pin must be connectd to V_{SS}.

Figure 13-9. 80C196KB Minimum Hardware Connections

left floating, they can float to a mid voltage level and draw excessive current. Some pins such as NMI or EXTINT may generate spurious interrupts if left unconnected.

14.0 SPECIAL MODES OF OPERATION

The 80C196KB has Idle and Powerdown Modes to reduce the amount of current consumed by the chip. The 80C196KB also has an ONCE (ON-Circuit-Emulation) Mode to isolate itself from the rest of the components in the system.

14.1 Idle Mode

The Idle Mode is entered by executing the instruction 'IDLPD #1'. In the Idle Mode, the CPU stops executing. The CPU clocks are frozen at logic state zero, but the peripheral clocks continue to be active. CLKOUT continues to be active. Power consumption in the Idle Mode is reduced to about 40% of the active Mode.

The CPU exits the Idle Mode by any enabled interrupt source or a hardware reset. Since all of the peripherals are running, the interrupt can be generated by the HSI, HSO, A/D, serial port, etc. When an interrupt brings the CPU out of the Idle Mode, the CPU vectors to the corresponding interrupt service routine and begins executing. The CPU returns from the interrupt service routine to the next instruction following the 'IDLPD #1' instruction that put the CPU in the Idle Mode.

In the Idle Mode, the system bus control pins (ALE, $\overline{RD}$, $\overline{WR}$, INST, and $\overline{BHE}$), go to their inactive states. Ports 3 and 4 will retain the value present in their data latches if being used as I/O ports. If these ports are the ADDR/DATA bus, the pins will float.

It is important to note the Watchdog Timer continues to run in the Idle Mode if it is enabled. So the chip must be awakened every 64K state times to clear the Watchdog or the chip will reset.

14.2 Powerdown Mode

The Powerdown Mode is entered by executing the instruction, 'IDLPD #2'. In the Powerdown Mode, all internal clocks are frozen at logic state zero and the oscillator is shut off. All 232 bytes of registers and most peripherals hold their values if V_{CC} is maintained. Power is reduced to the device leakage and is in the uA range. The 87C196KB (EPROM part) will consume more power if the EPROM window is not covered.

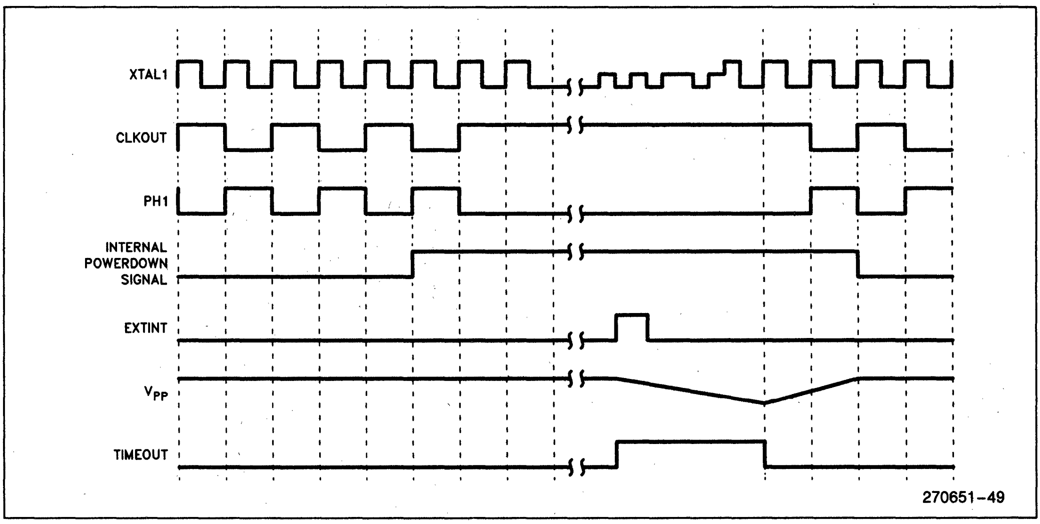


Figure 14-1. Power Up and Power Down Sequence

In Powerdown, the bus control pins go to their inactive states. All of the output pins will assume the value in their data latches. Ports 3 and 4 will continue to act as ports in the single chip mode or will float if acting as the ADDR/DATA bus.

To prevent accidental entry into the Powerdown Mode, this feature may be disabled at reset by clearing bit 0 of the CCR (Chip Configuration Register). Since the default value of the CCR bit 0 is 1, the Powerdown Mode is normally enabled.

The Powerdown Mode can be exited by a chip reset or a high level on the external interrupt pin. If the $\overline{\text{RESET}}$ pin is used, it must be asserted long enough for the oscillator to stabilize.

When exiting Powerdown with an external interrupt, a positive level on the pin mapped to INT7 (either EXTINT or port0.7) will bring the chip out of Powerdown Mode. The interrupt does not have to be unmasked to exit Powerdown. An internal timing circuit ensures that the oscillator has time to stabilize before turning on the internal clocks. Figure 14-1 shows the power down and power up sequence using an external interrupt.

During normal operation, before entering Powerdown Mode, the V_{PP} pin will rise to V_{CC} through an internal pullup. The user must connect a capacitor between V_{PP} and V_{SS}. A positive level on the external interrupt pin starts to discharge this capacitor. The internal current source that discharges the capacitor can sink approximately 100 uA. When the voltage goes below about 1 volt on the V_{PP} pin, the chip begins executing code. A 1uF capacitor would take about 4 ms to discharge to 1 volt.

If the external interrupt brings the chip out of Powerdown, the corresponding bit will be set in the interrupt pending register. If the interrupt is unmasked, the part will immediately execute the interrupt service routine, and return to the instruction following the IDLPD instruction that put the chip into Powerdown. If the interrupt is masked, the chip will start at the instruction following the IDLPD instruction. The bit in the pending register will remain set, however.

All peripherals should be in an inactive state before entering Powerdown. If the A/D converter is in the middle of a conversion, it is aborted. If the chip comes out of Powerdown by an external interrupt, the serial port will continue where it left off. Make sure that the serial port is done transmitting or receiving before entering Powerdown. The SFRs associated with the A/D and the serial port may also contain incorrect information when returning from Powerdown.

When the chip is in Powerdown, it is impossible for the watchdog timer to time out because its clock has stopped. Systems which must use the Watchdog and Powerdown, should clear the Watchdog right before entering Powerdown. This will keep the Watchdog from timing out when the oscillator is stabilizing after leaving Powerdown.

14.3 ONCE™ and Test Modes

Test Modes can be entered on the 80C196KB by holding ALE, $\overline{\text{INST}}$ or $\overline{\text{RD}}$ in their active state on the rising edge of $\overline{\text{RESET}}$. The only Test Mode not reserved for use by Intel is the ONCE, or ON-Circuit-Emulation Mode.

ONCE is entered by driving ALE high, INST low and RD low on the rising edge of RESET. All pins except XTAL1 and XTAL2 are floated. Some of the pins are not truly high impedance as they have weak pullups or pulldowns. The ONCE Mode is useful in electrically removing the 80C196KB from the rest of the system. A typical application of the ONCE Mode would be to program discrete EPROMs onboard without removing the 80C196KB from its socket.

ALE, INST, and RD are weakly pulled high or low during reset. It is important that a circuit does not inadvertantly drive these signals during reset, or a Test Mode could be entered by accident.

15.0 EXTERNAL MEMORY INTERFACING

15.1 Bus Operation

There are several different external operating modes on the 80C196KB. The standard bus mode uses a 16 bit multiplexed address/data bus. Other bus modes include an 8 bit external bus mode and a mode in which the bus size can be dynamically switched between 8-bits and 16-bits. In addition, there are several options available on the type of bus control signals which make an external bus simple to design.

In the standard mode, external memory is addressed through lines AD0-AD15 which form a 16 bit multiplexed bus. The address/data bus shares pins with ports 3 and 4. Figure 15-1 shows an idealized timing diagram for the external bus signals.

Address Latch Enable (ALE) provides a strobe to transparent latches (74AC373s) to demultiplex the bus. To avoid confusion, the latched address signals will be called MA0-MA15 and the data signals will be named MD0-MD15.

The data returned from external memory must be on the bus and stable for a specified setup time before the rising edge of RD (read). The rising edge of RD signals the end of the sampling window. Writing to external memory is controlled with the WR (write) pin. Data is valid on MD0-MD15 on the rising edge of WR. At this time data must be latched by the external system. The 80C196KB has ample setup and hold times for writes.

When BHE is asserted, the memory connected to the high byte of the data bus is selected. When MA0 is a 0, the memory connected to the low byte of the data bus is selected. In this way accesses to a 16-bit wide memory can be to the low (even) byte only (MA0 = 0, BHE = 1), to the high (odd) byte only (MA0 = 1, BHE = 0), or the both bytes (MA0 = 0, BHE = 0).

When a block of memory is decoded for reads only, the system does not have to decode BHE and MA0. The 80C196KB will discard the byte it does not need. For systems that write to external memory, a system must generate separate write strobes to both the high and low byte of memory. This is discussed in more detail later.

All of the external bus signals are gated by the rising and falling edges of CLKOUT. A zero waitstate bus cycle consists of two CLKOUT periods. Therefore, there are 4 clock edges that generate a complete bus cycle. The first falling edge of CLKOUT asserts ALE and drives an address on the bus. The rising edge of

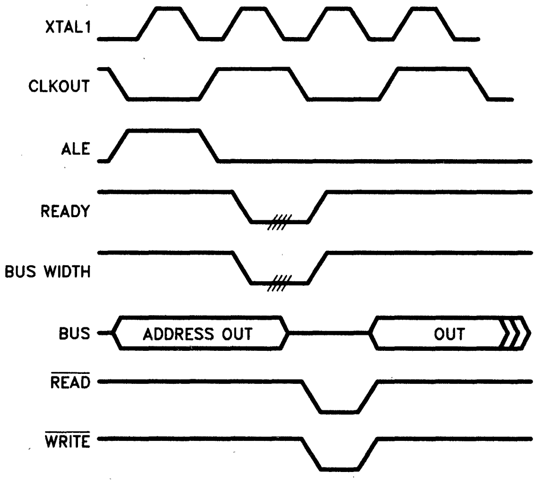


Figure 15-1. Idealized Bus Timings

CLKOUT drives ALE inactive. The next falling edge of CLKOUT asserts $\overline{RD}$ (read) and floats the bus for a read cycle. During a $\overline{WR}$ (write) cycle, this edge asserts $\overline{WR}$ and drives valid data on the bus. On the last rising edge of CLKOUT, data is latched into the 80C196KB for a read cycle, or data is valid for a write cycle.

READY PIN

The READY pin can insert wait states into the bus cycle for interfacing to slow memory or peripherals. A wait state is 2 Tosc in length. Since the bus is synchronized to CLKOUT, it can only be held for an integral number of waitstates. Because the 80C196KB is a completely static part, the number of waitstates that can be inserted into a bus cycle is unbounded. Refer to the next section for information on internally controlling the number of waitstates inserted into a bus cycle.

There are several setup and hold times associated with the READY signal. If these timings are not met, the part may insert the incorrect number of waitstates.

INST pin

The INST pin is useful for decoding more than 64K of addressing space. The INST pin allows both 64K of code space and 64K of data space. For instruction fetches from external memory, the INST pin is asserted, or high for the entire bus cycle. For data reads and writes, the INST pin is low. The INST pin is low for the Chip Configuration Byte fetch and for interrupt vector fetches.

15.2 Chip Configuration Register

The CCR (Chip Configuration Register) is the first byte fetched from memory following a chip reset. The CCR is fetched from the CCB (Chip Configuration Byte) at location 2018H in either internal or external memory depending on the state of the $\overline{EA}$ pin. The CCR is only written once during the reset sequence. Once loaded, the CCR cannot be changed until the next reset.

The CCR is shown in Figure 15-2. The two most significant bits control the level of ROM/EPROM protection. ROM/EPROM protection is covered in the last section. The next two bits control the internal READY mode. The next three bits determine the bus control signals. The last bit enables or disables the Powerdown Mode.

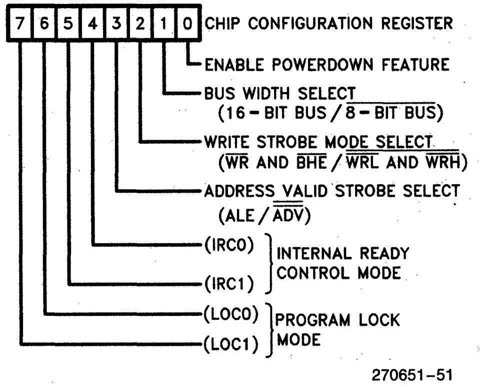

7 6 5 4 3 2 1 0
CHIP CONFIGURATION REGISTER
ENABLE POWERDOWN FEATURE
BUS WIDTH SELECT (16-BIT BUS/$\overline{8\text{-BIT BUS}}$)
WRITE STROBE MODE SELECT ($\overline{WR}$ AND $\overline{BHE}$/$\overline{WRL}$ AND $\overline{WRH}$)
ADDRESS VALID STROBE SELECT (ALE/$\overline{ADV}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE
270651-51

Figure 15-2. Chip Configuration Register

READY control

To simplify ready control, four modes of internal ready control are available. The modes are chosen by bits 4 and 5 of the CCR and are shown in Figure 15-3.

IRC1	IRC0	Description
0	0	Limit to one wait state
0	1	Limit to two wait states
1	0	Limit to three wait states
1	1	Wait states not limited internally

Figure 15-3. Ready control modes

The internal ready control logic limits the number of waitstates that slow devices can insert into the bus cycle. When the READY pin is pulled low, waitstates are inserted into the bus cycle until the READY pin goes high, or the number of waitstate equal the number programmed into the CCR. So the ready control is a simple logical OR between the READY pin and the internal ready control.

This feature gives very simple and flexible ready control. For example, every slow memory chip select line could be ORed together and connected to the READY pin with Internal Ready Control programmed to insert the desired number of waitstates into the bus cycle.

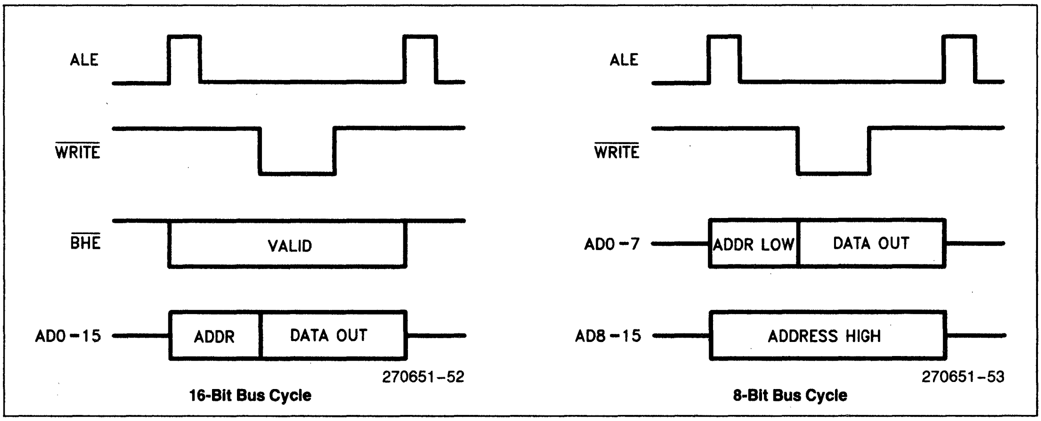
If the READY pin is pulled low during the CCR fetch, the bus controller will automatically insert 3 waitstates into the CCR bus cycle. This allows the CCR fetch to come from slow memory without having to assert the READY pin.

Bus control

Using the CCR, the 80C196KB can generate several types of control signals designed to reduce external hardware. The ALE, $\overline{\text{WR}}$, and $\overline{\text{BHE}}$ pins serve dual functions. Bits 2 and 3 of the CCR specify the function performed by these control lines.

Standard bus control

If CCR bits 2 and 3 are 1s, the standard bus control signals ALE, $\overline{\text{WR}}$, and $\overline{\text{BHE}}$ are generated as shown in Figure 15-4. ALE rises as the address starts to be driven, and falls to externally latch the address. $\overline{\text{WR}}$ is driven for every write. $\overline{\text{BHE}}$ and MA0 can be combined to form $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$ for even and odd byte writes.



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 15-4. Standard Bus Control

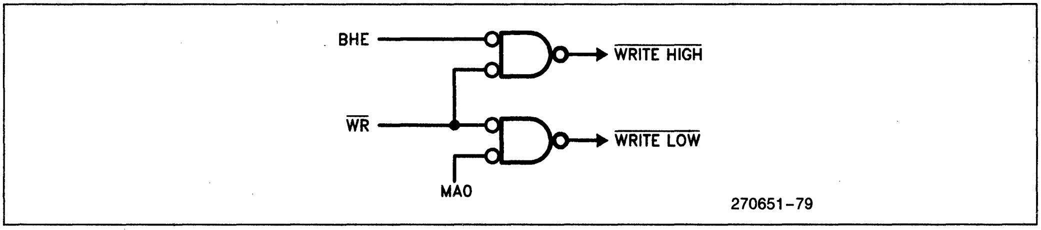


Figure 15-5. Decoding $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$

Figure 15-5 is an example of external circuitry to decode $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$.

Write Strobe Mode

The Write Strobe Mode eliminates the need to externally decode for odd and even byte writes. If CCR bit 2 is 0, and the bus is a 16-bit cycle, $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$ are generated in place of $\overline{\text{WR}}$ and $\overline{\text{BHE}}$. $\overline{\text{WRL}}$ is asserted for all byte writes to an even address and all word writes. $\overline{\text{WRH}}$ is asserted for all byte writes to odd addresses and all word writes. The Write Strobe mode is shown in Figure 15-6.

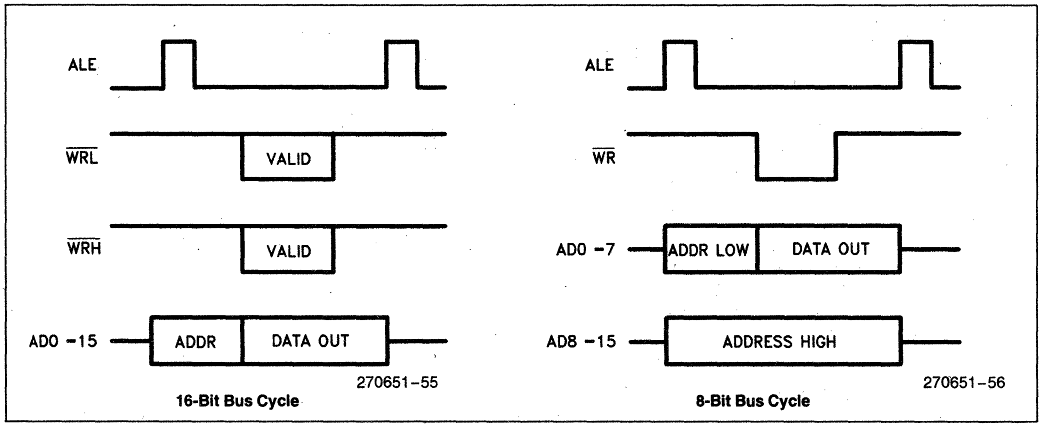
In the eight bit mode, $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$ are asserted for both even and odd addresses.

Address Valid Strobe Mode

Address Valid strobe replaces ALE if CCR bit 3 is 0. When Address valid Strobe mode is selected, $\overline{\text{ADV}}$ will be asserted after an external address is setup. It will stay asserted until the end of the bus cycle as shown in Figure 15-7. $\overline{\text{ADV}}$ can be used as a simple chip select for external memory. $\overline{\text{ADV}}$ looks exactly like ALE for back to back bus cycles. The only difference is $\overline{\text{ADV}}$ will be inactive when the external bus is idle.

Address Valid with Write Strobe

If CCR bits 2 and 3 are 0, the Address Valid with Write Strobe mode is enabled. Figure 15-8 shows the signals.



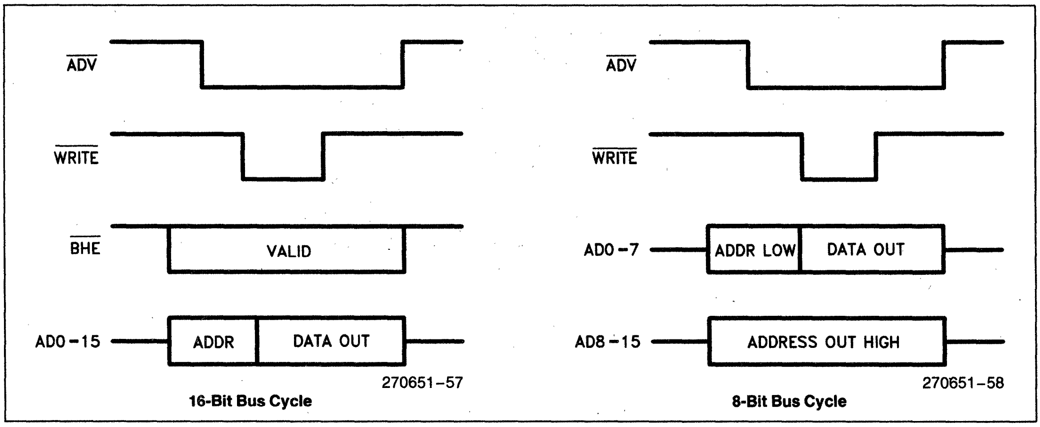
270651-55

16-Bit Bus Cycle

270651-56

8-Bit Bus Cycle

Figure 15-6. Write Strobe Mode



270651-57

16-Bit Bus Cycle

270651-58

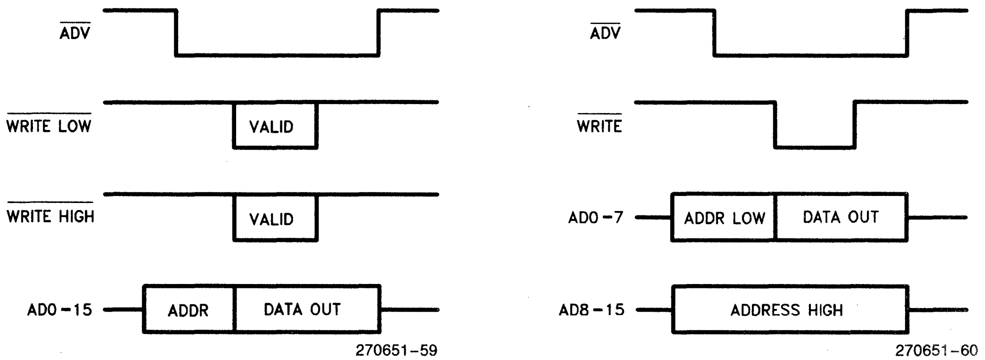
8-Bit Bus Cycle

Figure 15-7. Address Valid Strobe Mode

15.3 Bus Width

The 80C196KB external bus width can be run-time conFigured to operate as a 16 bit multiplexed address/data bus, or as an MCS-51 style multiplexed 16 bit address/8 bit data bus.

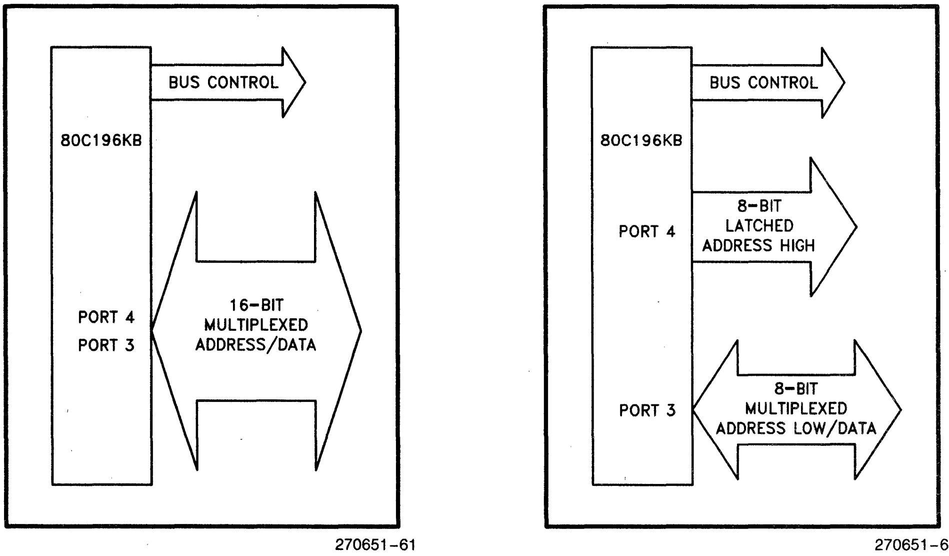
During 16 bit bus cycles, Ports 3 and 4 contain the address multiplexed with data using ALE to latch the address. In 8-bit bus cycles, Port 3 is multiplexed with address/data but Port 4 only outputs the upper 8 address bits. The Addresses on Port 4 are valid throughout the entire bus cycle. Figure 15-9 shows the two bus width options.



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 15-8. Address Valid with Write Strobe Mode


80C196KB
BUS CONTROL
PORT 4
PORT 3
16-BIT MULTIPLEXED ADDRESS/DATA
270651-61
80C196KB
BUS CONTROL
PORT 4
8-BIT LATCHED ADDRESS HIGH
PORT 3
8-BIT MULTIPLEXED ADDRESS LOW/DATA
270651-62

(a) 16-Bit Bus

(b) 8-Bit Bus

Figure 15-9. Bus Width Options

The external bus width can be changed every bus cycle if a 1 was loaded into bit CCR.1 at reset. The bus width is changed on the fly by using the BUSWIDTH pin. If the BUSWIDTH pin is a 1, the bus cycle is 16-bits. For an 8-bit bus cycle, the BUSWIDTH pin is a zero. The BUSWIDTH is sampled by the 80C196KB after the address is on the bus. The BUSWIDTH pin has about the same timing as the READY pin.

Applications for the BUSWIDTH pin are numerous. For example, a system could have code fetched from 16 bit memory, while data would come from 8 bit memory. This saves the cost of using two 8 bit static RAMS if only the capacity of one is needed. This system could be easily implemented by tying the chip select input of the 8-bit memory to the BUSWIDTH pin.

If CCR bit 1 is a 0, the 80C196KB is locked into the 8 bit mode and the BUSWIDTH pin is ignored.

When executing code from a 8-bit bus, some performance degradation is to be expected. The prefetch queue cannot be kept full under all conditions from an 8-bit bus. Also, word reads and writes to external memory will take an extra bus cycle for the extra byte.

15.4 HOLD/HLDA Protocol

The 80C196KB supports a bus exchange protocol, allowing other devices to gain control of the bus. The protocol consists of three signals, HOLD, HLDA, and BREQ. HOLD is an input asserted by a device which requests the 80C196KB bus. Figure 15-10 shows the timing for HOLD/HLDA. The 80C196KB responds by releasing the bus and asserting HLDA. When the device is done accessing the 80C196KB memory, it relinquishes the bus by deactivating the HOLD pin. The 80C196KB will remove its HDLA and assume control of the bus. The third signal, BREQ, is asserted by the 80C196KB during the hold sequence when it has a pending external bus cycle. The 80C196KB deactivates BREQ at the same time it deactivates HDLA.

The HOLD, HLDA, and BREQ pin are multiplexed with P1.7, P1.6, and P1.5, respectively. To enable HOLD, HLDA and BREQ, the HLDEN bit (WSR.7) must be to 1. HLDEN is cleared during reset. Once this bit is set, the port1 pins cannot be returned to being quasi-bidirectional pins until the device is reset, but can still be read. The HOLD/HLDA feature, however, can be disabled by clearing the HLDEN bit.

The HOLD is sampled on phase 1, or when CLKOUT is low.

When the 80C196KB acknowledges the hold request, the output buffers for the addr/data bus, RD, WR, BHE and INST are floated. Although the strong pullup and pulldown on ALE/ADV are disabled, a weak pulldown is turned on. This provides the option to wire OR ALE with other bus masters. The request to hold latency is dependent on the state of the bus controller.

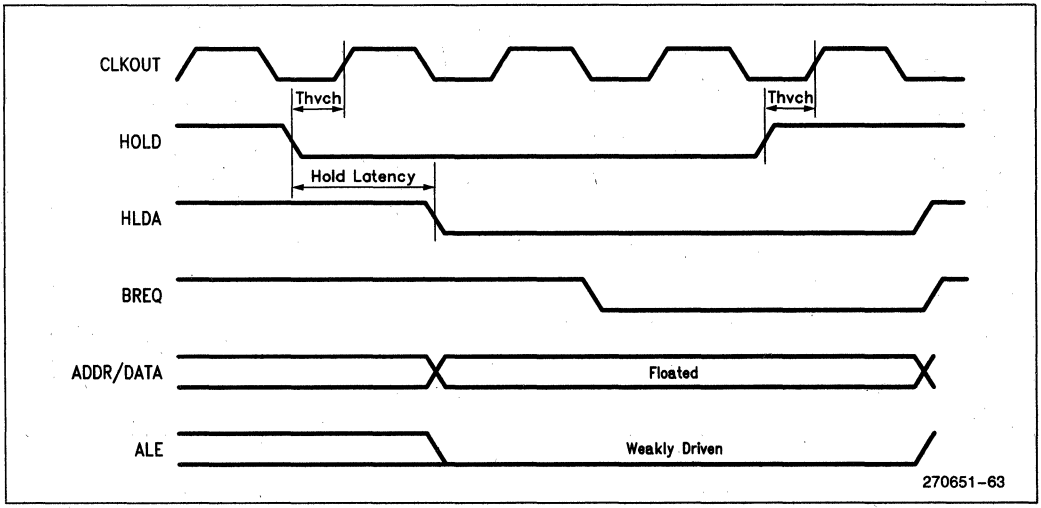


Figure 15-10. HOLD/HLDA Timings

MAXIMUM HOLD LATENCY

The time between HOLD being asserted and HLDA being driven is known as Hold Latency. After recognizing HOLD, the 80C196KB waits for any current bus cycle to finish, and then asserts HLDA. There are 3 types bus cycles; 8-bit external cycle, 16-bit external cycle, and an idle bus. Accessing on-chip ROM/EPROM is an idle bus.

HOLD is an asynchronous input. There are two different system configurations for asserting HOLD. The 80C196KB will recognize HOLD internally on the next clock edge if the system meets Thvch (HOLD valid to CLKOUT high). If Thvch is not met (HOLD applied asynchronously), HOLD may be recognized one clock later (see Figure 15-12). Consult the latest 80C196KC data sheet for the Thvch specification.

Figure 15-12 shows the 80C196KB entering HOLD when the bus is idle. This is the minimum hold latency for both the synchronous and asynchronous cases. If Thvch is met, HLDA is asserted about on the next falling edge of CLKOUT. See the data sheet for Tclhal (CLKOUT low to HLDA low) specification. For this case, the minimum hold latency = Thvcl + 0.5 states + Tclhal.

If HOLD is asserted asynchronously, the minimum hold latency increases by one state time and = Thvcl + 1.5 states + Tclhal.

Figure 15-11 summarizes the additional hold latency added to the minimum latency for the 3 types of bus cycles. When accessing external memory, add one state for each waitstate inserted into the bus cycle. For an 8-bit bus, worst case hold latency is for word reads or writes. For this case, the bus controller must access the bus twice, which increases latency by two states.

For exiting Hold, the minimum hold latency times apply for when the 80C196KB will deassert HLDA in response to HOLD being removed.

Idle Bus	Min
16-bit External Access	Min + 1 state
8-bit External Access	Min + 3 states

Min = Thvcl + 0.5 states + Tclhal if Thvcl is met
= Thvcl + 1.5 states + Tclhal for asynchronous HOLD

Figure 15-11. Maximum Hold Latency

REGAINING BUS CONTROL

There is no delay from the time the 80C196KB removes HLDA to the time it takes control of the bus. After HOLD is removed, the 80C196KB drops HLDA in the following state and resumes control of the bus.

BREQ is asserted when the part is in hold and needs to perform an external memory cycle. An external memory cycle can be a data access or a request from the prefetch queue for a code request. A request comes from the queue when it contains two bytes or less. Once asserted, it remains asserted until HOLD is removed. At the earliest, BREQ can be asserted with HLDA.

Hold requests do not freeze the 80C196KB when executing out of internal memory. The part continues executing as long as the resources it needs are located internal to the 80C196KB. As soon as the part needs to access external memory, it asserts BREQ and waits for the HOLD to be removed. At this time, the part cannot respond to any interrupt requests until HOLD is removed.

When executing out of external memory during a HOLD, the 80C196KB keeps running until the queue is empty or it needs to perform an external data cycle. The 80C196KB cannot service any interrupts until HOLD is removed.

The 80C196KB will also respond to hold requests in the Idle Mode. The latency for entering bus hold from the Idle Mode is the same as when executing out of internal memory.

Special consideration must be given to the bus arbiter design if the 80C196KB can be reset while in HOLD. For example, a CPU part would try and fetch the CCR from external memory after RESET is brought high. Now there would be two parts attempting to access 80C196KB memory. Also, if another bus master is directly driving ALE, RD, and INST, the ONCE mode or another test mode could be entered. The simplest solution is to make the RESET pin of the 80C196KB a system reset. This way the other bus master would also be reset. Examples of system reset circuits are given in Section 13.

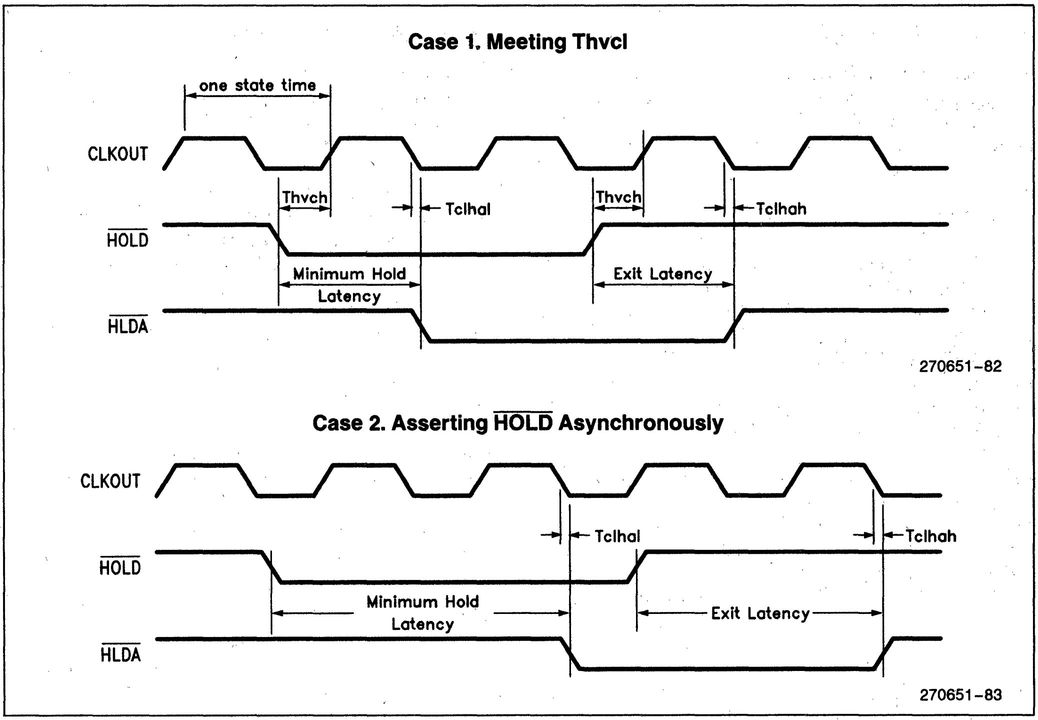


Figure 15-12. HOLD Applied Asynchronously

DISABLING HOLD REQUESTS

Clearing the HLDEN bit (WSR.7), can disable HOLD requests when consecutive memory cycles are required. Clearing the HDLEN bit, however, does not cause the 80C196KB to take over the bus immediately. The 80C196KB waits for the current HOLD request to finish. Then it disables the bus hold feature, causing any new requests to be ignored until the HLDEN bit is set again. Since there is a delay from the time the code for clearing this bit is fetched to the time it is actually executed, the code that clears HLDEN needs to be a few instructions ahead of the block that needs to be protected from HOLD requests.

The safest way is to add a JBC instruction to check the status of the HLDA pin after the code that clears the HLDEN bit. Figure 15-13 is an example of code that prevents the part from executing a new instruction until both current HOLD requests are serviced and the hold feature is disabled.

15.5 AC Timing Explanations

Figure 15-14 shows the timing of the ADDR/DATA bus and control signals. Refer to the latest data sheet for the AC timings to make sure your system meets specifications. The major timing specifications are explained in Figure 15-15.

```
       DI                 ; disable interrupts
       ANDB WSR, #0EFH    ; disable hold request
WAIT:  JBC PORT1, 6, WAIT; Check the HLDA pin
             •            ; If set, execute
             •            ; protected instructions
             •
       ORB WSR,#80h       ; enable HOLD requests
       EI                 ; enable interrupts
```

NOTE:
Interrupts should be disabled to prevent code interruption

Figure 15-13. HOLD code

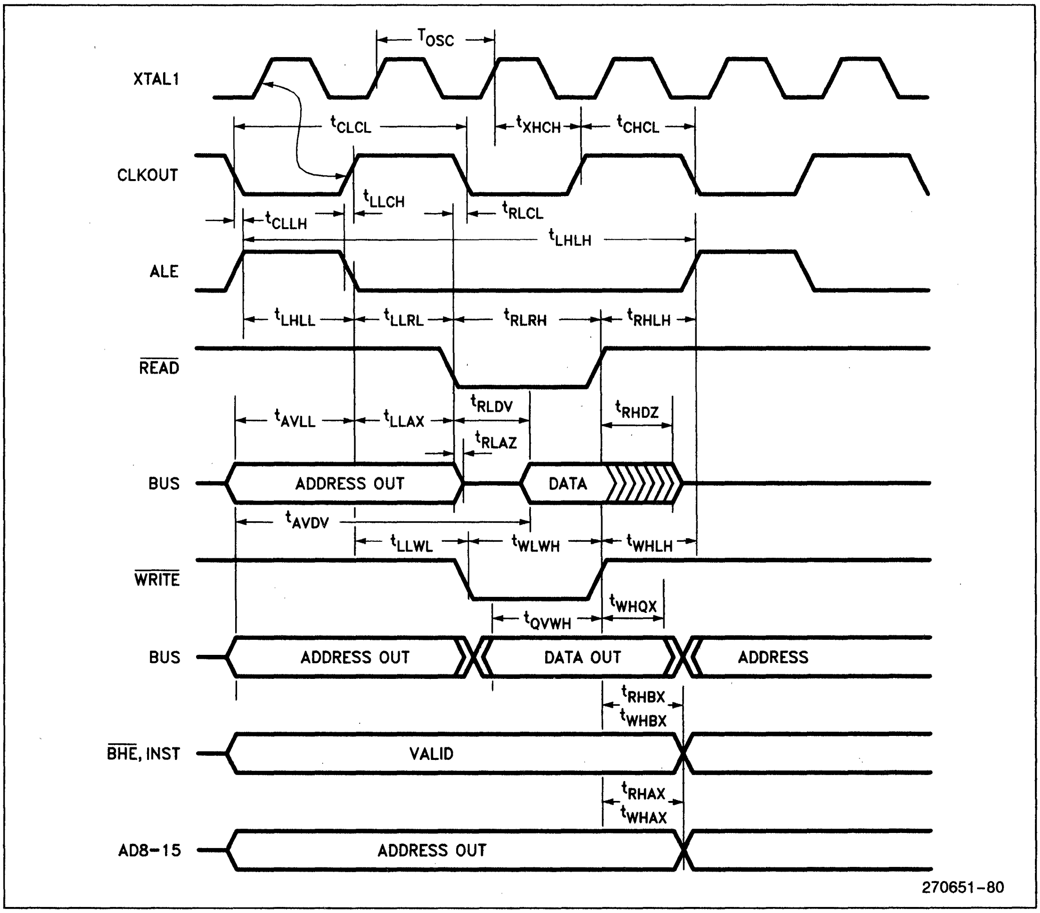


Figure 15-14. AC Timing Diagrams

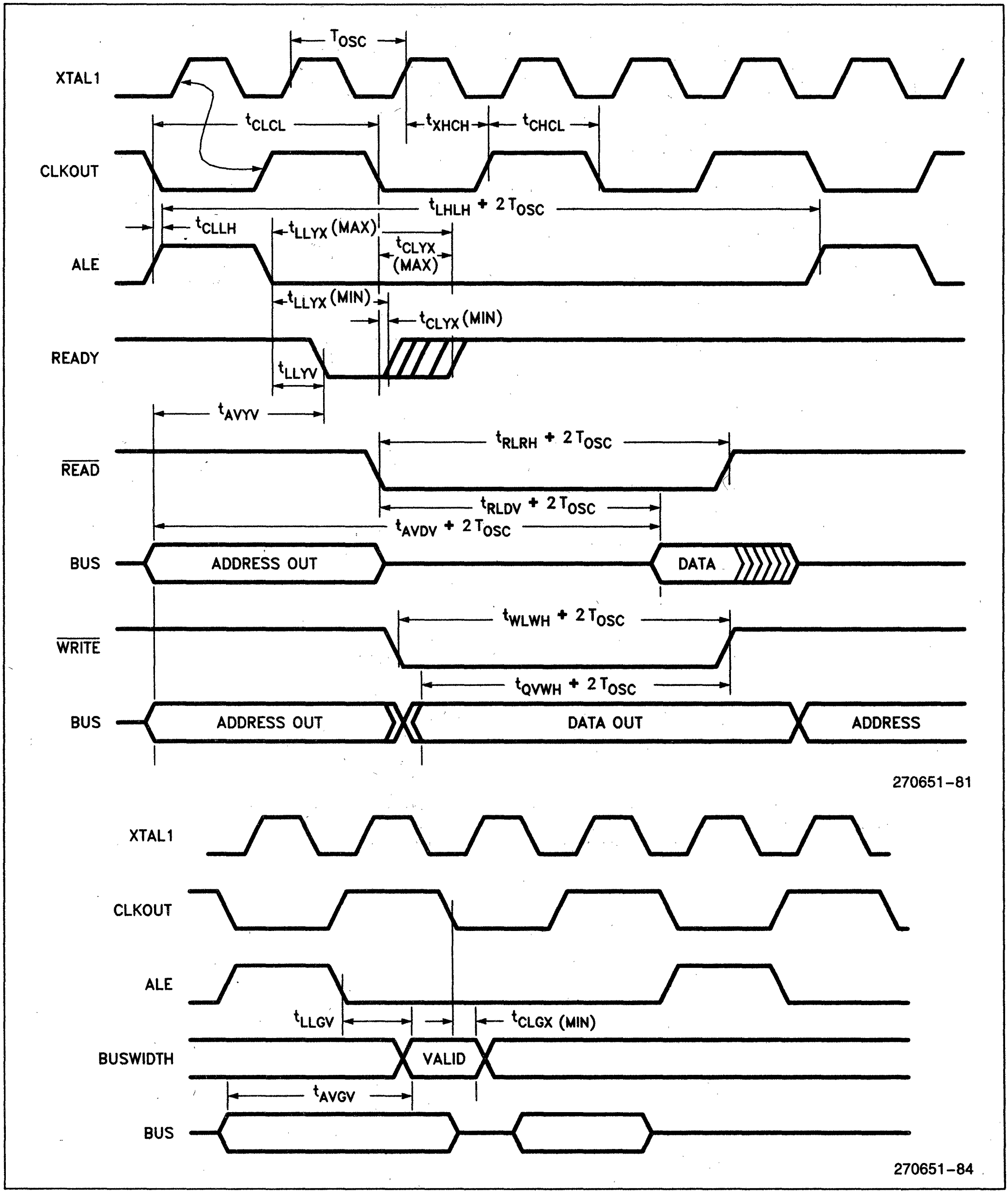


Figure 15-14. AC Timing Diagrams (Continued)

TIMINGS THE MEMORY SYSTEM MUST MEET:

T_{AVYV} — **ADDRESS Valid to READY Setup:** Maximum time the memory system has to decode READY after ADDRESS is output by the 80C196KB to guarantee at least one-wait state will occur.

T_{LLYV} — **ALE Low to READY Setup:** Maximum time the memory system has to decode READY after ALE falls to guarantee at least one wait state will occur.

T_{YLYH} — **READY Low to READY HIGH:** Maximum amount of nonREADY time or the maximum number of wait states that can be inserted into a bus cycle. Since the 80C196KB is a completely static part, T_{YLYH} is unbounded.

T_{CLYX} — **READY Hold after CLKOUT Low:** Minimum time the level on the READY pin must be valid after CLKOUT falls. The minimum hold time is always 0 ns. If maximum value is exceeded, additional wait states will occur.

T_{LLYX} — **READY Hold AFTER ALE Low:** Minimum time the level on the READY pin must be valid after ALE falls. If maximum value is exceeded, additional wait states will occur.

T_{AVGV} — **ADDRESS Valid to BUSWIDTH Valid:** Maximum time the memory system has to decode BUSWIDTH after ADDRESS is output by the 80C196KB. If exceeded, it is not guaranteed the 80C196KB will respond with an 8- or 16-bit bus cycle.

T_{LLGV} — **ALE Low to BUSWIDTH Valid:** Maximum time after ALE/$\overline{\text{ADV}}$ falls until BUSWIDTH must be valid. If exceeded, it is not guaranteed the 80C196KB will respond with an 8- or 16-bit bus cycle.

T_{CLGX} — **BUSWIDTH Hold after CLKOUT Low:** Minimum time BUSWIDTH must be held valid after CLKOUT falls. Always 0 ns of the 80C196KB.

T_{AVDV} — **ADDRESS Valid to Input Data Valid:** Maximum time the memory system has to output valid data after the 80C196KB outputs a valid address.

T_{RLDV} — **$\overline{\text{RD}}$ Low to Input Data Valid:** Maximum time the memory system has to output valid data after the 80C196KB asserts $\overline{\text{RD}}$.

T_{CLDV} — **CLKOUT Low to Input Data Valid:** Maximum time the memory system has to output valid data after the CLKOUT falls.

T_{RHDZ} — **$\overline{\text{RD}}$ High to Input Data Float:** Time after $\overline{\text{RD}}$ is inactive until the memory system must float the bus. If this timing is not met, bus contention will occur.

T_{RXDX} — **Data Hold after $\overline{\text{RD}}$ Inactive:** Time after $\overline{\text{RD}}$ is inactive that the memory system must hold Data on the bus. Always 0 ns on the 80C196KB.

TIMINGS THE 80C196KB WILL PROVIDE:

F_{XTAL} — **Frequency on XTAL1:** Frequency of signal input into the 80C196KB. The 80C196KB runs internally at ½ F_{XTAL}.

T_{OSC} — **1/F_{XTAL}:** All A.C. Timings are referenced to T_{OSC}.

T_{XHCH} — **XTAL1 High to CLKOUT High or Low:** Needed in systems where the signal driving XTAL1 is also a clock for external devices.

T_{CLCL} — **CLKOUT Cycle Time:** Nominally 2 T_{OSC}.

T_{CHCL} — **CLKOUT High Period:** Needed in systems which use CLKOUT as clock for external devices.

T_{CLLH} — **CLKOUT Falling Edge to ALE/$\overline{\text{ADV}}$ Rising:** A help in deriving other timings.

T_{LLCH} — **ALE/$\overline{\text{ADV}}$ Falling Edge to CLKOUT Rising:** A help in deriving other timings.

T_{LHLH} — **ALE Cycle Time:** Time between ALE pulses.

T_{LHLL} — **ALE/$\overline{\text{ADV}}$ High Period:** Useful in determining ALE/$\overline{\text{ADV}}$ rising edge to ADDRESS valid. External latches must also meet this spec.

T_{AVLL} — **ADDRESS Setup to ALE/$\overline{\text{ADV}}$ Falling Edge:** Length of time ADDRESS is valid before ALE/$\overline{\text{ADV}}$ falls. External latches must meet this spec.

T_{LLAX} — **ADDRESS Hold after ALE/$\overline{\text{ADV}}$ Falling Edge:** Length of Time ADDRESS is valid after ALE/$\overline{\text{ADV}}$ falls. External latches must meet this spec.

T_{LLRL} — **ALE/$\overline{\text{ADV}}$ Low to $\overline{\text{RD}}$ Low:** Length of time after ALE/$\overline{\text{ADV}}$ falls before $\overline{\text{RD}}$ is asserted. Could be needed to insure proper memory decoding takes place before a device is enabled.

Figure 15-15. AC Timing Explanations

T_{RLCL} — **$\overline{RD}$ Low to CLKOUT Falling Edge:** Length of time from $\overline{RD}$ asserted to CLKOUT falling edge: Useful for systems based on CLKOUT.

T_{RLRH} — **$\overline{RD}$ Low to $\overline{RD}$ High:** $\overline{RD}$ pulse width.

T_{RHLH} — **$\overline{RD}$ High to ALE/$\overline{ADV}$ Asserted:** Time between $\overline{RD}$ going inactive and next ALE/$\overline{ADV}$, also used to calculate time between inactive and next ADDRESS valid.

T_{RLAZ} — **$\overline{RD}$ Low to ADDRESS Float:** Used to calculate when the 80C196KB stops driving ADDRESS on the bus.

T_{LLWL} — **ALE/$\overline{ADV}$ Low Edge to $\overline{WR}$ Low:** Length of time ALE/$\overline{ADV}$ falls before $\overline{WR}$ is asserted. Could be needed to ensure proper memory decoding takes place before a device is enabled.

T_{CLWL} — **CLKOUT Falling Edge to $\overline{WR}$ Low:** Time between CLKOUT going low and $\overline{WR}$ being asserted. Useful in systems based on CLKOUT.

T_{QVWH} — **Data Valid to $\overline{WR}$ Rising Edge:** Time between data being valid on the bus and $\overline{WR}$ going inactive. Memory devices must meet this spec.

T_{CHWH} — **CLKOUT High to $\overline{WR}$ Rising Edge:** Time between CLKOUT going high and $\overline{WR}$ going inactive. Useful in systems based on CLKOUT.

T_{WLWH} — **$\overline{WR}$ Low to $\overline{WR}$ High:** $\overline{WR}$ pulse width. Memory devices must meet this spec.

T_{WHQX} — **Data Hold after $\overline{WR}$ Rising Edge:** Amount of time data is valid on the bus after $\overline{WR}$ going inactive. Memory devices must meet this spec.

T_{WHLH} — **$\overline{WR}$ Rising Edge to ALE/$\overline{ADV}$ Rising Edge:** Time between $\overline{WR}$ going inactive and next ALE/$\overline{ADV}$. Also used to calculate $\overline{WR}$ inactive and next ADDRESS valid.

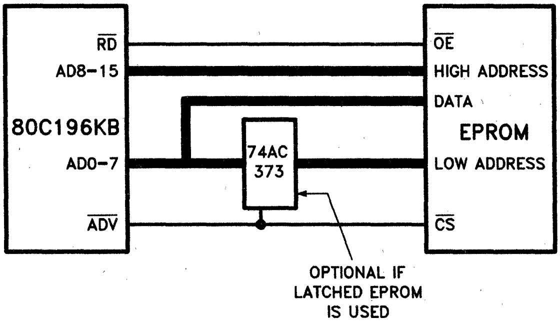
T_{WHBX} — **$\overline{BHE}$, INST, Hold after $\overline{WR}$ Rising Edge:** Minimum time these signals will be valid after $\overline{WR}$ inactive.

T_{RHBX} — **$\overline{BHE}$, INST HOLD after $\overline{RD}$ Rising Edge:** Minimum time these signals will be valid after $\overline{RD}$ inactive.

T_{WHAX} — **AD8–15 Hold after $\overline{WR}$ Rising Edge:** Minimum time the high byte of the address in 8-bit mode will be valid after $\overline{WR}$ inactive.

T_{RHAX} — **AD8–15 Hold after $\overline{RD}$ Rising Edge:** Minimum time the high byte of the address in 3-bit mode will be valid after $\overline{RD}$ inactive.

Figure 15-15. AC Timing Explanations (Continued)



270651–66

Figure 15-16. 8-Bit System with EPROM

15.6 Memory System Examples

External memory systems for the 80C196KB can be set up in many different ways. Figure 15-16 shows a simple 8 bit system with a single EPROM. The $\overline{\text{ADV}}$ Mode can be selected to provide a chip select to the memory. By setting bit CCR.1 to 0, the system is locked into the eight bit mode. An eight bit system with EPROM and RAM is shown in Figure 15-17. The EPROM is decoded in the lower half of memory,and the RAM in the upper half.

Figure 15-18 shows a 16 bit system with 2 EPROMs. Again, $\overline{\text{ADV}}$ is used to chip select the memory. Figure 15-19 shows a system with dynamic bus width. Code is executed from the two EPROMs and data is stored in the single RAM. Note the Chip Select of the RAM also is input to the BUSWIDTH pin to select an eight bit cycle.

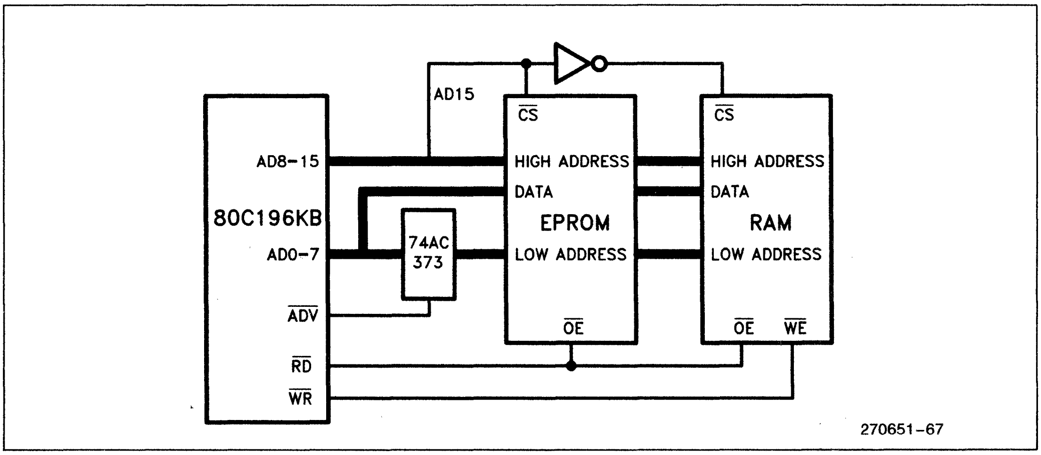


Figure 15-17. 8-Bit System with EPROM and RAM

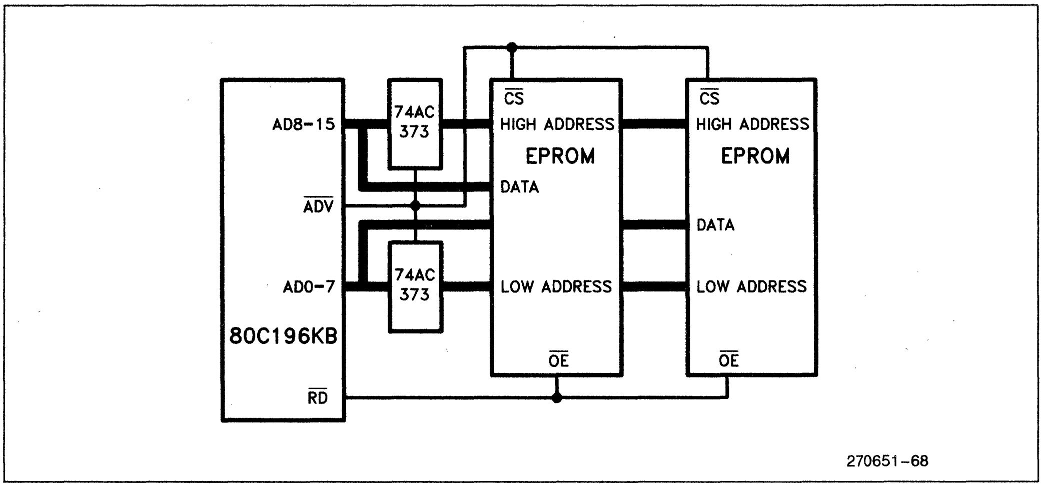


Figure 15-18. 16-Bit System with EPROM

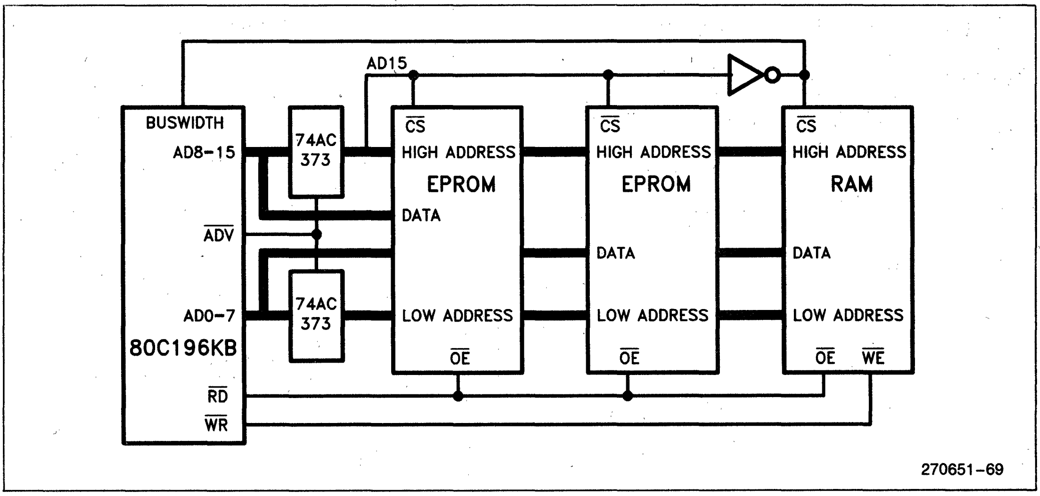


Figure 15-19. 16-Bit System with Dynamic Buswidth

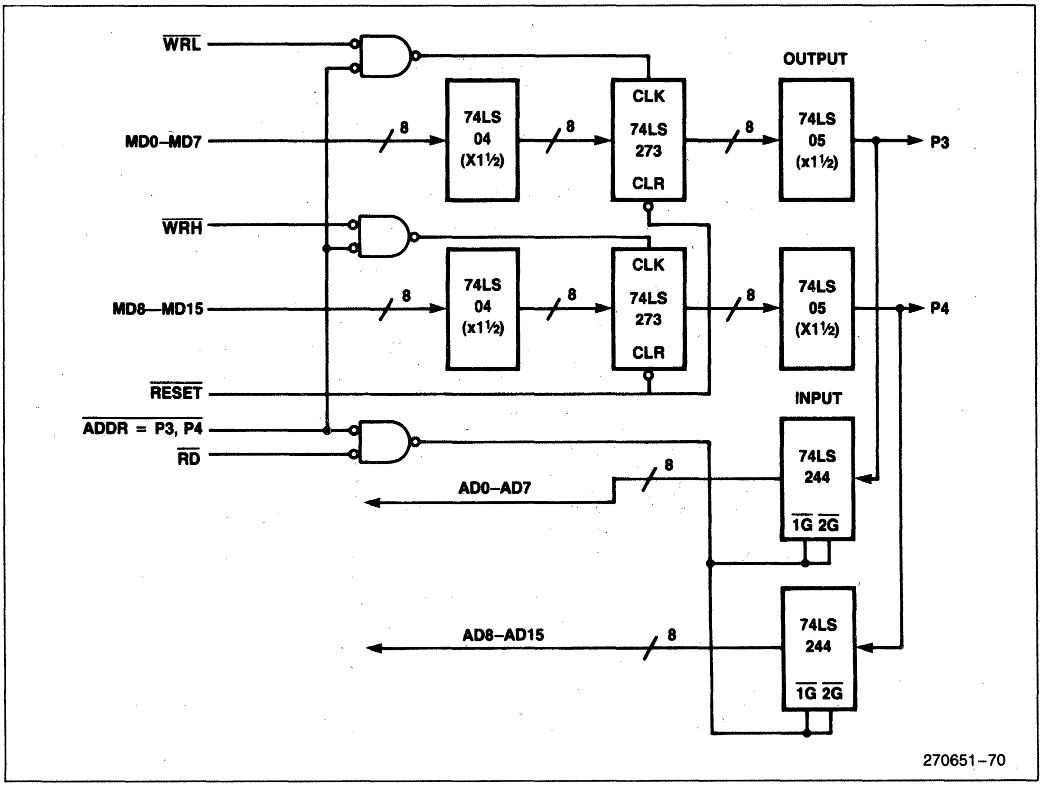


Figure 15-20. I/O Port Reconstruction

15.7 I/O Port Reconstruction

When a single-chip system is being designed using a multiple chip system as a prototype, it may be necessary to reconstruct I/O Ports 3 and 4 using a memory mapped I/O technique. The circuit to reconstruct the Ports is shown in Figure 15-20. It can be attached to a 80C196KB system which has the required address decoding and bus demultiplexing.

The output circuitry is a latch that operates when 1FFEH or 1FFFH are placed on the MA lines. The inverters surrounding the latch create an open-collector output to emulate the open-drain output found on the 80C196KB. The $\overline{\text{RESET}}$ line sets the ports to all 1s when the chip is reset. The voltage and current specifications of the port will be different from the 80C196KB, but the functionality will be the same.

The input circuitry is a bus transceiver that is addressed at 1FFEH and 1FFFH. If the ports are going to be either inputs or outputs, but not both, some of the circuitry may be eliminated.

16.0 USING ROM AND EPROM DEVICES

Three flexible EPROM programming modes are available on the 87C196KB; Auto, Slave, and Run-time. These modes can program the 87C196KB in a stand alone, gang, or run-time environment.

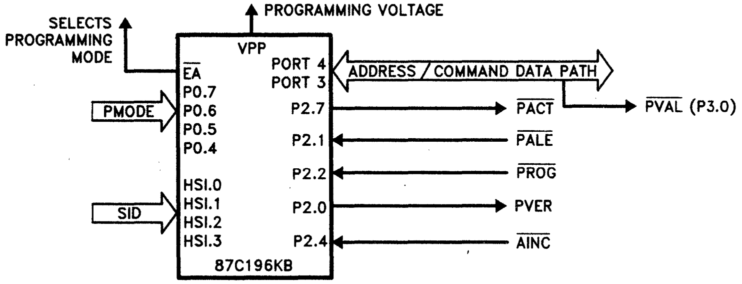
The 87C196KB contains 8K bytes of ultraviolet Erasable and Electrically Programmable Read Only Memory (EPROM). The 83C196KB contains 8K bytes of Read Only Memory (ROM).

16.1 Programming the 87C196KB

The EPROM is mapped into memory locations 2000H-3FFFH if $\overline{\text{EA}}$ is at logical 1. However, applying +12.75V to $\overline{\text{EA}}$ when $\overline{\text{RESET}}$ is asserted will place the 87C196KB in Programming Mode. The Programming Mode has been implemented to support programming as well as verification of 87C196KB EPROMs.

The Auto Programming Mode enables an 87C196KB to program itself with the 8K bytes of code beginning at address 4000H on its external bus. The Slave Mode provides a standard interface that enables any number of 87C196KBs to be programmed by a master device such as an EPROM programmer. The Run-time Mode allows individual EPROM locations to be programmed at run-time under complete software control.

In the Programming Mode, some I/O pins have new functions. These new pin functions determine and support the different programming modes. Figure 16-1 shows how the pins are renamed and Figure 16-2 describes in detail each new pin function.



270651-71

Figure 16-1. Programming Mode Pin Functions

Name	Function
PMODE	Programming Mode Select. Determines the EPROM programming algorithm that is performed. PMODE is sampled after a chip reset and should be static while the part is operating.
SID	Slave ID Number. Used to assign each slave a pin of Port 3 or 4 to use for passing programming verification acknowledgement. For example, if gang programming in the Slave Programming Mode, the slave with SID = 001 will use Port 3.1 to signal correct or incorrect program verification.
$\overline{\text{PALE}}$	Programming ALE Input. Accepted by an 87C196KB that is in Slave Programming Mode. Used to indicate that Ports 3 and 4 contain a command/address.
$\overline{\text{PROG}}$	Programming Input. Falling edge indicates valid data on PBUS and the beginning of programming. Rising edge indicates end of programming.
$\overline{\text{PACT}}$	Programming Active Output. Used in the Auto Programming Mode to indicate when programming activity is complete.
PVER	Program Verification Output. Signal is low after rising edge of PROG if the programming was not successful.
$\overline{\text{PVAL}}$	Program Valid Output. This output signal indicates the success or failure of programming in the Auto Programming Mode. A zero indicates successful programming.
$\overline{\text{AINC}}$	Auto Increment Input. Active low input signal indicates that the auto increment mode is enabled. Auto Increment will allow reading or writing of sequential EPROM locations without address transactions across the PBUS for each read or write.
PORTS	Address/Command/Data Bus. Used to pass commands, addresses and data to and from slave mode 87C196KBs. Used by chips in Auto Programming Mode to pass command, addresses and data to slaves. Also used in the Auto Programming Mode as a regular system bus to access external memory. Should have pullups to V_{CC} (15 kΩ).

Figure 16-2. Programming Mode Pin Definitions

When going into Programming Mode, PMODE selects the programming function (see Figure 16-3). PMODE does not affect run-time programming.

PMODE	Programming Mode
0–4	Reserved
5	Slave Programming
6	ROM Dump Mode
7–0BH	Reserved
0CH	Auto Programming
0DH	Program Configuration Byte
0EH–0FH	Reserved

Figure 16-3. Programming Function Pmode Values

To guarantee proper functionality, the pins of PMODE and SID must be in their desired state before $\overline{\text{RESET}}$ rises. Once the part is reset, it should not be switched to another mode without a new reset sequence.

When $\overline{\text{EA}}$ selects the Programming Mode, the chip reset sequence loads the CCR from the PCCB (Programming Chip Configuration Byte). The PCCB is a separate EPROM location that is not mapped under normal operation. PCCB is important when in the Auto Programming Mode and for the memory protection options, discussed later. In this mode, the 87C196KB gets the programming data over the external bus. Therefore, the PCCB must correctly correspond to the memory system in the programming setup, which is not necessarily the memory system of the application.

The following sections describe the 87C196KB programming modes in detail.

PROGRAMMING PULSE WIDTH REGISTER (PPW)

When the 87C196KB programs itself the width of the programming pulse is determined by the 8 bit PPW (Programming Pulse Width) register. In the Auto Programming Mode, the PPW is loaded from location 4014H in external memory. In Run-time Programming Mode, the PPW is located in window 14 at 04H. In order for the EPROM to properly program, the pulse width must be set to approximately 100 uS. The pulse width is dependent on the oscillator frequency and is calculated with the following formula:

$$\text{Pulse Width} = \text{PPW} * (\text{Tosc} * 8)$$

$$\text{PPW} = 150 \text{ @ } 12 \text{ Mhz}$$

16.2 Auto Programming Mode

The Auto Programming Mode allows an 87C196KB EPROM to be programmed without a special EPROM programmer. In this mode, the 87C196KB simply programs itself with the data at external locations 4000H-5FFFH. Location 4014H in external memory contains the value that is loaded into the PPW. All that is required is that of external memory reside at these locations, $\overline{\text{EA}}$ is at 12.75V, the PPW is in location 4014H, and Vpp is applied. Figure 16-4 shows a minimum configuration using an 8K × 8 EPROM to program an 87C196KB in the Auto Programming Mode.

In the Auto Programming Mode, CCR is loaded with the PCCB. The PCCB must correspond to the memory system of the programming setup, which is not necessarily the same as the memory system of the actual application.

The 87C196KB reads a word from external memory, then the Modified Quick-Pulse Programming Algorithm (described later) programs the corresponding EPROM location. Since the erased state of a byte is 0FFH, the Auto Programming Mode will skip locations with 0FFH for data. When all 8K have been programmed, $\overline{\text{PACT}}$ goes high and the part outputs a 0 on $\overline{\text{PVAL}}$ (P3.0) if it programmed correctly and a 1 if it failed.

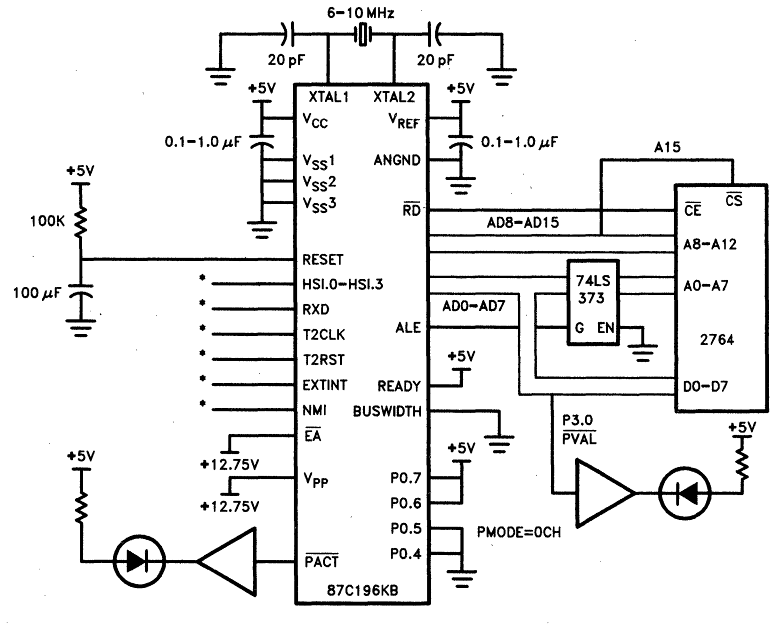
V_{PP} and EA must be kept noise-free and must never go above 12.75V at any time.

PCCB Programming Mode

This is the only way to program the PCCB. PCCB is a non-memory mapped location that gets loaded into the CCR when the 87C196KB is placed in Programming Mode. If the PCCB is not programmed, the CCR will be loaded with 0FFH when entering the Programming Mode. Figure 16-5 shows a block diagram for the PCCB Programming Mode.

With PMODE = 0DH and 0FF on Port 4, the CCB and PCCB will be programmed with the data on Port 3 when a 0 is placed on PALE. After programming is complete, PVER is driven high if the bytes programmed correctly, and low if the programming failed.

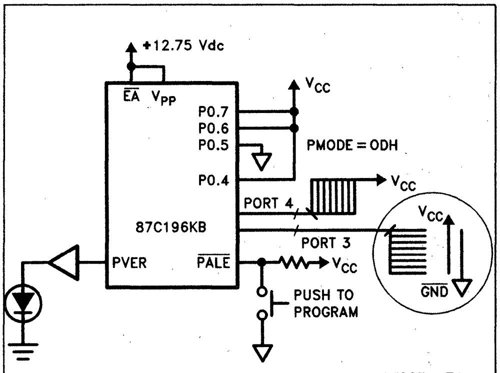
This mode can also program the CCB. The CCB (2018H) can be treated just like any other EPROM location, and programmed using any programming mode. But the PCCB Programming Mode allows a simple way to program the CCB when no other locations need be programmed.



270651–72

NOTE:
*Inputs must be driven high or low.

Figure 16-4. Auto Programming Mode



NOTES:
Tie Port 3 to the value desired to be programmed into CCB and PCCB.
Make all necessary minimum connections for power, ground and clock.

Figure 16-5. The PCCB Programming Mode

16.3 Slave Programming Mode

Any number of 87C196KBs can be programmed by a master programmer through the Slave Programming Mode.

In this mode, the 87C196KB programs like a simple EPROM device. The 87C196KB responds to three different commands while in this mode: data program, data verify, and word dump. These commands, along with the transfer of appropriate data and addresses are selected using Ports 3 and 4 and five other pins for handshaking. The two most significant bits on Ports 3 and 4 specify the command and the lower 14 bits contain the address. The address ranges from 2000H-3FFFH and refers to internal memory space. Figure 16-6 is a list of valid Programming Commands.

P4.7	P4.6	Action
0	0	Word Dump
0	1	Data Verify
1	0	Data Program
1	1	Reserved

Figure 16-6. Slave Programming Mode Commands

The 87C196KB receives an input signal, PALE, to indicate a valid command is present. PROG causes the 87C196KB to read in or output a data word. An output signal, PVER, indicates if the programming was successful. AINC automatically increments the address for the Data Program and Word Dump commands. There is no 87C196KB dependent limit to the number of parts that can be gang programmed in the Slave Mode.

Data Program Command

A Data Program Command is illustrated in Figure 16-7. Asserting PALE latches the command and address on Ports 3 and 4. PROG is asserted to latch the data present on Ports 3 and 4. PROG also starts the actual programming sequence. The width of the PROG pulse determines the programming pulse width.

After the rising edge of PROG, the slaves automatically perform a verification of the address just programmed. PVER is asserted if the location programmed correctly. This gives verification information to programmers which can not use the Data Verify Command. The AINC pin can increment to the next location or a new Data Program Command can be issued.

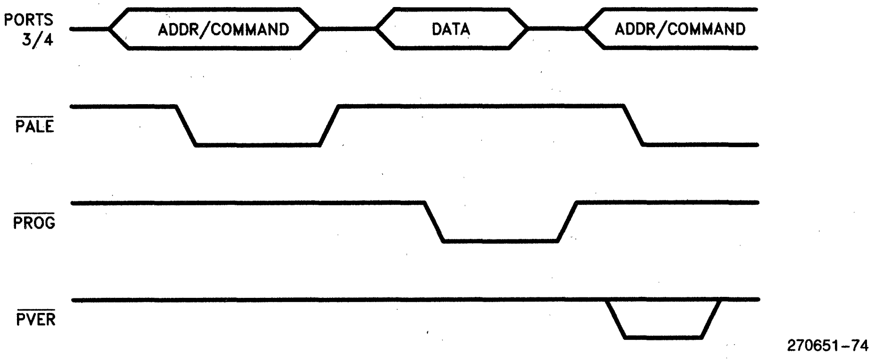


Figure 16-7. Data Program Command in Slave Mode

If PVER of all slaves are 1s after PROG rises then the data program was successful for every slave. If PVER is a 0 for any slave, then the part did not program correctly. Figure 16-7 shows the relationship of PALE, PROG, and PVER to the Command/Data path on Ports 3 and 4 for the Data Program Command.

Data Verify Command

When the Data Verify Command is sent, the slaves respond by driving one bit of Ports 3 and 4 to indicate correct or incorrect verification of the previous Data Program Command. A 1 indicates a correct verification, and a 0 indicates incorrect verification. The SID (Slave I.D) of each slave determines which bit of Ports 3 and 4 will be driven. For example, a SID of 0001 would drive Port 3.1. PROG governs when the slaves drive the bus. Figure 16-8 shows the relationship of ports 3 and 4 to PALE and PROG.

A Data Verify Command is always preceded by a Data Program Command in a programming system with as many as 16 slaves. However, a Data Verify Command does not have to follow every Data Program Command.

Word Dump Command

When the Word Dump Command is issued, the 87C196KB adds 2000H to the address field of the command and places the value at the new address on Ports 3 and 4. For example, sending the command 0100H to a slave results in the slave placing the word at internal address 2100H on Ports 3 and 4. PROG governs when the slave drives the bus. The Timings are the same as shown in Figure 16-7.

Note that the Word Dump Command will work only when just one slave is attached to the bus. Also, there is no restriction on commands that precede or follow a Word Dump Command.

Gang Programming With the Slave Programming Mode.

Gang Programming of 87C196KBs can be done using the Slave Programming Mode. There is no 87C196KB based limit on the number of chips that may be hooked to the same Port 3 and 4 data path for gang programming.

If more than 16 chips are being gang programmed, the PVER outputs of each chip could be used for verification. The master programmer could issue a Data Program Command, then either watch every chip's error signal, or AND all the signals together to form a system PVER.

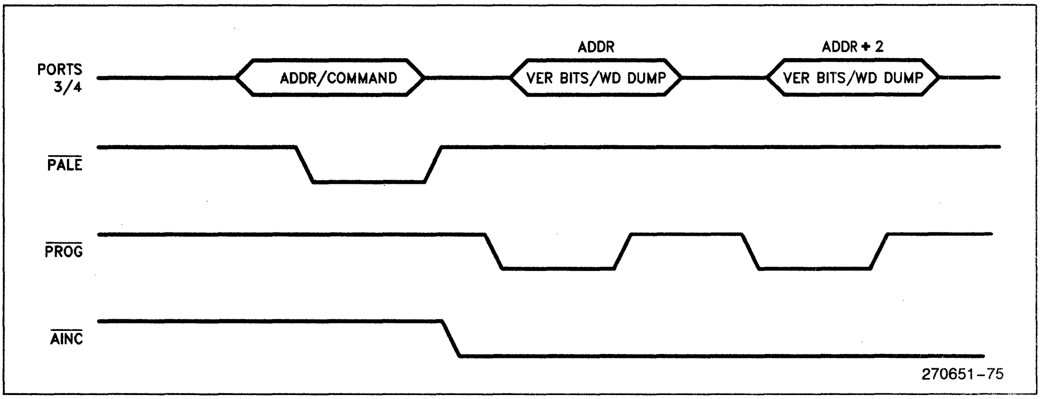


Figure 16-8. Ports 3 and 4 to PALE and PROG

If 16 or fewer 87C196KBs are to be Gang programmed at once, a more flexible form of verification is available. By giving each chip a unique SID, the master programmer could issue a Data Verify Command after the Data Program Command. When a verify command is seen by the slaves, each will drive a bit of Ports 3 or 4 corresponding to its unique SID. A 1 driven signals the address verified, while a 0 means it failed.

16.4 Run-Time Programming Mode

Run-Time Programming allows the user complete flexibility in the ways the internal EPROM is programmed. That flexibility includes the ability to program just one byte or word instead of the whole EPROM. The only additional requirement of a system doing Run-Time Programming is that programming voltage is applied V_{PP}. Run-Time Programming is done with $\overline{EA}$ at a TTL high.

To Run-Time Program, the user writes to the location to be programmed. The value of the PPW register determines the programming pulse. To ensure 87C196KC compatibility, the Idle Mode should be used for Run-Time Programming. Figure 16-9 is the recommended code sequence for Run-Time Programming. The Modified Quick Pulse algorithm guarantees the programmed EPROM cell for the life of the part.

16.5 ROM/EPROM Memory Protection Options

Write protection is available for EPROM parts, and read protection is provided for both ROM and EPROM parts.

Write protection is enabled by clearing the LOC0 bit in the CCR. When write protection is enabled, the bus controller will cycle through the write sequence but will not actually drive data to the EPROM or enable V_{PP} to the EPROM. This protects the entire EPROM 2000-3FFFH from inadvertent or unauthorized programming.

Read protection is enabled by clearing the LOC1 bit of the CCR. When read protection is selected, the bus controller will only perform a data read from the address range 2020H-202FH (Security Key) and 2040H-3FFFH if the Slave Program Counter is in the range 2000H-3FFFH. Since the Slave PC can be as many as 4 bytes ahead of the CPU program counter, an instruction after address 3FFAH may not access protected memory. Also note the interrupt vectors and CCB are not read protected.

$\overline{EA}$ is latched on reset so the device cannot be switched from internal to external memory by toggling $\overline{EA}$.

```
                LD WSR,#14                          ;Initialize programmable
                LD PPW,#VALUE                       ;pulse width

PROGRAM:        POP ADDRESS_TEMP                    ;Load program data
                POP DATA_TEMP                       ;and address
                PUSHF
                LD COUNT, #25T                      ;program using Modified
                                                     Quick Pulse
   LOOP:      * LDB INT_MASK,#ENABLE_SWT            ;program SWT for
              * LDB HSO_COMMAND,#SWTO_OVF           ;program pulse width
              * ADD HSO_TIME,TIMER1,#PROGRAM_PULSE
              * EI
                ST DATA_TEMP,[ADDR_TEMP]            ;enter idle mode until
              * IDLPD 1                             ;swt expires
                DJNZ COUNT, LOOP                    ;loop 25 times
                POPF
                RET

SWT_EXPIRED:    *POP 0                              ;service swt and return
               *RET
```

NOTE:
*Not Really Needed on Current 87C196KB Part

Figure 16-9. Future Run-Time Programming Algorithm

If the CCR has any protection enabled, the security key is write protected to keep unauthorized users from overwriting the key with a known security key.

Even though the PCCB is loaded into the CCR at reset, the device still checks the lock bits programmed into the CCB.

NOTE:

Substantial effort has been made to provide an excellent program protection scheme. However, Intel cannot and does not guarantee that these protection methods will always prevent unauthorized access.

Authorized Access of Protected Memory

A "Security Key" mechanism has been implemented for authorized access of protected internal memory to test internal ROM/EPROM. This allows users to verify the EPROM array and still keep their code protected.

The security key is a 128 bit number located in internal memory at locations 2020H-202FH The security key must be matched before a ROM dump will occur.

The ROM Dump Mode is entered just like any other Programming Mode with PMODE = 6H. For the 87C196KB, $\overline{EA}$ should be = 12.75V. Because the ROM process does not support putting a high voltage on $\overline{EA}$, tthe ROM dump Mode for the 83C196KB cannot be entered by putting 12.75V on $\overline{EA}$. Instead, the Programming Modes are entered by holding ALE low, and INST, $\overline{RD}$, and $\overline{EA}$ high on the rising edge of RESET. After this, the ROM dump mode functions as expected. Write out the entire EPROM or ROM to locations 4000H-5FFFH in external memory.

The ROM Dump Mode begins with a security key verification. The user puts the same security key at external locations 4020H-402FH that is in internal locations 2020H-202FH. Before doing a ROM dump, the 87C196KB compares the two security keys. If they do not match, the 87C196KB enters an endless loop of internal execution.

When using the Auto Configuration byte or Auto Programming Modes, a security key verification is done if the CCB has read and or write protection enabled.

If the PCCB is programmed with any read or write protection, there is no way to enter any of the programming modes. So the last thing that should be done to protect the part from unauthorized access, is to program the PCCB.

16.6 Algorithms

The Modified Quick Pulse Algorithm

The Modified Quick Pulse Algorithm must be used to guarantee programming over the life of the EPROM in Run-time and Slave Programming Modes.

The Modified Quick-Pulse Algorithm calls for each EPROM location to receive 25 separate 100 uS (±5 μs) programming cycles. Verification is done after the 25th pulse. If the location verifies, the next location is programmed. If the location fails to verify, the location fails the programming sequence.

Once all locations are programmed and verified, the entire EPROM is again verified.

Programming of 87C196KB EPROMs is done with V_{PP} = 12.75V ±0.25V and V_{CC} = 5.0V ±0.5V.

Signature Word

The 87C196KB contains a signature word at location 2070H. The word can be accessed in the Slave Mode by executing a Word Dump Command. The programming voltages are determined by reading the test ROM at locations 2072H and 2073H. The voltages are calculated by using the following equation.

Voltage = 20/256 * (test ROM data)

The values for the signature word and voltage levels are shown in Figure 16-10.

Description	Location	Value
Signature Word	2070H	897CH
Programming V_{CC}	2072H	040H (5.0V)
Programming V_{PP}	2073H	0A3H (12.75V)

Figure 16-10. Signature Word and Voltage Levels

Erasing the 87C196KB

After each erasure, all bits of the 87C196KB are logical ‘1s‘. Data is introduced by selectively programming ‘0s‘. The only way to change a ‘0‘ to a ‘1‘ is by exposure to ultraviolet light.

The erasure characteristics of the 87C196KB are so that erasure begins upon exposure to light with wavelengths shorter than approximately 4000 Angstroms. It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000-4000 Angstrom range. Constant exposure to room level fluorescent lighting could erase a 87C196KB in about 3 years. It would take about 1 week in direct sunlight to erase an 87C196KB.

Opaque labels should always be placed over the window to prevent unintentional erasure. In the Powerdown Mode, the part will draw more current than normal if the EPROM window is exposed to light.

The recommended erasure procedure for the 87C196KB is exposure to ultraviolet light which has a wavelength of 2537 Angstroms. The integrated dose (UV intensity * exposure time) should be a minimum of 15 Wsec/cm^2. The total time for erasure is about 15 to 20 minutes at this level of exposure. The 87C196KB should be placed within 1 inch of the lamp during exposure. The maximum integrated dose an 87C196KB can be exposed to without damage is 7258 Wsec/cm^2 (1 week @ 12000 uW/cm^2). Exposure to UV light greater than this can cause permanent damage.

ADVANCE INFORMATION

83C196KB/80C196KB 16-BIT HIGH PERFORMANCE CHMOS MICROCONTROLLER

83C196KB — 8 Kbytes of Factory Mask-Programmed ROM
80C196KB — ROMless

- **232 Byte Register File**
- **Register-to-Register Architecture**
- **28 Interrupt Sources/16 Vectors**
- **2.3 μs 16 x 16 Multiply (12 MHz)**
- **4.0 μs 32/16 Divide (12 MHz)**
- **Powerdown and Idle Modes**
- **Five 8-Bit I/O Ports**
- **16-Bit Watchdog Timer**
- **Dynamically Configurable 8-Bit or 16-Bit Buswidth**
- **Full Duplex Serial Port**
- **High Speed I/O Subsystem**
- **16-Bit Timer**
- **16-Bit Up/Down Counter with Capture**
- **Pulse-Width-Modulated Output**
- **Four 16-Bit Software Timers**
- **10-Bit A/D Converter with Sample/Hold**
- **$\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ Bus Protocol**
- **12 MHz Version — 83C196KB12/80C196KB12**
  10 MHz Version — 83C196KB10/80C196KB10

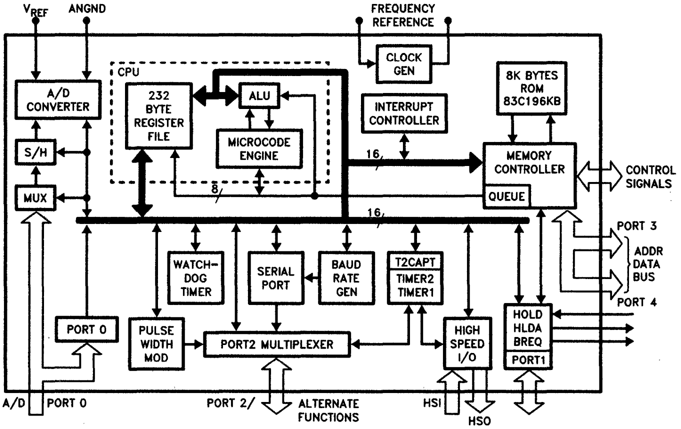
The 80C196KB 16-bit microcontroller is a high performance member of the MCS®-96 microcontroller family. The 80C196KB is compatible with the 8096BH and uses a true superset of the 8096BH instructions. Intel's CHMOS process provides a high performance processor along with low power consumption. To further reduce power requirements, the processor can be placed into Idle or Powerdown Mode.

The 83C196KB is an 80C196KB with 8 Kbytes on-chip ROM. In this document, the 80C196KB will refer to both products unless otherwise stated.

Bit, byte, word and some 32-bit operations are available on the 80C196KB. With a 12 MHz oscillator a 16-bit addition takes 0.66 μs, and the instruction times average 0.5 μs to 1.5 μs in typical applications.

Four high-speed capture inputs are provided to record times when events occur. Six high-speed outputs are available for pulse or waveform generation. The high-speed output can also generate four software timers or start an A/D conversion. Events can be based on the timer or up/down counter.

Also provided on-chip are an A/D converter, serial port, watchdog timer, and a pulse-width-modulated output signal.



270634–1

Figure 1. 83C196KB/80C196KB Block Diagram

MCS®-96 is a registered trademark of Intel Corporation.

ARCHITECTURE

The 80C196KB is a member of the MCS®-96 family, and as such has the same architecture and uses the same instruction set as the 8096BH. Many new features have been added on the 80C196KB including:

CPU FEATURES

Divide by 2 instead of divide by 3 clock for 1.5X performance

Faster instructions, especially indexed/indirect data operations

2.33 μs 16 × 16 multiply with 12 MHz clock (was 6.25 μs on the 8096BH)

Faster interrupt response (almost twice as fast as 8096BH)

Powerdown and Idle Modes

5 new instructions including Compare Long and Block Move

8 new interrupt vectors/6 new interrupt sources

PERIPHERAL FEATURES

SFR Window switching allows read-only registers to be written and vice-versa

Timer2 can count up or down by external selection

Timer2 has an independent capture register

HSO line events are stored in a register

HSO has CAM Lock and CAM Clear commands

New Baud Rate values are needed for serial port, higher speeds possible in all modes

Double buffered serial port transmit register

Serial Port Receive Overrun and Framing Error Detection

PWM has a Divide-by-2 Prescaler

$\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ Bus Protocol

PACKAGING

The 80C196KB and 83C196KB are available in a 68-pin PLCC package and an 80-pin QFP package. In addition, the 80C196KB is available in a 68-pin PGA package. Contact your local sales office to determine the exact ordering code for the part desired.

The 8XC196TB is an 8XC196KB without the A/D converter, and is only available in the PLCC package.

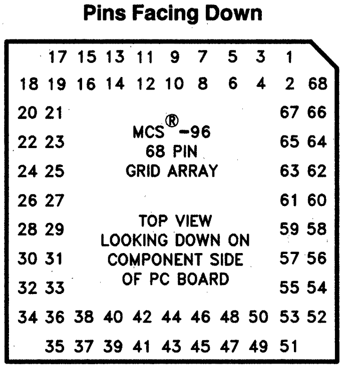
	With A/D	Without A/D
ROMless	N80C196KB—PLCC A80C196KB—PGA S80C196KB—QFP	N80C196TB—PLCC
ROM	N83C196KB—PLCC S83C196KB—QFP	N83C196TB—PLCC

PGA	PLCC	Description	PGA	PLCC	Description	PGA	PLCC	Description
1	9	ACH7/P0.7	24	54	AD6/P3.6	47	31	P1.6/$\overline{\text{HLDA}}$
2	8	ACH6/P0.6	25	53	AD7/P3.7	48	30	P1.5/$\overline{\text{BREQ}}$
3	7	ACH2/P0.2	26	52	AD8/P4.0	49	29	HSO.1
4	6	ACH0/P0.0	27	51	AD9/P4.1	50	28	HSO.0
5	5	ACH1/P0.1	28	50	AD10/P4.2	51	27	HSO.5/HSI.3
6	4	ACH3/P0.3	29	49	AD11/P4.3	52	26	HSO.4/HSI.2
7	3	NMI	30	48	AD12/P4.4	53	25	HSI.1
8	2	$\overline{\text{EA}}$	31	47	AD13/P4.5	54	24	HSI.0
9	1	V_{CC}	32	46	AD14/P4.6	55	23	P1.4
10	68	V_{SS}	33	45	AD15/P4.7	56	22	P1.3
11	67	XTAL1	34	44	T2CLK/P2.3	57	21	P1.2
12	66	XTAL2	35	43	READY	58	20	P1.1
13	65	CLKOUT	36	42	T2RST/P2.4/$\overline{\text{AINC}}$	59	19	P1.0
14	64	BUSWIDTH	37	41	$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	60	18	TXD/P2.0
15	63	INST	38	40	$\overline{\text{WR}}$/$\overline{\text{WRL}}$	61	17	RXD/P2.1
16	62	ALE/$\overline{\text{ADV}}$	39	39	PWM/P2.5	62	16	$\overline{\text{RESET}}$
17	61	$\overline{\text{RD}}$	40	38	P2.7/T2CAPTURE/$\overline{\text{PACT}}$	63	15	EXTINT/P2.2
18	60	AD0/P3.0	41	37	V_{PP}	64	14	V_{SS}[1]
19	59	AD1/P3.1	42	36	V_{SS}	65	13	V_{REF}
20	58	AD2/P3.2	43	35	HSO.3/SID3	66	12	ANGND
21	57	AD3/P3.3	44	34	HSO.2/SID2	67	11	ACH4/P0.4
22	56	AD4/P3.4	45	33	P2.6/T2UP-DN	68	10	ACH5/P0.5
23	55	AD5/P3.5	46	32	P1.7/$\overline{\text{HOLD}}$			

Figure 2. Pin Definitions

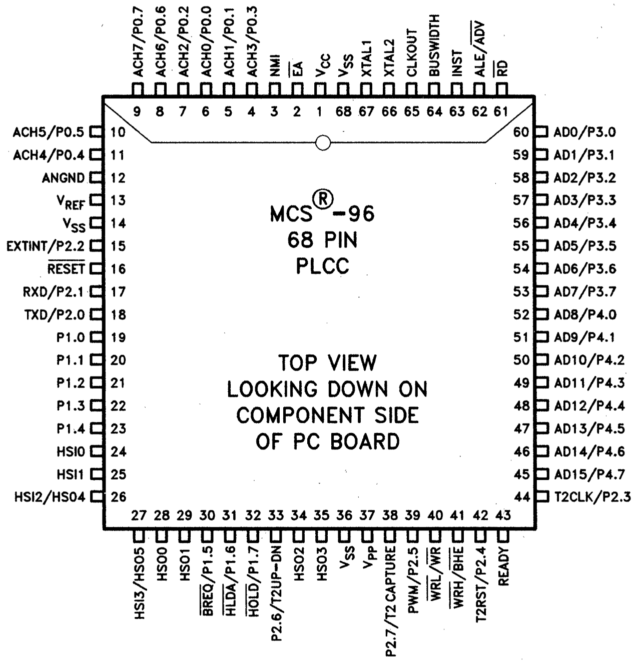
NOTE:

1. This pin was formerly the Clock Detect Enable pin. This function is not guaranteed to work. This pin must be directly connected to V_{SS}.



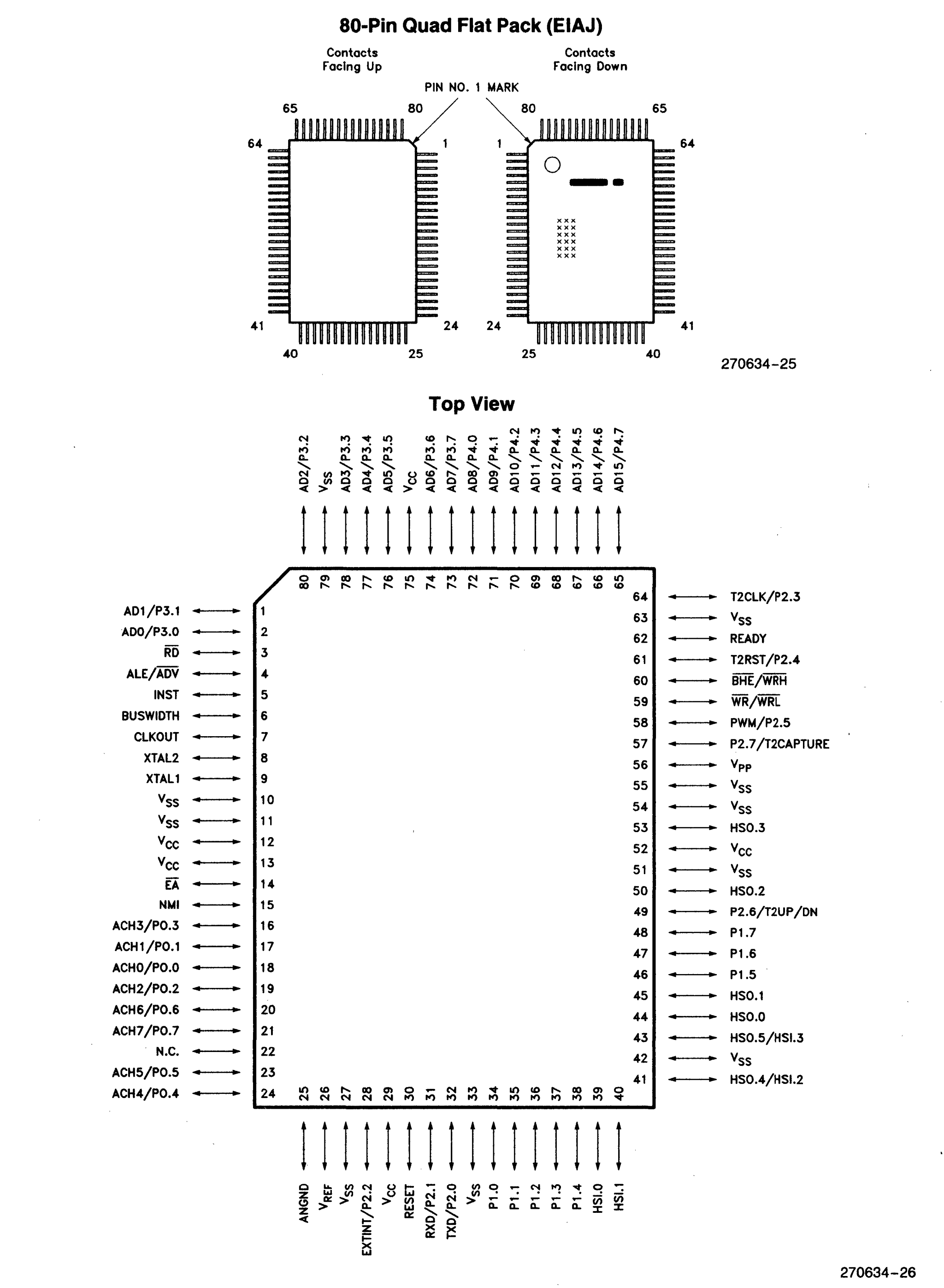
270634-23

Figure 3. 68-Pin Package (Pin Grid Array — Top View) 80C196KB Only



270634-2

Figure 4. 68-Pin Package (PLCC—Top View)



80-Pin Quad Flat Pack (EIAJ)

Top View

Figure 5. 80-Pin Quad Flat Pack (QFP)

Thermal Characteristics

Package Type	θ_{ja}	θ_{jc}
PGA	28°C/W	3.5°C/W
PLCC	35°C/W	12°C/W
QFP	85°C/W	—

PIN DESCRIPTIONS

Symbol	Name and Function
V_{CC}	Main supply voltage (5V).
V_{SS}	Digital circuit ground (0V). There are two V_{SS} pins, both of which must be connected.
V_{REF}	Reference voltage for the A/D converter (5V). V_{REF} is also the supply voltage to the analog portion of the A/D converter and the logic used to read Port 0. Must be connected for A/D and Port 0 to function.
ANGND	Reference ground for the A/D converter. Must be held at nominally the same potential as V_{SS}.
V_{PP}	Timing pin for the return from powerdown circuit. Connect this pin with a 1 µF capacitor to V_{SS}. If this function is not used V_{PP} may be tied to V_{CC}. This pin is the programming voltage on the EPROM device.
XTAL1	Input of the oscillator inverter and of the internal clock generator.
XTAL2	Output of the oscillator inverter.
CLKOUT	Output of the internal clock generator. The frequency of CLKOUT is ½ the oscillator frequency. It has a 50% duty cycle.
$\overline{\text{RESET}}$	Reset input to the chip. Input low for at least 4 state times to reset the chip. The subsequent low-to-high transition re- synchronizes CLKOUT and commences a 10-state-time sequence in which the PSW is cleared, a byte read from 2018H loads CCR, and a jump to location 2080H is executed. Input high for normal operation. $\overline{\text{RESET}}$ has an internal pullup.
BUSWIDTH	Input for buswidth selection. If CCR bit 1 is a one, this pin selects the bus width for the bus cycle in progress. If BUSWIDTH is a 1, a 16-bit bus cycle occurs. If BUSWIDTH is a 0 an 8-bit cycle occurs. If CCR bit 1 is a 0, the bus is always an 8-bit bus.
NMI	A positive transition causes a vector through 203EH.
INST	Output high during an external memory read indicates the read is an instruction fetch. INST is valid throughout the bus cycle. INST is activated only during external memory accesses and output low for a data fetch.
$\overline{\text{EA}}$	Input for memory select (External Access). $\overline{\text{EA}}$ equal to a TTL-high causes memory accesses to locations 2000H through 3FFFH to be directed to on-chip ROM/EPROM. $\overline{\text{EA}}$ equal to a TTL-low causes accesses to these locations to be directed to off-chip memory. $\overline{\text{EA}}$ must be tied low for the 80C196KB ROMless device.
ALE/$\overline{\text{ADV}}$	Address Latch Enable or Address Valid output, as selected by CCR. Both pin options provide a latch to demultiplex the address from the address/data bus. When the pin is $\overline{\text{ADV}}$, it goes inactive high at the end of the bus cycle. $\overline{\text{ADV}}$ can be used as a chip select for external memory. ALE/$\overline{\text{ADV}}$ is activated only during external memory accesses.
$\overline{\text{RD}}$	Read signal output to external memory. $\overline{\text{RD}}$ is activated only during external memory reads.
$\overline{\text{WR}}$/$\overline{\text{WRL}}$	Write and Write Low output to external memory, as selected by the CCR. $\overline{\text{WR}}$ will go low for every external write, while $\overline{\text{WRL}}$ will go low only for external writes where an even byte is being written. $\overline{\text{WR}}$/$\overline{\text{WRL}}$ is activated only during external memory writes.

PIN DESCRIPTIONS (Continued)

Symbol	Name and Function
$\overline{\text{BHE}}/\overline{\text{WRH}}$	Bus High Enable or Write High output to external memory, as selected by the CCR. $\overline{\text{BHE}}$ =0 selects the bank of memory that is connected to the high byte of the data bus. A0 = 0 selects the bank of memory that is connected to the low byte of the data bus. Thus accesses to a 16-bit wide memory can be to the low byte only (A0 = 0, $\overline{\text{BHE}}$ = 1), to the high byte only (A0 = 1, $\overline{\text{BHE}}$ = 0), or both bytes (A0 = 0, $\overline{\text{BHE}}$ = 0). If the $\overline{\text{WRH}}$ function is selected, the pin will go low if the bus cycle is writing to an odd memory location. $\overline{\text{BHE}}/\overline{\text{WRH}}$ is valid only during 16-bit external memory write cycles.
READY	Ready input to lengthen external memory cycles, for interfacing to slow or dynamic memory, or for bus sharing. If the pin is high, CPU operation continues in a normal manner. If the pin is low prior to the falling edge of CLKOUT, the memory controller goes into a wait mode until the next positive transition in CLKOUT occurs with READY high. When the external memory is not being used, READY has no effect. Internal control of the number of wait states inserted into a bus cycle held not ready is available through configuration of CCR.
HSI	Inputs to High Speed Input Unit. Four HSI pins are available: HSI.0, HSI.1, HSI.2, and HSI.3. Two of them (HSI.2 and HSI.3) are shared with the HSO Unit. The HSI pins are also used as the SID in Slave Programming Mode on the EPROM device.
HSO	Outputs from High Speed Output Unit. Six HSO pins are available: HSO.0, HSO.1, HSO.2, HSO.3, HSO.4, and HSO.5. Two of them (HSO.4 and HSO.5) are shared with the HSI Unit.
Port 0	8-bit high impedance input-only port. These pins can be used as digital inputs and/or as analog inputs to the on-chip A/D converter. These pins set the Programming Mode on the EPROM device.
Port 1	8-bit quasi-bidirectional I/O port.
Port 2	8-bit multi-functional port. All of its pins are shared with other functions in the 80C196KB.
Ports 3 and 4	8-bit bi-directional I/O ports with open drain outputs. These pins are shared with the multiplexed address/data bus which has strong internal pullups. Available as I/O only on the ROM and EPROM devices.
$\overline{\text{HOLD}}$	Bus Hold input requesting control of the bus. Enabled by setting WSR.7.
$\overline{\text{HLDA}}$	Bus Hold acknowledge output indicating release of the bus. Enabled by setting WSR.7.
$\overline{\text{BREQ}}$	Bus Request output activated when the bus controller has a pending external memory cycle. Enabled by setting WSR.7.
TxD	The TxD pin is used for serial port transmission in Modes 1, 2, and 3. The TxD function is enabled by setting IOC1 5. In mode 0 the pin is used as the serial clock output.
RxD	Serial Port Receive pin used for serial port reception. The RxD function is enabled by setting SPCON.3. In mode 0 the pin functions as input or output data.
EXTINT	A rising edge on the EXTINT pin will generate an external interrupt. EXTINT is selected as the external interrupt source by setting IOC1.1 high.
T2CLK	The T2CLK pin is the Timer2 clock input or the serial port baud rate generator input.
T2RST	A rising edge on the T2RST pin will reset Timer2. The external reset function is enabled by setting IOCO.03 T2RST is enabled as the reset source by clearing IOCO.5.
PWM	Port 2.5 can be enabled as a PWM output by setting IOC1.O The duty cycle of the PWM is determined by the value loaded into the PWM-CONTROL register (17H).
T2UPDN	The T2UPDN pin controls the direction of Timer2 as an up or down counter. The Timer2 up/down function is enabled by setting IOC2.1.
T2CAP	A rising edge on P2.7 will capture the value of Timer2 in the T2CAPTURE register (location OCH in Window 15).

NEW INSTRUCTIONS

The following five instructions have been added to the 8096BH instruction set for the 80C196KB.

PUSHA — PUSHes the PSW, IMASK, IMASK1, and WSR
(Used instead of PUSHF when new interrupts and registers are used.)

assembly language format: PUSHA

object code format: <11110100>

bytes: 1

states: on-chip stack: 12
off-chip stack: 18

POPA — POPs the PSW, IMASK, IMASK1, and WSR
(Used instead of POPF when new interrupts and registers are used.)

assembly language format: POPA

object code format: <11110101>

bytes: 1

states: on-chip stack: 12
off-chip stack:18

IDLPD — Sets the part into Idle or Powerdown Mode

assembly language format: IDLPD #key (key=1 for Idle, key=2 for Powerdown.)

object code format: <11110110> <key>

bytes: 2

states: legal key: 8
illegal key: 25

CMPL — Compare 2 long direct values

```
assembly language format:          DST   SRC
                           CMPL    Lreg, Lreg
```

object code format: <11000101> <src Lreg> <dst Lreg>

bytes: 3

states: 7

BMOV — Block move using 2 auto-incrementing pointers and a counter

```
assembly language format:          PTRS  CNTREG
                           BMOV    Lreg, wreg
```

object code format: <11000001> <wreg> <Lreg>

bytes: 3

```
states:    internal/internal:  8 per transfer + 6
           external/internal: 11 per transfer + 6
           external/external: 14 per transfer + 6
```

SFR OPERATION

All of the registers that were present on the 8096BH work the same way as they did, except that the baud rate value is different. The new registers shown in the memory map control new functions. The most important new register is the Window Select Register (WSR) which allows reading of the formerly write-only registers and vice-versa.

USING THE ALTERNATE REGISTER WINDOW (WSR = 15)

I/O register expansion on the new CHMOS members of the MCS-96 family has been provided by making two register windows available. Switching between these windows is done using the Window Select Register (WSR). The PUSHA and POPA instructions can be used to push and pop the WSR and second interrupt mask when entering or leaving interrupts, so it is easy to change between windows.

On the 80C196KB only Window 0 and Window 15 are active. Window 0 is a true superset of the standard 8096BH SFR space, while Window 15 allows the read-only registers to be written and write-only registers to be read. The only major exception to this is the Timer2 register which is the Timer2 capture register in Window 15. The writeable register for Timer2 is in Window 0. There are also some minor changes and cautions. The descriptions of the registers which have different functions in Window 15 than in Window 0 are listed below:

AD_COMMAND (02H) — Read the last written command

AD_RESULT (02H, 03H) — Write a value into the result register

HSI_MODE (03H) — Read the value in HSI_MODE

HSI_TIME (04H,05H) — Write to FIFO Holding register

HSO_TIME (04H,05H) — Read the last value placed in the holding register

HSI_STATUS (06H) — Write to status bits but not to HSI pin bits. (Pin bits are 1,3,5,7).

HSO_COMMAND (06H) — Read the last value placed in the holding register

SBUF(RX) (07H) — Write a value into the receive buffer

SBUF(TX) (07H) — Read the last value written to the transmit buffer

WATCHDOG(0AH) — Read the value in the upper byte of the WDT

TIMER1 (0AH,0BH) — Write a value to Timer1

TIMER2 (0CH,0DH) — Read/Write the Timer2 capture register.
Note that Timer2 read/write is done with WSR = 0.

IOC2 (0BH) — Last written value is readable, except bit 7 (note 1)

BAUDRATE (0EH) — No function, cannot be read

PORT0 (0EH) — No function, no output drivers on the pins. Register reserved.

PORT1 — IOPORT1 cannot be read or written in Window 15. Register reserved.

SP_STAT (11H) — Set the status bits, TI and RI can be set, but it will not cause an interrupt

SP_CON (11H) — Read the current control byte

IOS0 (15H) — Writing to this register controls the HSO pins. Bits 6 and 7 are inactive for writes.

IOC0 (15H) — Last written value is readable, except bit 1 (note 1)

IOS1 (16H) — Writing to this register will set the status bits, but not cause interrupts. Bits 6 and 7 are not functional

IOC1 (16H) — Last written value is readable

IOS2 (17H) — Writing to this register will set the status bits, but not cause interrupts.

PWM_CONTROL (17H) — Read the duty cycle value written to PWM_CONTROL

NOTE:

1. IOC2.7 (CAM CLEAR) and IOC0.1 (T2RST) are not latched and will read as a 1 (precharged bus) .

Being able to write to the read-only registers and vice-versa provides a lot of flexibility. One of the most useful advantages is the ability to set the timers and HSO lines for initial conditions other than zero.

Reserved registers may be used for testing or for future features. Do not write to these registers. Reads from reserved registers will return indeterminate values.

MEMORY MAP

	0FFFFH
EXTERNAL MEMORY OR I/O	
	4000H
INTERNAL ROM/EPROM OR EXTERNAL MEMORY	
	2080H
RESERVED	
	2040H
UPPER 8 INTERRUPT VECTORS	
	2030H
ROM/EPROM SECURITY KEY	
	2020H
RESERVED	
	2019H
CHIP CONFIGURATION BYTE	
	2018H
RESERVED	
	2014H
LOWER 8 INTERRUPT VECTORS PLUS 2 SPECIAL INTERRUPTS	
	2000H
PORT 3 AND PORT 4	
	1FFEH
EXTERNAL MEMORY OR I/O	
	0100H
INTERNAL DATA MEMORY - REGISTER FILE (STACK POINTER, RAM AND SFRS) EXTERNAL PROGRAM CODE MEMORY	
	0000H

80C196KB INTERRUPTS

Number	Source	Vector Location	Priority
INT15	NMI	203EH	15
INT14	HSI FIFO Full	203CH	14
INT13	EXTINT Pin	203AH	13
INT12	TIMER2 Overflow	2038H	12
INT11	TIMER2 Capture	2036H	11
INT10	4th Entry into HSI FIFO	2034H	10
INT09	RI	2032H	9
INT08	TI	2030H	8
SPECIAL	Unimplemented Opcode	2012H	N/A
SPECIAL	Trap	2010H	N/A
INT07	EXTINT	200EH	7
INT06	Serial Port	200CH	6
INT05	Software Timer	200AH	5
INT04	HSI.0 Pin	2008H	4
INT03	High Speed Outputs	2006H	3
INT02	HSI Data Available	2004H	2
INT01	A/D Conversion Complete	2002H	1
INT00	Timer Overflow	2000H	0

	WHEN READ	WHEN WRITTEN
19H	STACK POINTER	STACK POINTER
18H		
17H	*IOS2	PWM_CONTROL
16H	IOS1	IOC1
15H	IOS0	IOC0
14H	*WSR	*WSR
13H	*INT_MASK 1	*INT_MASK 1
12H	*INT_PEND 1	*INT_PEND 1
11H	*SP_STAT	*SP_CON
10H	PORT2	PORT2
0FH	PORT1	PORT1
0EH	PORT0	BAUD RATE
0DH	TIMER2 (HI)	TIMER2 (HI)
0CH	TIMER2 (LO)	TIMER2 (LO)
0BH	TIMER1 (HI)	*IOC2
0AH	TIMER1 (LO)	WATCHDOG
09H	INT_PENDING	INT_PENDING
08H	INT_MASK	INT_MASK
07H	SBUF(RX)	SBUF(TX)
06H	HSI_STATUS	HSO_COMMAND
05H	HSI_TIME (HI)	HSO_TIME (HI)
04H	HSI_TIME (LO)	HSO_TIME (LO)
03H	AD_RESULT (HI)	HSI_MODE
02H	AD_RESULT (LO)	AD_COMMAND
01H	ZERO REG (HI)	ZERO REG (HI)
00H	ZERO REG (LO)	ZERO REG (LO)

WSR = 0

	RESERVED (1)
0FH	RESERVED (1)
0EH	RESERVED (1)
0DH	*T2 CAPTURE (HI)
0CH	*T2 CAPTURE (LO)

WSR = 15

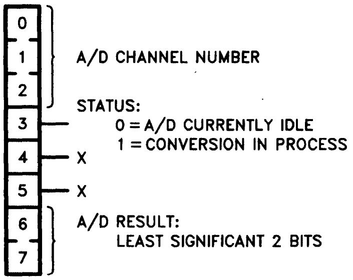
OTHER SFRS IN WSR
15 BECOME READABLE
IF THEY WERE WRITABLE
IN WSR = 0 AND WRITABLE
IF THEY WERE READABLE
IN WSR = 0

*NEW OR CHANGED
REGISTER FUNCTION FROM 8096BH

NOTE:
1. Reserved registers should not be written.

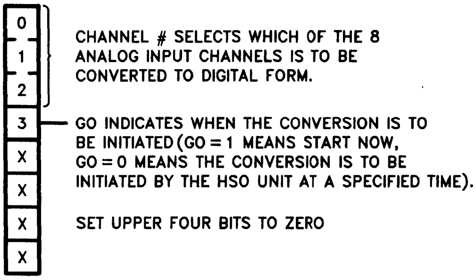
SFR BIT SUMMARY

AD—Result (LO) (02H)



270634–27

AD_Command (02H)



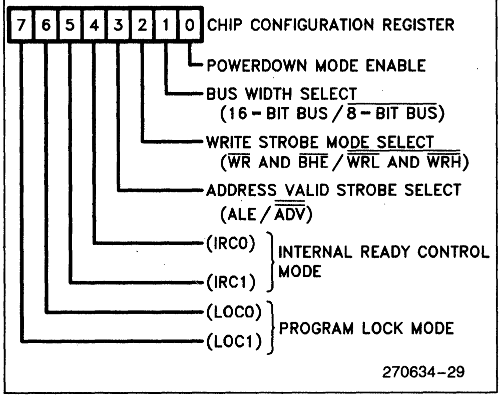
CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.

GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).

SET UPPER FOUR BITS TO ZERO

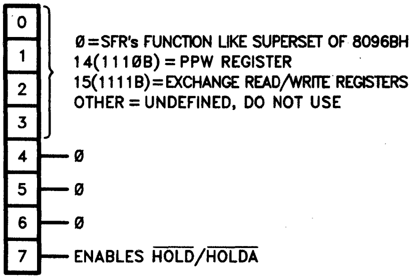
270634–28

Chip Configuration (2018H)


7
6
5
4
3
2
1
0
CHIP CONFIGURATION REGISTER
POWERDOWN MODE ENABLE
BUS WIDTH SELECT
(16-BIT BUS / $\overline{8\text{-BIT BUS}}$)
WRITE STROBE MODE SELECT
($\overline{WR}$ AND $\overline{BHE}$ / $\overline{\text{WRL AND WRH}}$)
ADDRESS VALID STROBE SELECT
(ALE / $\overline{ADV}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE

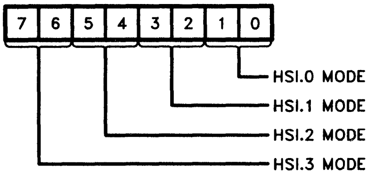
270634–29

WSR (14H)


0
1
2
3
4
5
6
7
Ø = SFR's FUNCTION LIKE SUPERSET OF 8096BH
14(111ØB) = PPW REGISTER
15(1111B) = EXCHANGE READ/WRITE REGISTERS
OTHER = UNDEFINED, DO NOT USE
Ø
Ø
Ø
ENABLES $\overline{HOLD}$/$\overline{HOLDA}$

270634–30

HSI_Mode (03H)

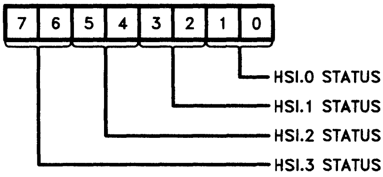


WHERE EACH 2-BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

270634–31

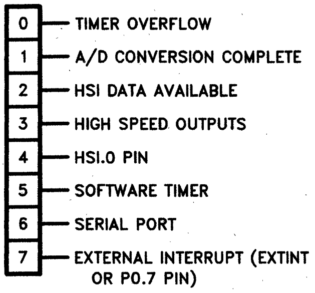
HSI_Status (06H)



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT INDICATES THE CURRENT STATUS OF THE PIN.

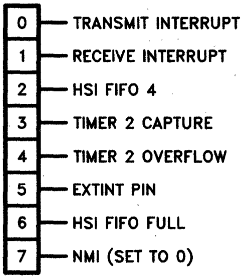
270634–32

INT_PEND/INT_MASK (09H/08H)



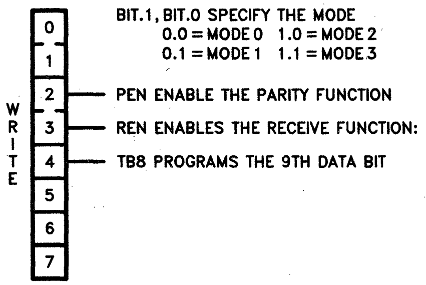
270634-33

INT_PEND1/INT_MASK1 (12H/13H)



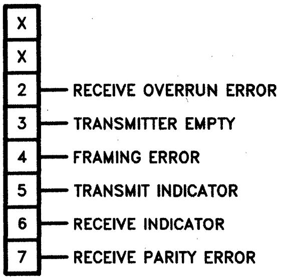
270634-34

SP_CON (11H)


BIT.1, BIT.0 SPECIFY THE MODE
0.0 = MODE 0 1.0 = MODE 2
0.1 = MODE 1 1.1 = MODE 3
0
1
2 PEN ENABLE THE PARITY FUNCTION
3 REN ENABLES THE RECEIVE FUNCTION:
4 TB8 PROGRAMS THE 9TH DATA BIT
5
6
7
WRITE

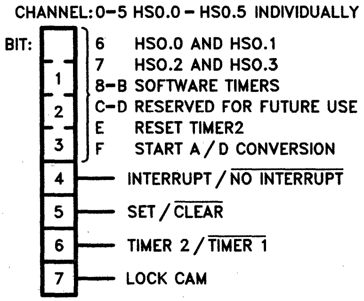
270634-35

SP_STAT (11H)

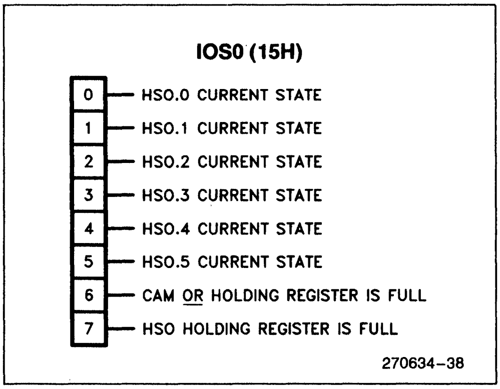


270634-36

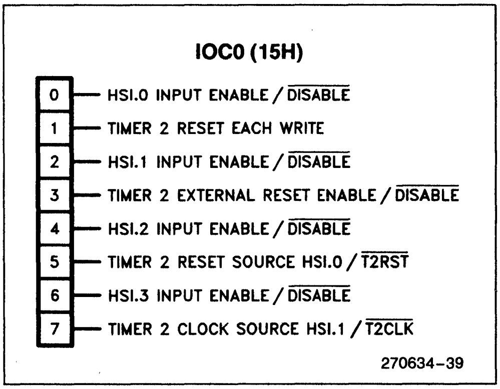
HSO Command (06H)


CHANNEL: 0–5 HSO.0 – HSO.5 INDIVIDUALLY
BIT:
6 HSO.0 AND HSO.1
7 HSO.2 AND HSO.3
8–B SOFTWARE TIMERS
C–D RESERVED FOR FUTURE USE
E RESET TIMER2
F START A/D CONVERSION
1
2
3
4 INTERRUPT / $\overline{\text{NO INTERRUPT}}$
5 SET / $\overline{\text{CLEAR}}$
6 TIMER 2 / $\overline{\text{TIMER 1}}$
7 LOCK CAM

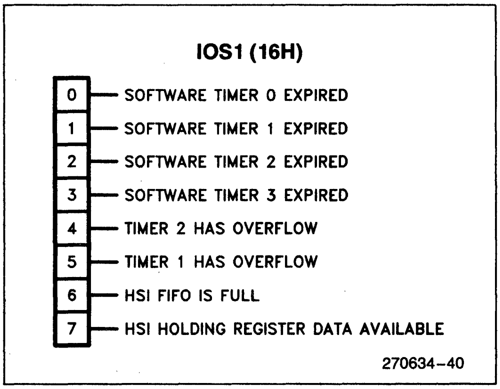
270634-37



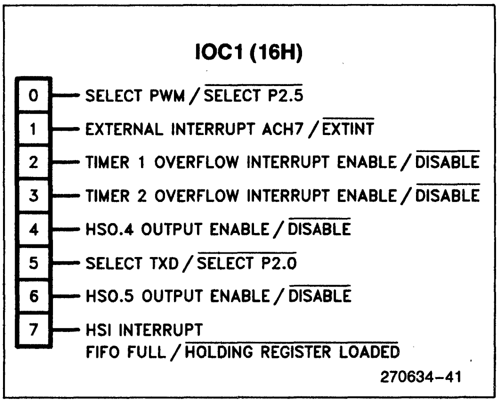
270634-38


IOC0 (15H)
0 HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1 TIMER 2 RESET EACH WRITE
2 HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3 TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4 HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5 TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6 HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7 TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$

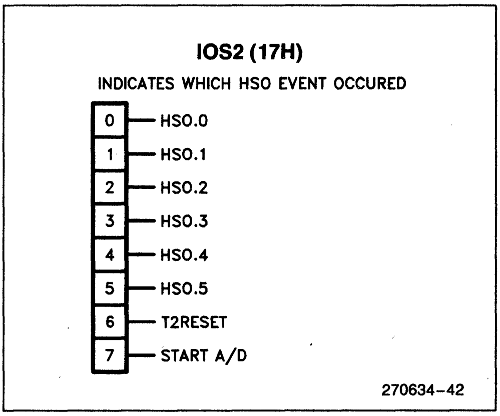
270634-39


IOS1 (16H)
0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE

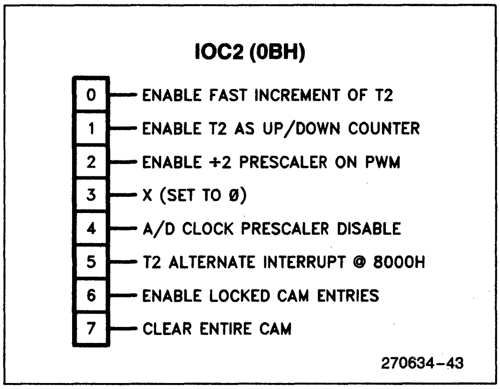
270634-40


IOC1 (16H)
0 SELECT PWM / $\overline{\text{SELECT P2.5}}$
1 EXTERNAL INTERRUPT ACH7 / $\overline{\text{EXTINT}}$
2 TIMER 1 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
3 TIMER 2 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
4 HSO.4 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
5 SELECT TXD / $\overline{\text{SELECT P2.0}}$
6 HSO.5 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
7 HSI INTERRUPT FIFO FULL / $\overline{\text{HOLDING REGISTER LOADED}}$

270634-41



270634-42


IOC2 (0BH)
0 ENABLE FAST INCREMENT OF T2
1 ENABLE T2 AS UP/DOWN COUNTER
2 ENABLE ÷2 PRESCALER ON PWM
3 X (SET TO Ø)
4 A/D CLOCK PRESCALER DISABLE
5 T2 ALTERNATE INTERRUPT @ 8000H
6 ENABLE LOCKED CAM ENTRIES
7 CLEAR ENTIRE CAM

270634-43

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings*

Ambient Temperature
Under Bias . . . . 0°C to +70°C

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

Voltage On Any Pin to V_{SS} . . . . −0.5V to +7.0V

Power Dissipation 1.5W

**Notice: Stresses 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 specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

NOTICE: Specifications contained within the following tables are subject to change.

Operating Conditions

Symbol	Description	Min	Max	Units
T_A	Ambient Temperature Under Bias	0	+70	°C
V_{CC}	Digital Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	3.5	12	MHz

NOTE:
ANGND and V_{SS} should be nominally at the same potential.

This is an Advance Data Sheet. It is expected that parameters may change before Intel releases this product for sale. Contact your local sales office before finalizing the Timing and D.C. Characteristics section of a design to verify you have the latest information.

D.C. Characteristics (Over specified operating conditions)

Symbol	Description	Min	Max	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage (Note 1)	0.2 V_{CC} + 0.9	V_{CC} + 0.5	V	
V_{IH1}	Input High Voltage on XTAL 1	0.7 V_{CC}	V_{CC} + 0.5	V	
V_{IH2}	Input High Voltage on RESET	2.2	V_{CC} + 0.5	V	
V_{OL}	Output Low Voltage		0.3 0.45 1.5	V V V	I_{OL} = 200 μA I_{OL} = 3.2 mA I_{OL} = 7 mA
V_{OH}	Output High Voltage (Standard Outputs)	V_{CC} − 0.3 V_{CC} − 0.7 V_{CC} − 1.5		V V V	I_{OH} = −200 μA I_{OH} = −3.2 mA I_{OH} = −7 mA
V_{OH1}	Output High Voltage (Quasi-bidirectional Outputs)	V_{CC} − 0.3 V_{CC} − 0.7 V_{CC} − 1.5		V V V	I_{OH} = −10 μA I_{OH} = −30 μA I_{OH} = −60 μA
I_{LI}	Input Leakage Current (Std. Inputs)		±10	μA	0 < V_{IN} < V_{CC} − 0.3V
I_{LI1}	Input Leakage Current (Port 0)		+3	μA	0 < V_{IN} < V_{REF}
I_{TL}	1 to 0 Transition Current (QBD Pins)		−650	μA	V_{IN} = 2.0V
I_{IL}	Logical 0 Input Current (QBD Pins)		−50	μA	V_{IN} = 0.45V
I_{IL1}	Logical 0 Input Current in Reset (Note 2) (ALE, $\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, INST, P2.0)		−950	μA	V_{IN} = 0.45 V

NOTE:
1. All pins except RESET and XTAL1.
2. Holding these pins below V_{IH} in Reset may cause the part to enter test modes.

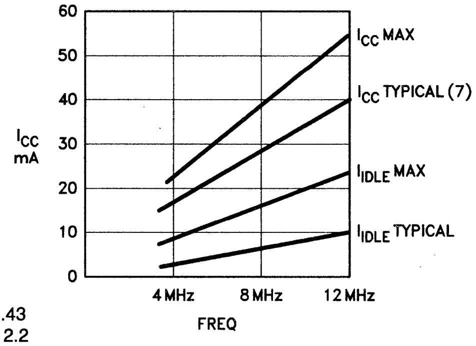
D.C. Characteristics (Over specified operating conditions) (Continued)

Symbol	Description	Min	Typ(7)	Max	Units	Test Conditions
I_{CC}	Active Mode Current in Reset		40	55	mA	XTAL1 = 12 MHz $V_{CC} = V_{PP} = V_{REF} = 5.5V$
I_{REF}	A/D Converter Reference Current		2	5	mA	
I_{IDLE}	Idle Mode Current		10	22	mA	
I_{CC1}	Active Mode Current		15	22	mA	XTAL1 = 3.5 MHz
I_{PD}(8)	Powerdown Mode Current		5		μA	$V_{CC} = V_{PP} = V_{REF} = 5.5V$
R_{RST}	Reset Pullup Resistor	6K		50K	Ω	
C_S	Pin Capacitance (Any Pin to V_{SS})			10	pF	$f_{TEST} = 1.0$ MHz

NOTES:
(Notes apply to all specifications)
1. QBD (Quasi-bidirectional) pins include Port 1, P2.6 and P2.7.
2. Standard Outputs include AD0–15, $\overline{RD}$, $\overline{WR}$, ALE, $\overline{BHE}$, INST, HSO pins, PWM/P2.5, CLKOUT, RESET, Ports 3 and 4, TXD/P2.0, and RXD (in serial mode 0). The V_{OH} specification is not valid for RESET. Ports 3 and 4 are open-drain outputs.
3. Standard Inputs include HSI pins, CDE, $\overline{EA}$, READY, BUSWIDTH, NMI, RXD/P2.1, EXTINT/P2.2, T2CLK/P2.3, and T2RST/P2.4.
4. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V or V_{OH} is held below V_{CC} – 0.7V:
   I_{OL} on Output pins: 10 mA
   I_{OH} on quasi-bidirectional pins: self limiting
   I_{OH} on Standard Output pins: 10 mA
5. Maximum current per bus pin (data and control) during normal operation is ±3.2 mA.
6. During normal (non-transient) conditions the following total current limits apply:

Port 1, P2.6	I_{OL}: 29 mA	I_{OH} is self limiting
HSO, P2.0, RXD, $\overline{RESET}$	I_{OL}: 29 mA	I_{OH}: 26 mA
P2.5, P2.7, $\overline{WR}$, $\overline{BHE}$	I_{OL}: 13 mA	I_{OH}: 11 mA
AD0–AD15	I_{OL}: 52 mA	I_{OH}: 52 mA
$\overline{RD}$, ALE, INST–CLKOUT	I_{OL}: 13 mA	I_{OH}: 13 mA

7. Typicals are based on a limited number of samples and are not guaranteed. The values listed are at room temperature and $V_{REF} = V_{CC} = 5V$.
8. I_{PD} is not guaranteed on the standard 80C196KB part and may exceed 100 μA on some parts. Customers whose applications use the powerdown mode and require a guaranteed maximum value of I_{PD} should contact an Intel Field Sales Representative.



I_{CC} Max = 3.88 × FREQ + 8.43
I_{IDLE} Max = 1.65 × FREQ + 2.2

270634–15

Figure 5. I_{CC} and I_{IDLE} vs Frequency

A.C. Characteristics

For use over specified operating conditions

Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, f_{OSC} = 12 MHz

The system must meet these specifications to work with the 83C196KB/80C196KB: (Note 1)

Symbol	Description	Min	Max	Units	Notes
T_{AVYV}	Address Valid to Ready Setup 83C196KB10 83C196KB12/80C196KB		 $2T_{OSC} - 90$ $2T_{OSC} - 85$	 ns ns	
T_{LLYV}	ALE Low to READY Setup 80C196KB 83C196KB10 83C196KB12		 $T_{OSC} - 65$ $T_{OSC} - 80$ $T_{OSC} - 72$	 ns ns ns	
T_{YLYH}	Non READY Time	No upper limit		ns	
T_{CLYX}	READY Hold after CLKOUT Low	0	$T_{OSC} - 30$	ns	(Note 2)
T_{LLYX}	READY Hold after ALE Low	$T_{OSC} - 15$	$2T_{OSC} - 40$	ns	(Note 2)
T_{AVGV}	Address Valid to Buswidth Setup		$2T_{OSC} - 85$	ns	
T_{LLGV}	ALE Low to Buswidth Setup 80C196KB 83C196KB		 $T_{OSC} - 60$ $T_{OSC} - 70$	 ns ns	
T_{CLGX}	Buswidth Hold after CLKOUT Low	0		ns	
T_{AVDV}	Address Valid to Input Data Valid 80C196KB 83C196KB10 83C196KB12		 $3T_{OSC} - 60$ $3T_{OSC} - 70$ $3T_{OSC} - 67$	 ns ns ns	(Note 3)
T_{RLDV}	$\overline{RD}$ Active to Input Data Valid 83C196KB10 83C196KB12/80C196KB		 $T_{OSC} - 30$ $T_{OSC} - 23$	 ns ns	(Note 3)
T_{CLDV}	CLKOUT Low to Input Data Valid		$T_{OSC} - 50$	ns	
T_{RHDZ}	End of $\overline{RD}$ to Input Data Float		$T_{OSC} - 20$	ns	
T_{RXDX}	Data Hold after $\overline{RD}$ Inactive	0		ns	

NOTES:

1. Customers whose applications require an 83C196KB to meet the 80C196KB specifications listed above should contact an Intel Field Sales Representative.
2. If max is exceeded, additional wait states will occur.
3. When using wait states, add $2T_{OSC} \times n$, where n = number of wait states.

A.C. Characteristics

For use over specified operating conditions

Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, f_{OSC} = 12 MHz

The 83C196KB/80C196KB will meet these specifications: (Note 1)

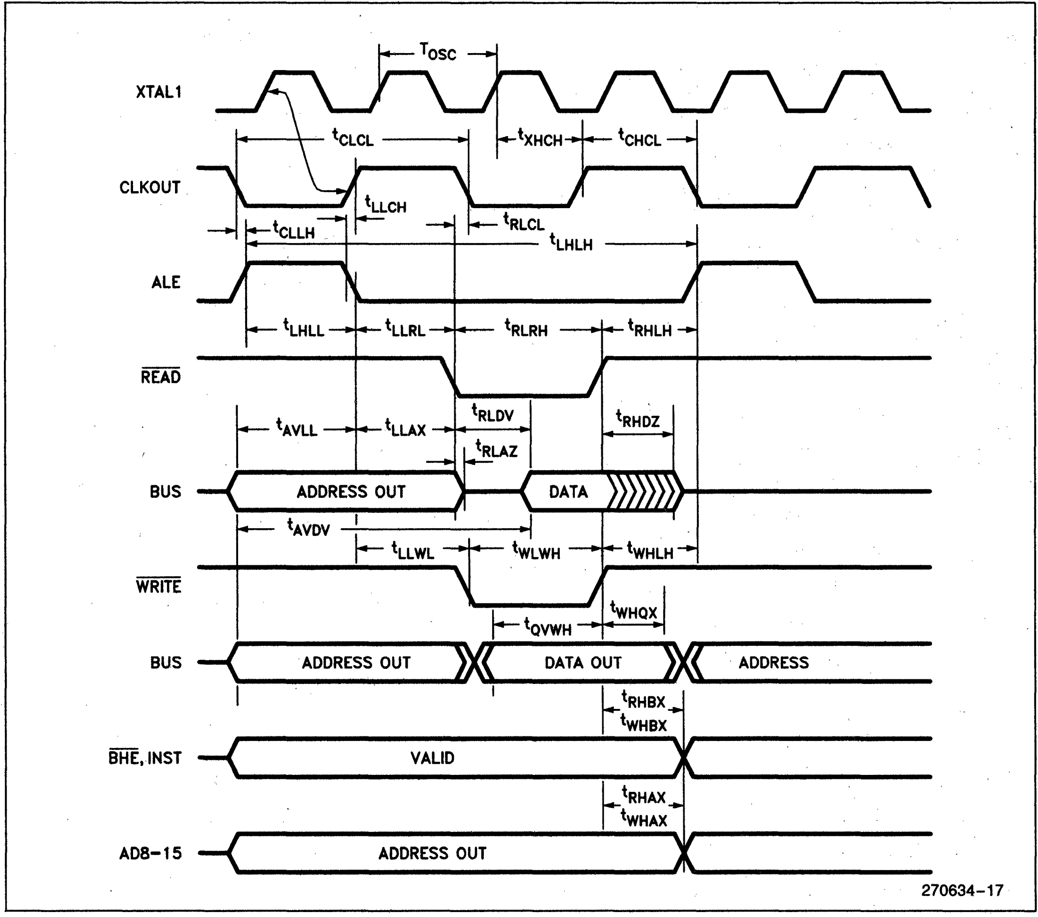
Symbol	Description	Min	Max	Units	Notes
F_{XTAL}	Frequency on $XTAL_1$ 83C196KB10 83C196KB12/80C196KB	 3.5 3.5	 10 12	 MHz MHz	 (Note 2) (Note 2)
T_{OSC}	I/F_{XTAL} 83C196KB10 83C196KB12/80C196KB	 100 83	 286 286	 ns ns	
T_{XHCH}	XTAL1 High to CLKOUT High or Low	40	110	ns	(Note 3)
T_{CLCL}	CLKOUT Cycle Time	$2T_{OSC}$		ns	
T_{CHCL}	CLKOUT High Period	$T_{OSC} - 10$	$T_{OSC} + 10$	ns	
T_{CLLH}	CLKOUT Falling Edge to ALE Rising	−5	15	ns	
T_{LLCH}	ALE Falling Edge to CLKOUT Rising	−15	15	ns	
T_{LHLH}	ALE Cycle Time	$4T_{OSC}$		ns	(Note 5)
T_{LHLL}	ALE High Period	$T_{OSC} - 10$	$T_{OSC} + 10$	ns	
T_{AVLL}	Address Setup to ALE Falling Edge	$T_{OSC} - 20$			
T_{LLAX}	Address Hold after ALE Falling Edge	$T_{OSC} - 40$		ns	
T_{LLRL}	ALE Falling Edge to $\overline{RD}$ Falling Edge 80C196KB 83C196KB	 $T_{OSC} - 30$ $T_{OSC} - 40$		 ns ns	
T_{RLCL}	$\overline{RD}$ Low to CLKOUT Falling Edge	5	30	ns	
T_{RLRH}	$\overline{RD}$ Low Period	$T_{OSC} - 5$	$T_{OSC} + 25$	ns	(Note 5)
T_{RHLH}	$\overline{RD}$ Rising Edge to ALE Rising Edge	T_{OSC}	$T_{OSC} + 25$	ns	(Note 4)
T_{RLAZ}	$\overline{RD}$ Low to Address Float		10	ns	
T_{LLWL}	ALE Falling Edge to $\overline{WR}$ Falling Edge	$T_{OSC} - 10$		ns	
T_{CLWL}	CLKOUT Low to $\overline{WR}$ Falling Edge	0	25	ns	
T_{QVWH}	Data Stable to $\overline{WR}$ Rising Edge 83C196KB10 83C196KB12/80C196KB	 $T_{OSC} - 30$ $T_{OSC} - 23$		 ns ns	 (Note 5)
T_{CHWH}	CLKOUT High to $\overline{WR}$ Rising Edge	−10	10	ns	
T_{WLWH}	$\overline{WR}$ Low Period	$T_{OSC} - 30$	$T_{OSC} + 5$	ns	(Note 5)
T_{WHQX}	Data Hold after $\overline{WR}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{WHLH}	$\overline{WR}$ Rising Edge to ALE Rising Edge	$T_{OSC} - 10$	$T_{OSC} + 15$	ns	(Note 4)
T_{WHBX}	$\overline{BHE}$, INST Hold after $\overline{WR}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{RHBX}	$\overline{BHE}$, INST Hold after $\overline{RD}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{WHAX}	AD8–15 Hold after $\overline{WR}$ Rising Edge	$T_{OSC} - 50$		ns	
T_{RHAX}	AD8–15 Hold after $\overline{RD}$ Rising Edge	$T_{OSC} - 25$		ns	

NOTES:

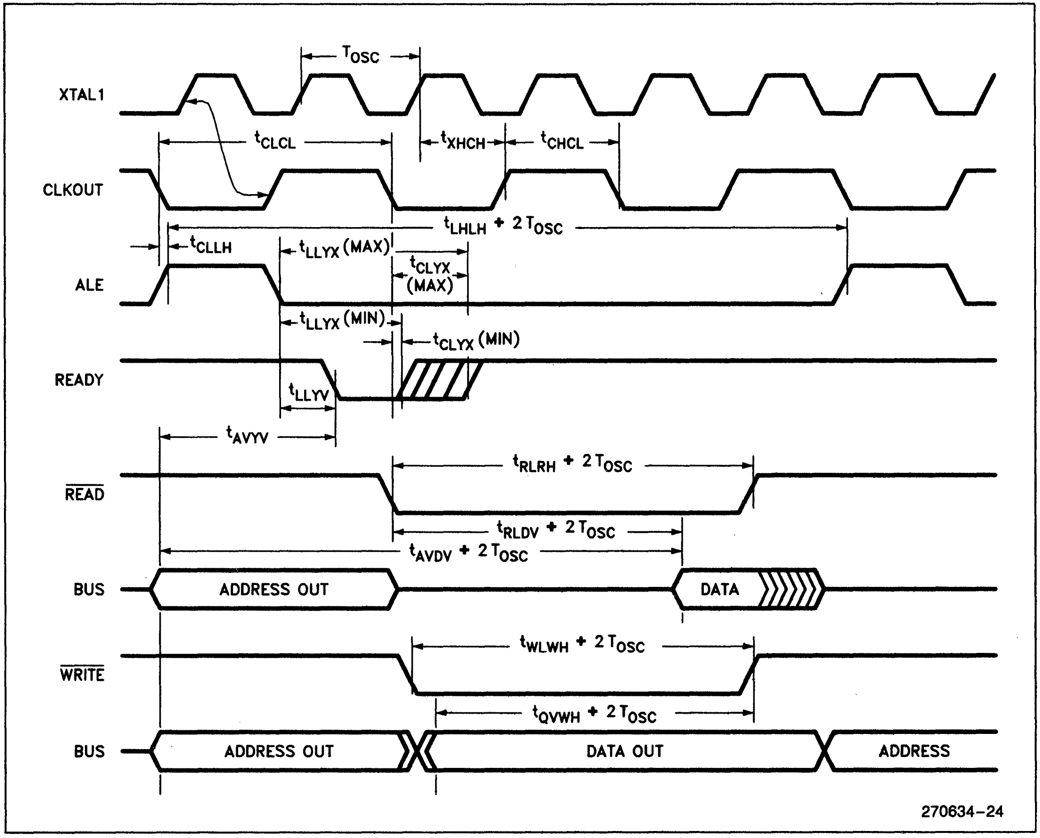
T_{OSC} = 83.3 ns at 12 MHz; T_{OSC} = 100 ns at 10 MHz.

1. Customers whose applications require an 83C196KB to meet the 80C196KB specifications listed above should contact an Intel Field Sales Representative.
2. Testing performed at 3.5 MHz. However, the part is static by design and will typically operate below 1 Hz.
3. Typical specification, not guaranteed.
4. Assuming back-to-back bus cycles.
5. When using wait states, add $2T_{OSC} \times n$, where n = number of wait states.

System Bus Timings

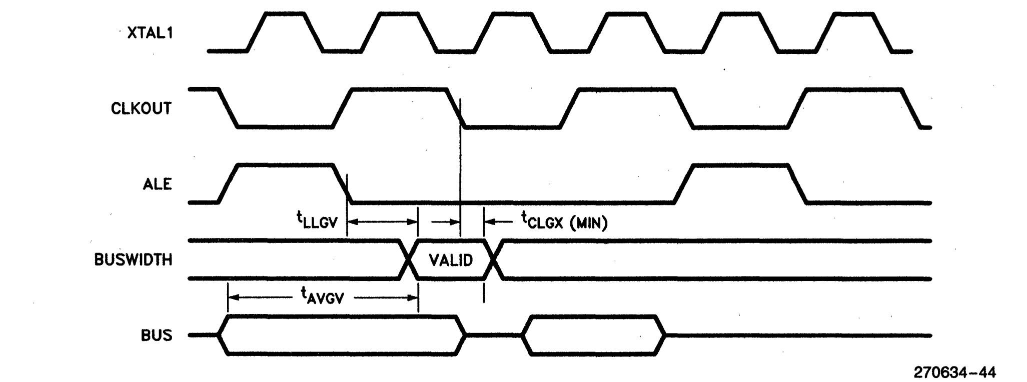


READY Timings (One Waitstate)



270634-24

Buswidth Timings

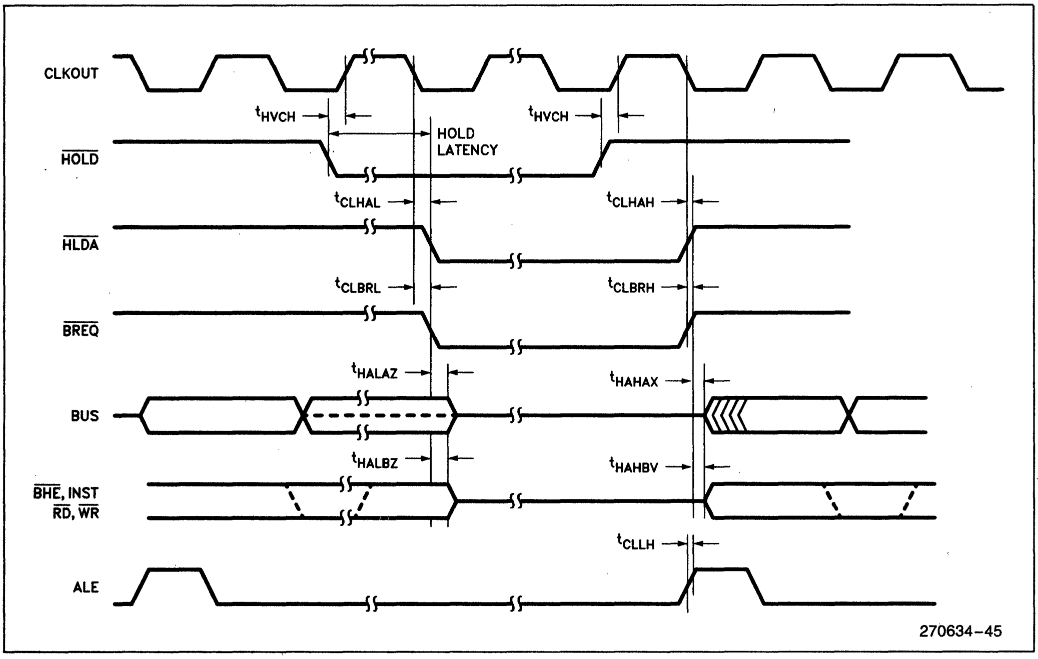


270634-44

$\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ TIMINGS

Symbol	Description	Min	Max	Units	Notes
T_{HVCH}	$\overline{\text{HOLD}}$ Setup 80C196KB 83C196KB	 75 85		ns	1
T_{CLHAL}	CLKOUT Low to $\overline{\text{HLDA}}$ Low	−15	15	ns	
T_{CLBRL}	CLKOUT Low to $\overline{\text{BREQ}}$ Low	−15	15	ns	
T_{HALAZ}	$\overline{\text{HLDA}}$ Low to Address Float 80C196KB 83C196KB		 15 20	ns	
T_{HALBZ}	$\overline{\text{HLDA}}$ Low to BHE, INST, RD, WR Float			ns	
T_{CLHAH}	CLKOUT Low to $\overline{\text{HLDA}}$ High	−15	15	ns	
T_{CLBRH}	CLKOUT Low to $\overline{\text{BREQ}}$ High	−15	15	ns	
T_{HAHAX}	$\overline{\text{HLDA}}$ High to Address No Longer Float	−5		ns	
T_{HAHBV}	$\overline{\text{HLDA}}$ High to $\overline{\text{BHE}}$, INST, $\overline{\text{RD}}$, $\overline{\text{WR}}$ Valid	−20		ns	
T_{CLLH}	CLKOUT Low to ALE High	−5	15	ns	

NOTE:
1. To guarantee recognition at next clock.

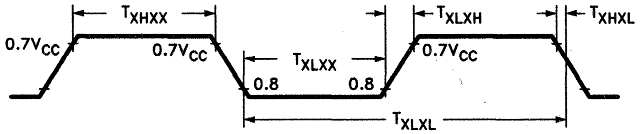


270634–45

EXTERNAL CLOCK DRIVE

Symbol	Parameter	Min	Max	Units
$1/T_{XLXL}$	Oscillator Frequency 80C196KB10 80C196KB12	 3.5 3.5	 10.0 12.0	 MHz MHz
T_{XLXL}	Oscillator Frequency 80C196KB10 80C196KB12	 100 83	 286 286	 ns ns
T_{XHXX}	High Time	32		ns
T_{XLXX}	Low Time	32		ns
T_{XLXH}	Rise Time		10	ns
T_{XHXL}	Fall Time		10	ns

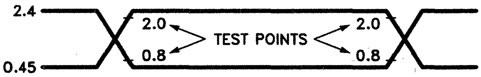
EXTERNAL CLOCK DRIVE WAVEFORMS



270634–18

An external oscillator may encounter as much as a 100 pF load at XTAL1 when it starts-up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{IL} and V_{IH} specifications the capacitance will not exceed 20 pF.

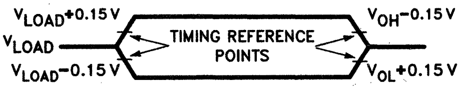
A.C. TESTING INPUT, OUTPUT WAVEFORM



270634–19

A.C. Testing inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0" Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

FLOAT WAVEFORM



270634–20

For Timing Purposes a Port Pin is no Longer Floating when a 100 mV change from Load Voltage Occurs and Begins to Float when a 100 mV change from the Loaded V_{OH}/V_{OL} Level occurs $I_{OL}/I_{OH} = \pm 15$ mA.

EXPLANATION OF AC SYMBOLS

Each symbol is two pairs of letters prefixed by "T" for time. The characters in a pair indicate a signal and its condition, respectively. Symbols represent the time between the two signal/condition points.

Conditions:		Signals:			
H	- High	A	- Address	HA	- $\overline{\text{HLDA}}$
L	- Low	B	- $\overline{\text{BHE}}$	L	- ALE/$\overline{\text{ADV}}$
V	- Valid	BR	- $\overline{\text{BREQ}}$	Q	- DATA OUT
X	- No Longer Valid	C	- CLKOUT	R	- $\overline{\text{RD}}$
Z	- Floating	D	- DATA IN	W	- $\overline{\text{WR}}/\overline{\text{WRH}}/\overline{\text{WRL}}$
		G	- Buswidth	X	- XTAL1
		H	- $\overline{\text{HOLD}}$	Y	- READY

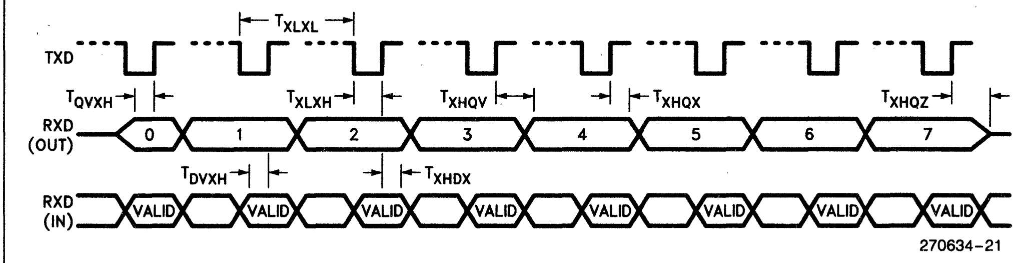
A.C. CHARACTERISTICS—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT TIMING—SHIFT REGISTER MODE

Symbol	Parameter	Min	Max	Units
T_{XLXL}	Serial Port Clock Period (BRR ≥ 8002H)	6 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR ≥ 8002H)	4 T_{OSC} ±50		ns
T_{XLXL}	Serial Port Clock Period (BRR = 8001H)	4 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR = 8001H)	2 T_{OSC} ±50		ns
T_{QVXH}	Output Data Setup to Clock Rising Edge	2 T_{OSC} −50		ns
T_{XHQX}	Output Data Hold after Clock Rising Edge	2 T_{OSC} −50		ns
T_{XHQV}	Next Output Data Valid after Clock Rising Edge		2 T_{OSC} +50	ns
T_{DVXH}	Input Data Setup to Clock Rising Edge	T_{OSC} +50		ns
T_{XHDX}	Input Data Hold after Clock Rising Edge	0		ns
T_{XHQZ}	Last Clock Rising to Output Float		1 T_{OSC}	ns

WAVEFORM—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT WAVEFORM—SHIFT REGISTER MODE



A TO D CHARACTERISTICS

There are two modes of A/D operation: with or without clock prescaler. The speed of the A/D converter can be adjusted by setting a clock prescaler on or off. At high frequencies more time is needed for the comparator to settle. The maximum frequency with the clock prescaler disabled is 8 MHz. The conversion times with the prescaler turned on or off is shown in the table below.

The converter is ratiometric, so the absolute accuracy is directly dependent on the accuracy and stability of V_{REF}. V_{REF} must be close to V_{CC} since it supplies both the resistor ladder and the digital section of the converter.

A/D CONVERTER SPECIFICATIONS

The specifications given below assume adherence to the Operating Conditions section of this data sheet. Testing is performed with V_{REF} = 5.12V.

Clock Prescaler On **IOC2.4 = 0**	**Clock Prescaler Off** **IOC2.4 = 1**
158 States 26.33 µs @ 12 MHz	91 States 22.75 µs @ 8 MHz

Parameter	Typical(1)	Minimum	Maximum	Units*	Notes
Resolution		512 9	1024 10	Levels Bits	
Absolute Error		0	±4	LSBs	
Full Scale Error	0.25 ±0.50			LSBs	
Zero Offset Error	−0.25 ±0.50			LSBs	
Non-Linearity Error	1.5 ±2.5	0	±4	LSBs	
Differential Non-Linearity Error		> −1	+2	LSBs	
Channel-to-Channel Matching	±0.1	0	±1	LSBs	
Repeatability	±0.25			LSBs	
Temperature Coefficients: Offset Full Scale Differential Non-Linearity	 0.009 0.009 0.009			 LSB/°C LSB/°C LSB/°C	
Off Isolation		−60		dB	2, 3
Feedthrough	−60			dB	2
V_{CC} Power Supply Rejection	−60			dB	2
Input Resistance		1K	5K	Ω	
D.C. Input Leakage		0	3.0	µA	
Sample Time: Prescaler On Prescaler Off	15 8			States States	4 4
Input Capacitance	3			pF	

NOTES:
*An "LSB", as used here, has a value of approximately 5 mV.
1. Typical values are expected for most devices at 25°C but are not tested or guaranteed.
2. DC to 100 KHz.
3. Multiplexer Break-Before-Make Guaranteed.
4. One state = 167 ns at 12 MHz, 250 ns at 8 MHz.

A/D GLOSSARY OF TERMS

ABSOLUTE ERROR—The maximum difference between corresponding actual and ideal code transitions. Absolute Error accounts for all deviations of an actual converter from an ideal converter.

ACTUAL CHARACTERISTIC—The characteristic of an actual converter. The characteristic of a given converter may vary over temperature, supply voltage, and frequency conditions. An actual characteristic rarely has ideal first and last transition locations or ideal code widths. It may even vary over multiple conversions under the same conditions.

BREAK-BEFORE-MAKE—The property of multiplexer which guarantees that a previously selected channel will be deselected before a new channel is selected (e.g., the converter will not short inputs together).

CHANNEL-TO-CHANNEL MATCHING—The difference between corresponding code transitions of actual characteristics taken from different channels under the same temperature, voltage and frequency conditions.

CHARACTERISTIC—A graph of input voltage versus the resultant output code for an A/D converter. It describes the transfer function of the A/D converter.

CODE—The digital value output by the converter.

CODE TRANSITION—The point at which the converter changes from an output code of Q, to a code of Q + 1. The input voltage corresponding to a code transition is defined to be that voltage which is equally likely to produce either of two adjacent codes.

CODE WIDTH—The voltage corresponding to the difference between two adjacent code transitions.

D.C. INPUT LEAKAGE—Leakage current to ground from an analog input pin.

DIFFERENTIAL NON-LINEARITY—The difference between the ideal and actual code widths of the terminal based characteristic.

FEEDTHROUGH—Attenuation of a voltage applied on the selected channel of the A/D Converter after the sample window closes.

FULL SCALE ERROR—The difference between the expected and actual input voltage corresponding to the full scale code transition.

IDEAL CHARACTERISTIC—A characteristic with its first code transition at V_{IN} = 0.5 LSB, its last code transition at V_{IN} = (V_{REF} − 1.5 LSB) and all code widths equal to one LSB.

INPUT RESISTANCE—The effective series resistance from the analog input pin to the sample capacitor.

LSB—Least Significant Bit: The voltage corresponding to the full scale voltage divided by 2^n, where n is the number of bits of resolution of the converter. For an 8-bit converter with a reference voltage of 5.12V, one LSB is 20 mV. Note that this is different than digital LSBs, since an uncertainty of two LSB, when referring to an A/D converter, equals 40 mV. (This has been confused with an uncertainty of two digital bits, which would mean four counts, or 80 mV.)

NON-LINEARITY—The maximum deviation of code transitions of the terminal based characteristic from the corresponding code transitions of the ideal characteristic.

OFF-ISOLATION—Attenuation of a voltage applied on a deselected channel of the A/D converter. (Also referred to as Crosstalk.)

REPEATABILITY—The difference between corresponding code transitions from different actual characteristics taken from the same converter on the same channel at the same temperature, voltage and frequency conditions.

RESOLUTION—The number of input voltage levels that the converter can unambiguously distinguish between. Also defines the number of useful bits of information which the converter can return.

SAMPLE TIME—Begins when the sample capacitor is attached to a selected channel and ends when the sample capacitor is disconnected from the selected channel.

TEMPERATURE COEFFICIENTS—Change in the stated variable per degree centigrade temperature change. Temperature coefficients are added to the typical values of a specification to see the effect of temperature drift.

TERMINAL BASED CHARACTERISTIC—An actual characteristic which has been rotated and translated to remove zero offset and full scale error.

V_{CC} REJECTION—Attenuation of noise on the V_{CC} line to the A/D converter.

ZERO OFFSET—The difference between the expected and actual input voltage corresponding to the first code transition.

80C196KB FUNCTIONAL DEVIATIONS

The 80C196KB has the following problems.

1. The DJNZW instruction is not guaranteed to be functional. The instruction, if encountered, will not cause an unimplemented opcode interrupt. (The opcode for DJNZW is 0E1 Hex.) The DJNZ (byte) instruction works correctly and should be used instead.
2. The CDE function is not guaranteed to work. The CDE pin must be directly connected to V_{SS}.
3. The HSI unit has two errata: one dealing with resolution and the other with first entries into the FIFO.

   The HSI resolution is 9 states instead of 8 states. Events on the same line may be lost if they occur faster than once every 9 state times.

   There is a mismatch between the 9 state time HSI resolution and the 8 state time timer. This causes one time value to be unused every 9 timer counts. Events may receive a time-tag one count later than expected because of this "skipped" time value.

   If the first two events into an empty FIFO (not including the Holding Register) occur in the same internal phase, both are recorded with one time-tag. Otherwise, if the second event occurs within 9 states after the first, its time-tag is one count later than the first's. If this is the "skipped" time value, the second event's time-tag is 2 counts later than the first's.

   If the FIFO and Holding Register are empty, the first event will transfer into the Holding Register after 8 state times, leaving the FIFO empty again. If the second event occurs after this time, it will act as a new first event into an empty FIFO.
4. The serial port Framing Error flag fails to indicate an error if the bit preceding the stop bit is a 1. This is the case in both the 8-bit and 9-bit modes. False framing errors are never generated.

DIFFERENCES BETWEEN THE 80C196KA AND THE 80C196KB

The 8XC196KB is identical to 8XC196KA except for the following differences.

1. ALE is high after reset on the 80C196KB instead of low as on the 80C196KA.
2. The DJNZW instruction is not guaranteed to work on the 80C196KB.
3. The $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ bus protocol is available on the 80C196KB.

CONVERTING FROM OTHER 8096BH FAMILY PRODUCTS TO THE 80C196KB

The following list of suggestions for designing an 809XBH system will yield a design that is easily converted to the 80C196KB.

1. Do not base critical timing loops on instruction or peripheral execution times.
2. Use equate statements to set all timing parameters, including the baud rate.
3. Do not base hardware timings on CLKOUT or XTAL1. The timings of the 80C196KB are different than those of the 8X9XBH, but they will function with standard ROM/EPROM/Peripheral type memory systems.
4. Make sure all inputs are tied high or low and not left floating.
5. Indexed and indirect operations relative to the stack pointer (SP) work differently on the 80C196KB than on the 8096BH. On the 8096BH, the address is calculated based on the un-undated version of the stack pointer. The 80C196KB uses the updated version. The offset for POP[SP] and POP nn[SP] instructions may need to be changed by a count of 2.
6. The V_{PD} pin on the 8096BH has changed to a V_{SS} pin on the 80C196KB.

DATA SHEET REVISION HISTORY

The following differences exist between the -003 and the -002 version of the 83C196KB/80C196KB data sheets.

1. The CDE (Clock Detect Enable) pin was changed to a V_{SS} pin.
2. The 83C196TB/80C196TB, a C196 without an A/D Converter, was added.
3. The QFP package was added.
4. Thermal Characteristics for the PGA, PLCC and QFP packages were added.
5. The Instruction Summary Table, Execution State Times Table, and Baud Rate Calculations were deleted. This information is available in the 80C196KB User's Guide.
6. TAVYV for the 83C196KB10 was changed from 2Tosc − 85 to 2Tosc − 90.
7. The figures for System Bus Timings were redrawn to include missing timings and to more accurately describe a wait state condition.
8. $\overline{HOLD}/\overline{HLDA}$ timings were added.
9. On the A/D Converter, the maximum time for the Prescaler Off Mode was changed from 6 MHz to 8 MHz.
10. Errata was added for the HSI resolution and first event anomalies.
11. Errata was added for the serial port Framing Error anomaly.
12. The I_{IL1} max. specification was changed from –950 μA to –1.2 mA.

The following differences exist between the 83C196KB/80C196KB (270634-002) and the 80C196KB (270634-001) data sheets.

1. The 83C196KB ROM version was added.
2. The PGA pin out for 80C196KB was added.
3. $\overline{HOLD}/\overline{HLDA}$ text was deleted.
4. TRHBX, TWHAX, and TRHAX were added as specifications.
5. The I_{IL1} specification changed from −850 mA to −950 mA.
6. The following A.C. Characteristics have changed for the 80C196KB:

Symbol	Description	80C196KB (Rev -001)	80C196KB (Rev -002)
TLLYV	ALE Low to READY Setup	$T_{OSC} - 72$	$T_{OSC} - 65$
TLLGV	ALE Low to BUSWIDTH Setup	$T_{OSC} - 70$	$T_{OSC} - 60$
TAVDV	Address Valid to Input Data Valid	$3\,T_{OSC} - 67$	$3\,T_{OSC} - 60$
TAVLL	Address Setup to ALE Falling Edge	$T_{OSC} - 15$	$T_{OSC} - 20$
TLLRL	ALE Falling Edge to $\overline{RD}$ Falling Edge	$T_{OSC} - 40$	$T_{OSC} - 30$
TRLCL	$\overline{RD}$ Low to CLKOUT Falling Edge	Min = 10	Min = 5

7. The following A/D Converter Specifications have changed for the 80C196KB:

Parameter	80C196KB (Rev -001)	80C196KB (Rev -002)
Resolution	Min = 256 Levels 8 Bits	Min = 512 Levels 9 Bits
Full Scale Error	Typ = −0.5 ±0.5 LSB	Typ = 0.25 ±0.5 LSB
Zero Offset Error	Typ = ±0.5 LSB	Typ = −0.25 ±0.5 LSB

Differential Non-Linearity	Min = 0 LSB
	Max = ±2 LSB
		Min = > −1 LSB
		Max = +2 LSB

PRELIMINARY

87C196KB
16-BIT HIGH PERFORMANCE CHMOS MICROCONTROLLER

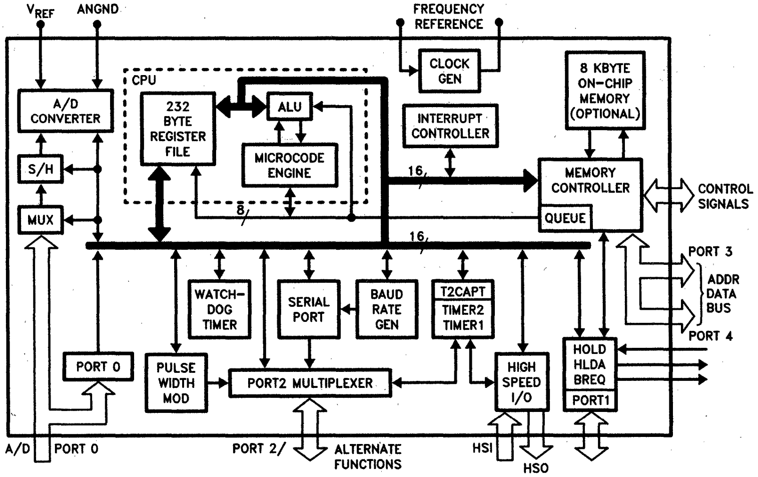
- **8 KBytes of On-Chip EPROM**
- **232 Byte Register File**
- **Register-to-Register Architecture**
- **28 Interrupt Sources/16 Vectors**
- **2.3 μs 16 x 16 Multiply (12 MHz)**
- **4.0 μs 32/16 Divide (12 MHz)**
- **Powerdown and Idle Modes**
- **Five 8-Bit I/O Ports**
- **16-Bit Watchdog Timer**
- **Dynamically Configurable 8-Bit or 16-Bit Buswidth**
- **Full Duplex Serial Port**
- **High Speed I/O Subsystem**
- **16-Bit Timer**
- **16-Bit Up/Down Counter with Capture**
- **Pulse-Width-Modulated Output**
- **Four 16-Bit Software Timers**
- **10-Bit A/D Converter with Sample/Hold**
- **$\overline{\text{HOLD}}/\overline{\text{HLDA}}$ Bus Protocol**
- **12 MHz Version—87C196KB12**
  10 MHz Version—87C196KB10

The 87C196KB is an 80C196KB 16-bit microcontroller with 8 Kbytes of on-chip EPROM. Both parts are high performance members of the MCS®-96 microcontroller family. The 87C196KB is compatible and uses a true superset of the 8096BH instructions. Intel's CHMOS process provides a high performance processor along with low power consumption. To further reduce power requirements, the processor can be placed into Idle or Powerdown Mode.

Bit, byte, word and some 32-bit operations are available on the 87C196KB. With a 12 MHz oscillator a 16-bit addition takes 0.66 μs, and the instruction times average 0.5 μs to 1.5 μs in typical applications.

Four high-speed capture inputs are provided to record times when events occur. Six high-speed outputs are available for pulse or waveform generation. The high-speed output can also generate four software timers or start an A/D conversion. Events can be based on the timer or up/down counter.

Also provided on-chip are an A/D converter, serial port, watchdog timer, and a pulse-width-modulated output signal.



270590-1

Figure 1. 87C196KB Block Diagram

ARCHITECTURE

The 87C196KB is a member of the MCS®-96 family, and as such has the same architecture and uses the same instruction set as the 8096BH. Many new features have been added on the 87C196KB including:

CPU FEATURES

Divide by 2 instead of divide by 3 clock for 1.5X performance

Faster instructions, especially indexed/indirect data operations

2.33 μs 16 × 16 multiply with 12 MHz clock (was 6.25 μs on the 8096BH)

Faster interrupt response (almost twice as fast as 8096BH)

Powerdown and Idle Modes

5 new instructions including Compare Long and Block Move

8 new interrupt vectors/6 new interrupt sources

PERIPHERAL FEATURES

SFR Window switching allows read-only registers to be written and vice-versa

Timer2 can count up or down by external selection

Timer2 has an independent capture register

HSO line events are stored in a register

HSO has CAM Lock and CAM Clear commands

New Baud Rate values are needed for serial port, higher speeds possible in all modes

Double buffered serial port transmit register

Serial Port Receive Overrun and Framing Error Detection

PWM has a Divide-by-2 Prescaler

$\overline{\text{HOLD}}/\overline{\text{HLDA}}$ Bus Protocol

PACKAGING

The 87C196KB is available in a 68-pin LCC (windowed) package and a 68-pin PLCC (One-Time Programmable) package. Contact your local sales office to determine the exact ordering code for the part desired.

PLCC	LCC	Description
9	1	ACH7/P0.7/PMD3
8	2	ACH6/P0.6/PMD2
7	3	ACH2/P0.2
6	4	ACH0/P0.0
5	5	ACH1/P0.1
4	6	ACH3/P0.3
3	7	NMI
2	8	$\overline{\text{EA}}$
1	9	V_{CC}
68	10	V_{SS}
67	11	XTAL1
66	12	XTAL2
65	13	CLKOUT
64	14	BUSWIDTH
63	15	INST
62	16	ALE/$\overline{\text{ADV}}$
61	17	$\overline{\text{RD}}$
60	18	AD0/P3.0
59	19	AD1/P3.1
58	20	AD2/P3.2
57	21	AD3/P3.3
56	22	AD4/P3.4
55	23	AD5/P3.5
54	24	AD6/P3.6
53	25	AD7/P3.7
52	26	AD8/P4.0
51	27	AD9/P4.1
50	28	AD10/P4.2
49	29	AD11/P4.3
48	30	AD12/P4.4
47	31	AD13/P4.5
46	32	AD14/P4.6
45	33	AD15/P4.7
44	34	T2CLK/P2.3
43	35	READY
42	36	T2RST/P2.4/$\overline{\text{AINC}}$
41	37	$\overline{\text{BHE}}$/$\overline{\text{WRH}}$
40	38	$\overline{\text{WR}}$/$\overline{\text{WRL}}$
39	39	PWM/P2.5
38	40	P2.7/T2CAPTURE/$\overline{\text{PACT}}$
37	41	V_{PP}
36	42	V_{SS}
35	43	HSO.3
34	44	HSO.2
33	45	P2.6/T2UP-DN
32	46	P1.7/$\overline{\text{HOLD}}$
31	47	P1.6/$\overline{\text{HLDA}}$
30	48	P1.5/$\overline{\text{BREQ}}$
29	49	HSO.1
28	50	HSO.0
27	51	HSO.5/HSI.3/SID3
26	52	HSO.4/HSI.2/SID2
25	53	HSI.1/SID1
24	54	HSI.0/SID0
23	55	P1.4
22	56	P1.3
21	57	P1.2
20	58	P1.1
19	59	P1.0
18	60	TXD/P2.0/PVER
17	61	RXD/P2.1/$\overline{\text{PALE}}$
16	62	$\overline{\text{RESET}}$
15	63	EXTINT/P2.2/$\overline{\text{PROG}}$
14	64	V_{SS}[(1)]
13	65	V_{REF}
12	66	ANGND
11	67	ACH4/P0.4/PMD0
10	68	ACH5/P0.5/PMD1

Figure 2. Pin Definitions

NOTE:
1. This pin was formerly the Clock Detect Enable pin. The CDE function is not guaranteed to work. To ensure proper 87C196KB operation, the pin must be directly connected to V_{SS}.

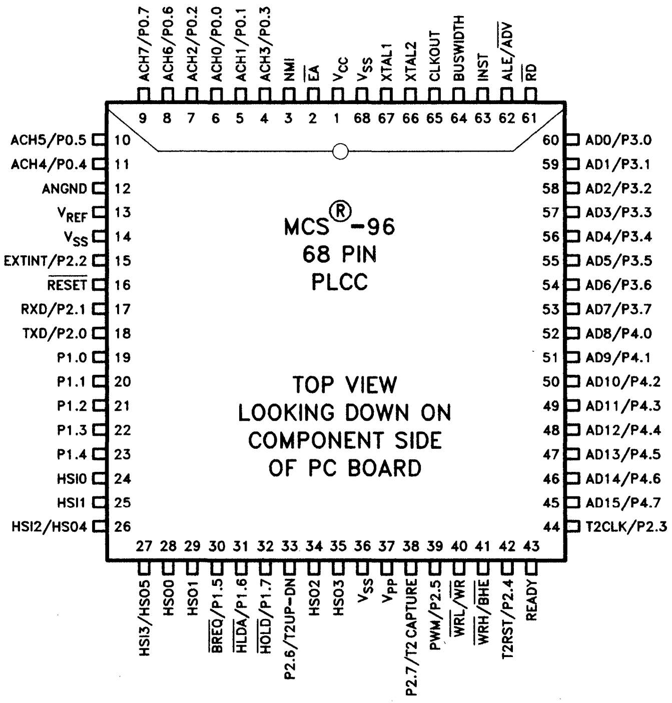


Figure 3. 68-Pin Package (PLCC—Top View)

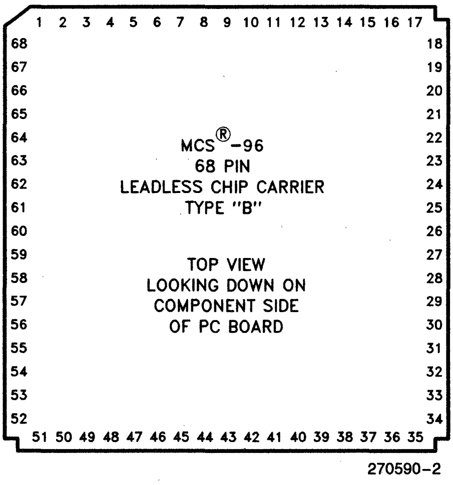


Figure 4. 68-Pin Package (LCC—Top View)

Thermal Characteristics

Package Type	θ_{JA}	θ_{JC}
PLCC	35°C/W	12°C/W
LCC	28°C/W	3.5°C/W

PIN DESCRIPTIONS

Symbol	Name and Function
V_{CC}	Main supply voltage (5V).
V_{SS}	Digital circuit ground (0V). There are two V_{SS} pins, both of which must be connected.
V_{REF}	Reference voltage for the A/D converter (5V). V_{REF} is also the supply voltage to the analog portion of the A/D converter and the logic used to read Port 0. Must be connected for A/D and Port 0 to function.
ANGND	Reference ground for the A/D converter. Must be held at nominally the same potential as V_{SS}.
V_{PP}	Timing pin for the return from powerdown circuit. Connect this pin with a 1 μF capacitor to V_{SS}. If this function is not used V_{PP} may be tied to V_{CC}. This pin is the programming voltage on the EPROM device.
XTAL1	Input of the oscillator inverter and of the internal clock generator.
XTAL2	Output of the oscillator inverter.
CLKOUT	Output of the internal clock generator. The frequency of CLKOUT is ½ the oscillator frequency. It has a 50% duty cycle.
$\overline{\text{RESET}}$	Reset input to the chip. Input low for at least 4 state times to reset the chip. The subsequent low-to-high transition re- synchronizes CLKOUT and commences a 10-state-time sequence in which the PSW is cleared, a byte read from 2018H loads CCR, and a jump to location 2080H is executed. Input high for normal operation. $\overline{\text{RESET}}$ has an internal pullup.
BUSWIDTH	Input for buswidth selection. If CCR bit 1 is a one, this pin selects the bus width for the bus cycle in progress. If BUSWIDTH is a 1, a 16-bit bus cycle occurs. If BUSWIDTH is a 0 an 8-bit cycle occurs. If CCR bit 1 is a 0, the bus is always an 8-bit bus.
NMI	A positive transition causes a vector through 203EH.
INST	Output high during an external memory read indicates the read is an instruction fetch and output low indicates a data fetch. INST is valid throughout the bus cycle. INST is activated only during external memory accesses.
$\overline{\text{EA}}$	Input for memory select (External Access). $\overline{\text{EA}}$ equal to a TTL-high causes memory accesses to locations 2000H through 3FFFH to be directed to on-chip ROM/EPROM. $\overline{\text{EA}}$ equal to a TTL-low causes accesses to these locations to be directed to off-chip memory.
ALE/$\overline{\text{ADV}}$	Address Latch Enable or Address Valid output, as selected by CCR. Both pin options provide a latch to demultiplex the address from the address/data bus. When the pin is $\overline{\text{ADV}}$, it goes inactive high at the end of the bus cycle. $\overline{\text{ADV}}$ can be used as a chip select for external memory. ALE/$\overline{\text{ADV}}$ is activated only during external memory accesses.
$\overline{\text{RD}}$	Read signal output to external memory. $\overline{\text{RD}}$ is activated only during external memory reads.
$\overline{\text{WR}}$/$\overline{\text{WRL}}$	Write and Write Low output to external memory, as selected by the CCR. $\overline{\text{WR}}$ will go low for every external write, while $\overline{\text{WRL}}$ will go low only for external writes where an even byte is being written. $\overline{\text{WR}}$/$\overline{\text{WRL}}$ is activated only during external memory writes.
$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	Bus High Enable or Write High output to external memory, as selected by the CCR. $\overline{\text{BHE}}$ = 0 selects the bank of memory that is connected to the high byte of the data bus. A0 = 0 selects the bank of memory that is connected to the low byte of the data bus. Thus accesses to a 16-bit wide memory can be to the low byte only (A0 = 0, $\overline{\text{BHE}}$ = 1), to the high byte only (A0 = 1, $\overline{\text{BHE}}$ = 0), or both bytes (A0 = 0, $\overline{\text{BHE}}$ = 0). If the $\overline{\text{WRH}}$ function is selected, the pin will go low if the bus cycle is writing to an odd memory location. $\overline{\text{BHE}}$/$\overline{\text{WRH}}$ is valid only during 16-bit external memory write cycles.

PIN DESCRIPTIONS (Continued)

Symbol	Name and Function
READY	Ready input to lengthen external memory cycles, for interfacing to slow or dynamic memory, or for bus sharing. If the pin is high, CPU operation continues in a normal manner. If the pin is low prior to the falling edge of CLKOUT, the memory controller goes into a wait mode until the next positive transition in CLKOUT occurs with READY high. When the external memory is not being used, READY has no effect. Internal control of the number of wait states inserted into a bus cycle (held not ready) is available through configuration of CCR.
HSI	Inputs to High Speed Input Unit. Four HSI pins are available: HSI.0, HSI.1, HSI.2, and HSI.3. Two of them (HSI.2 and HSI.3) are shared with the HSO Unit. The HSI pins are also used as the SID in Slave Programming Mode.
HSO	Outputs from High Speed Output Unit. Six HSO pins are available: HSO.0, HSO.1, HSO.2, HSO.3, HSO.4, and HSO.5. Two of them (HSO.4 and HSO.5) are shared with the HSI Unit.
Port 0	8-bit high impedance input-only port. Three pins can be used as digital inputs and/or as analog inputs to the on-chip A/D converter. These pins set the Programming Mode.
Port 1	8-bit quasi-bidirectional I/O port. These pins are shared with $\overline{\text{HOLD}}$, $\overline{\text{HLDA}}$ and $\overline{\text{BREQ}}$.
Port 2	8-bit multi-functional port. All of its pins are shared with other functions in the 87C196KB.
Ports 3 and 4	8-bit bi-directional I/O ports with open drain outputs. These pins are shared with the multiplexed address/data bus which has strong internal pullups.
$\overline{\text{HOLD}}$	Bus Hold input requesting control of the bus. Enabled by setting WSR.7.
$\overline{\text{HLDA}}$	Bus Hold acknowledge output indicating release of the bus. Enabled by setting WSR.7.
$\overline{\text{BREQ}}$	Bus Request output activated when the bus controller has a pending external memory cycle. Enabled by setting WSR.7.
TxD	The TxD pin is used for serial port transmission in Modes 1, 2, and 3. The TxD function is enabled by setting IOC1.5. In mode 0 the pin is used as the serial clock output.
RxD	Serial Port Receive pin used for serial port reception. The RxD function is enabled by setting SPCON.3. In mode 0 the pin functions as input or output data.
EXTINT	A rising edge on the EXTINT pin will generate an external interrupt. EXTINT is selected as the external interrupt source by setting IOC1.1 high.
T2CLK	The T2CLK pin is the Timer2 clock input or the serial port baud rate generator input.
T2RST	A rising edge on the T2RST pin will reset Timer2. The external reset function is enabled by setting IOCO.3. T2RST is enabled as the reset source by clearing IOCO.5.

PIN DESCRIPTIONS (Continued)

Symbol	Name and Function
PWM	Port 2.5 can be enabled as a PWM output by setting IOC1.0. The duty cycle of the PWM is determined by the value loaded into the PWM-CONTROL register (17H).
T2UPDN	The T2UPDN pin controls the direction of Timer2 as an up or down counter. The Timer2 up/down function is enabled by setting IOC2.1.
T2CAP	A rising edge on P2.7 will capture the value of Timer2 in the T2CAPTURE register (location 0CH in Window 15).
PMODE	Programming Mode Select. Determines the EPROM programming algorithm that is performed. PMODE is sampled after a chip reset and should be static while the part is operating.
SID	Slave ID Number. Used to assign each slave a pin of Port 3 or 4 to use for passing programming verification acknowledgement. For example, if gang programming in the Slave Programming Mode, the slave with SID = 001 will use Port 3.1 to signal correct or incorrect program verification.
$\overline{\text{PALE}}$	Programming ALE Input. Accepted by an 87C196KB that is in Slave Programming Mode. Used to indicate that Ports 3 and 4 contain a command/address.
$\overline{\text{PROG}}$	Programming. Falling edge indicates valid data on PBUS and the beginning of programming. Rising edge indicates end of programming.
$\overline{\text{PACT}}$	Programming Active. Used in the Auto Programming Mode to indicate when programming activity is complete.
$\overline{\text{PVAL}}$	Program Valid. This signal indicates the success or failure of programming in the Auto Programming Mode. A zero indicates successful programming.
PVER	Program Verification. Used in Slave Programming and Auto CLB Programming Modes. Signal is low after rising edge of PROG if the programming was not successful.
$\overline{\text{AINC}}$	Auto Increment. Active low signal indicates that the auto increment mode is enabled. Auto Increment will allow reading or writing of sequential EPROM locations without address transactions across the PBUS for each read or write.
PORTS	Address/Command/Data Bus. Used to pass commands, addresses, and data to and from slave mode 87C196KBs. Used by chips in Auto Programming Mode to pass command, addresses and data to slaves. Also used in the Auto Programming Mode as a regular system bus to access external memory. Should have pullups to V_{CC} (15 kΩ).

NEW INSTRUCTIONS

The following five instructions have been added to the 8096BH instruction set for the 87C196KB.

PUSHA — PUSHes the PSW, IMASK, IMASK1, and WSR
(Used instead of PUSHF when new interrupts and registers are used.)

assembly language format: PUSHA

object code format: <11110100>

bytes: 1

states: on-chip stack: 12
off-chip stack: 18

POPA — POPs the PSW, IMASK, IMASK1, and WSR
(Used instead of POPF when new interrupts and registers are used.)

assembly language format: POPA

object code format: <11110101>

bytes: 1

states: on-chip stack: 12
off-chip stack:18

IDLPD — Sets the part into Idle or Powerdown Mode

assembly language format: IDLPD #key (key=1 for Idle, key=2 for Powerdown.)

object code format: <11110110> <key>

bytes: 2

states: legal key: 8
illegal key: 25

CMPL — Compare 2 long direct values

```
assembly language format:        DST     SRC
                         CMPL    Lreg,   Lreg
```

object code format: <11000101> <src Lreg> <dst Lreg>

bytes: 3

states: 7

BMOV — Block move using 2 auto-incrementing pointers and a counter

```
assembly language format:        PTRS    CNTREG
                         BMOV    Lreg,   wreg
```

object code format: <11000001> <wreg> <Lreg>

bytes: 3

states: internal/internal: 8 per transfer + 6
external/internal: 11 per transfer + 6
external/external: 14 per transfer + 6

USING THE ALTERNATE REGISTER WINDOW (WSR = 15)

I/O register expansion on the new CHMOS members of the MCS-96 family has been provided by making three register windows available. Switching between these windows is done using the Window Select Register (WSR). The PUSHA and POPA instructions can be used to push and pop the WSR and second interrupt mask when entering or leaving interrupts, so it is easy to change between windows.

On the 87C196KB only Window 0, Window 14 and Window 15 are active. Window 0 is a true superset of the standard 8096BH SFR space, while Window 15 allows the read-only registers to be written and write-only registers to be read. The only major exception to this is the Timer2 register which is the Timer2 capture register in Window 15. The writeable register for Timer2 is in Window 0. There are also some minor changes and cautions. Window 14 contains the Programmable Pulse Width register (PPW) at location 14H. The descriptions of the registers which have different functions in Window 15 than in Window 0 are listed below:

AD_COMMAND (02H) — Read the last written command

AD_RESULT (02H, 03H) — Write a value into the result register

HSI_MODE (03H) — Read the value in HSI_MODE

HSI_TIME (04H,05H) — Write to FIFO Holding register

HSO_TIME (04H,05H) — Read the last value placed in the holding register

HSI_STATUS (06H) — Write to status bits but not to HSI pin bits. (Pin bits are 1,3,5,7).

HSO_COMMAND (06H) — Read the last value placed in the holding register

SBUF(RX) (07H) — Write a value into the receive buffer

SBUF(TX) (07H) — Read the last value written to the transmit buffer

WATCHDOG(0AH) — Read the value in the upper byte of the WDT

TIMER1 (0AH,0BH) — Write a value to Timer1

TIMER2 (0CH,0DH) — Read/Write the Timer2 capture register.
Note that Timer2 read/write is done with WSR = 0.

IOC2 (0BH) — Last written value is readable, except bit 7 (note 1)

BAUDRATE (0EH) — No function, cannot be read

PORT0 (0EH) — No function, no output drivers on the pins. Registers reserved.

PORT1 (0FH) — IOPORT1 cannot be read or written in Window 15. Register reserved.

SP_STAT (11H) — Set the status bits, TI and RI can be set, but it will not cause an interrupt

SP_CON (11H) — Read the current control byte

IOS0 (15H) — Writing to this register controls the HSO pins. Bits 6 and 7 are inactive for writes.

IOC0 (15H) — Last written value is readable, except bit 1 (note 1)

IOS1 (16H) — Writing to this register will set the status bits, but not cause interrupts. Bits 6 and 7 are not functional

IOC1 (16H) — Last written value is readable

IOS2 (17H) — Writing to this register will set the status bits, but not cause interrupts.

PWM_CONTROL (17H) — Read the duty cycle value written to PWM_CONTROL

NOTE:

1. IOC2.7 (CAM CLEAR) and IOC0.1 (T2RST) are not latched and will read as a 1 (precharged bus).

Being able to write to the read-only registers and vice-versa provides a lot of flexibility. One of the most useful advantages is the ability to set the timers and HSO lines for initial conditions other than zero.

Reserved registers may be used for testing or future features. Do not write to these registers. Reads from reserved registers will return indeterminate values.

MEMORY MAP

Memory	Address
	0FFFFH
EXTERNAL MEMORY OR I/O	
	4000H
INTERNAL ROM/EPROM OR EXTERNAL MEMORY	
	2080H
RESERVED	
	2040H
UPPER 8 INTERRUPT VECTORS	
	2030H
ROM/EPROM SECURITY KEY	
	2020H
RESERVED	
	2019H
CHIP CONFIGURATION BYTE	
	2018H
RESERVED	
	2014H
LOWER 8 INTERRUPT VECTORS PLUS 2 SPECIAL INTERRUPTS	
	2000H
PORT 3 AND PORT 4	
	1FFEH
EXTERNAL MEMORY OR I/O	
	0100H
INTERNAL DATA MEMORY - REGISTER FILE (STACK POINTER, RAM AND SFRS) EXTERNAL PROGRAM CODE MEMORY	
	0000H

87C196KB INTERRUPTS

Number	Source	Vector Location	Priority
INT15	NMI	203EH	15
INT14	HSI FIFO Full	203CH	14
INT13	EXTINT Pin	203AH	13
INT12	TIMER2 Overflow	2038H	12
INT11	TIMER2 Capture	2036H	11
INT10	4th Entry into HSI FIFO	2034H	10
INT09	RI	2032H	9
INT08	TI	2030H	8
SPECIAL	Unimplemented Opcode	2012H	N/A
SPECIAL	Trap	2010H	N/A
INT07	EXTINT	200EH	7
INT06	Serial Port	200CH	6
INT05	Software Timer	200AH	5
INT04	HSI.0 Pin	2008H	4
INT03	High Speed Outputs	2006H	3
INT02	HSI Data Available	2004H	2
INT01	A/D Conversion Complete	2002H	1
INT00	Timer Overflow	2000H	0

WSR = 0

Address	WHEN READ	WHEN WRITTEN
19H 18H	STACK POINTER	STACK POINTER
17H	*IOS2	PWM__CONTROL
16H	IOS1	IOC1
15H	IOS0	IOC0
14H	*WSR	*WSR
13H	*INT__MASK1	*INT__MASK1
12H	*INT__PEND1	*INT__PEND1
11H	*SP__STAT	*SP__CON
10H	PORT2	PORT2
0FH	PORT1	PORT1
0EH	PORT0	BAUD RATE
0DH	TIMER2 (HI)	TIMER2 (HI)
0CH	TIMER2 (LO)	TIMER2 (LO)
0BH	TIMER1 (HI)	*IOC2
0AH	TIMER1 (LO)	WATCHDOG
09H	INT__PEND	INT__PEND
08H	INT__MASK	INT__MASK
07H	SBUF(RX)	SBUF(TX)
06H	HSI__STATUS	HSO__COMMAND
05H	HSI__TIME (HI)	HSO__TIME (HI)
04H	HSI__TIME (LO)	HSO__TIME (LO)
03H	AD__RESULT (HI)	HSI__MODE
02H	AD__RESULT (LO)	AD__COMMAND
01H	ZERO REG (HI)	ZERO REG (HI)
00H	ZERO REG (LO)	ZERO REG (LO)

WSR = 15

Address	Register
10H	RESERVED(1)
0FH	RESERVED(1)
0EH	RESERVED(1)
0DH	*T2 CAPTURE (HI)
0CH	*T2 CAPTURE (LO)

OTHER SFRS IN WSR 15 BECOME READABLE: IF THEY WERE WRITABLE IN WSR = 0 AND WRITABLE IF THEY WERE READABLE IN WSR = 0

WSR = 14

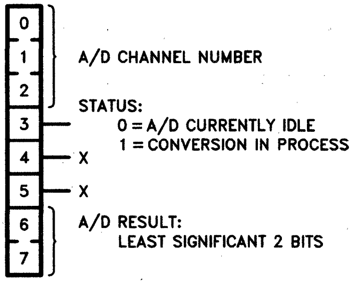
Address	Register
4H	PPW

*NEW OR CHANGED REGISTER FUNCTION FROM 8096BH

NOTE:
1. RESERVED REGISTERS SHOULD NOT BE WRITTEN

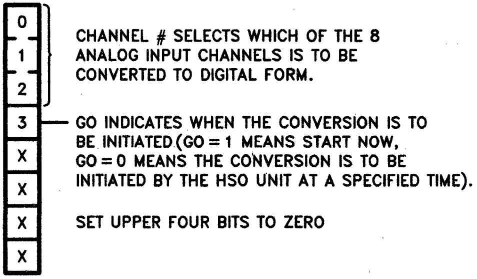
SFR BIT SUMMARY

AD_RESULT (LO) (02H)



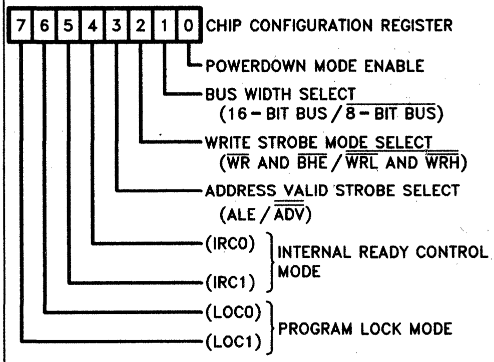
270590–42

AD_COMMAND


0
1
2
3
X
X
X
X
CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.
GO INDICATES WHEN THE CONVERSION IS TO BE INITIATED (GO = 1 MEANS START NOW, GO = 0 MEANS THE CONVERSION IS TO BE INITIATED BY THE HSO UNIT AT A SPECIFIED TIME).
SET UPPER FOUR BITS TO ZERO

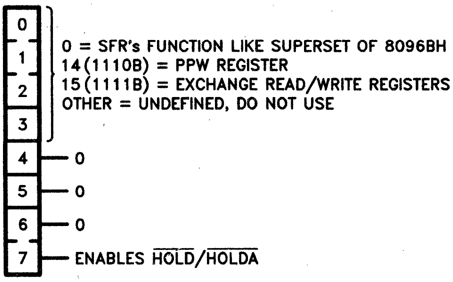
270590–43

CCR (2018H)


7
6
5
4
3
2
1
0
CHIP CONFIGURATION REGISTER
POWERDOWN MODE ENABLE
BUS WIDTH SELECT (16-BIT BUS/$\overline{\text{8-BIT BUS}}$)
WRITE STROBE MODE SELECT ($\overline{\text{WR}}$ AND $\overline{\text{BHE}}$/$\overline{\text{WRL}}$ AND $\overline{\text{WRH}}$)
ADDRESS VALID STROBE SELECT (ALE/$\overline{\text{ADV}}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE

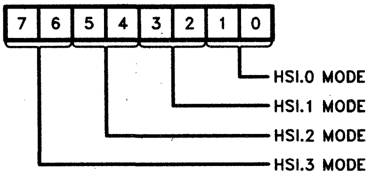
270590–44

WSR (14H)


0
1
2
3
4
5
6
7
0 = SFR's FUNCTION LIKE SUPERSET OF 8096BH
14(1110B) = PPW REGISTER
15(1111B) = EXCHANGE READ/WRITE REGISTERS
OTHER = UNDEFINED, DO NOT USE
0
0
0
ENABLES $\overline{\text{HOLD}}$/$\overline{\text{HOLDA}}$

270590–45

HSI_MODE

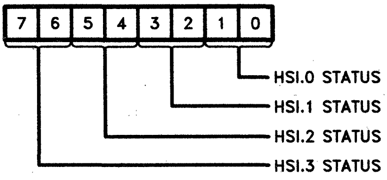


WHERE EACH 2-BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

270590–46

HSI_STATUS (06H)



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT INDICATES THE CURRENT STATUS OF THE PIN.

270590–47

SFR BIT SUMMARY

IOS0 (15H)

0	HSO.0 CURRENT STATE
1	HSO.1 CURRENT STATE
2	HSO.2 CURRENT STATE
3	HSO.3 CURRENT STATE
4	HSO.4 CURRENT STATE
5	HSO.5 CURRENT STATE
6	CAM OR HOLDING REGISTER IS FULL
7	HSO HOLDING REGISTER IS FULL

270590–31

IOC0 (15H)

0	HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1	TIMER 2 RESET EACH WRITE
2	HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3	TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4	HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5	TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6	HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7	TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$

270590–32

IOS1 (16H)

0	SOFTWARE TIMER 0 EXPIRED
1	SOFTWARE TIMER 1 EXPIRED
2	SOFTWARE TIMER 2 EXPIRED
3	SOFTWARE TIMER 3 EXPIRED
4	TIMER 2 HAS OVERFLOW
5	TIMER 1 HAS OVERFLOW
6	HSI FIFO IS FULL
7	HSI HOLDING REGISTER DATA AVAILABLE

270590–33

IOC1 (16H)

0	SELECT PWM / $\overline{\text{SELECT P2.5}}$
1	EXTERNAL INTERRUPT ACH7 / $\overline{\text{EXTINT}}$
2	TIMER 1 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
3	TIMER 2 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
4	HSO.4 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
5	SELECT TXD / $\overline{\text{SELECT P2.0}}$
6	HSO.5 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
7	HSI INTERRUPT FIFO FULL / $\overline{\text{HOLDING REGISTER LOADED}}$

270590–34

IOS2 (17H)

INDICATES WHICH HSO EVENT OCCURED

0	HSO.0
1	HSO.1
2	HSO.2
3	HSO.3
4	HSO.4
5	HSO.5
6	T2RESET
7	START A/D

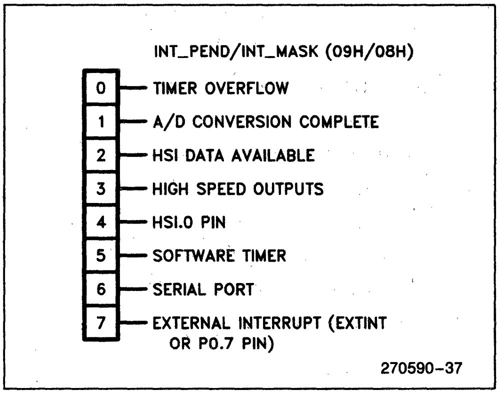
270590–35

IOC2 (0BH)

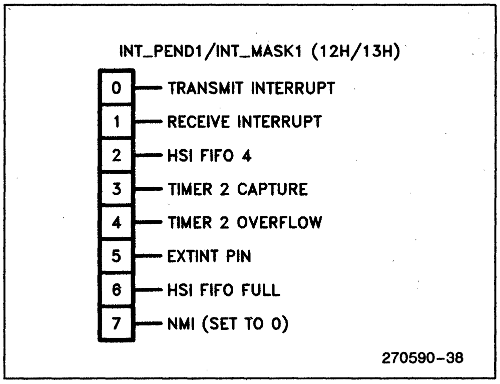
0	ENABLE FAST INCREMENT OF T2
1	ENABLE T2 AS UP/DOWN COUNTER
2	ENABLE ÷2 PRESCALER ON PWM
3	X (SET TO 0)
4	A/D CLOCK PRESCALER DISABLE
5	T2 ALTERNATE INTERRUPT @ 8000H
6	ENABLE LOCKED CAM ENTRIES
7	CLEAR ENTIRE CAM

270590–36

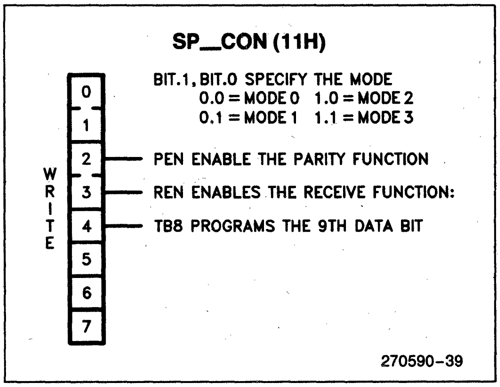
SFR BIT SUMMARY



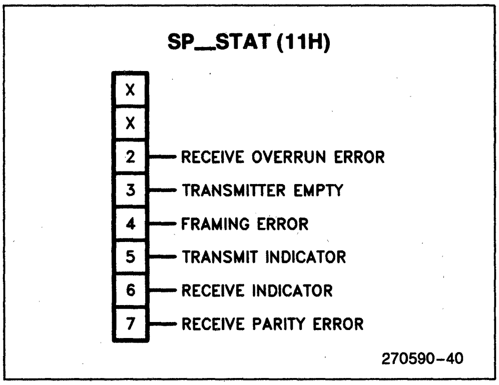
270590-37



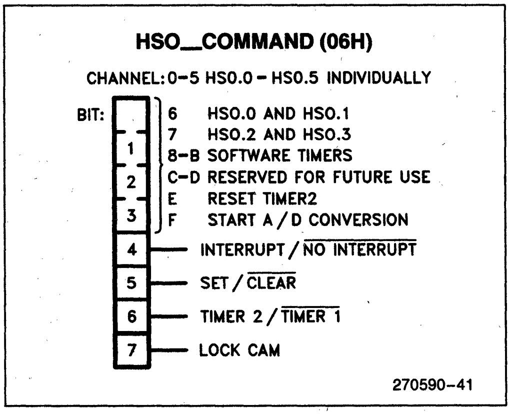
270590-38


SP_CON (11H)
BIT.1, BIT.0 SPECIFY THE MODE
0.0 = MODE 0 1.0 = MODE 2
0.1 = MODE 1 1.1 = MODE 3
WRITE
0
1
2 PEN ENABLE THE PARITY FUNCTION
3 REN ENABLES THE RECEIVE FUNCTION:
4 TB8 PROGRAMS THE 9TH DATA BIT
5
6
7

270590-39



270590-40


HSO_COMMAND (06H)
CHANNEL: 0–5 HSO.0 – HSO.5 INDIVIDUALLY
BIT:
1
2
3
6 HSO.0 AND HSO.1
7 HSO.2 AND HSO.3
8–B SOFTWARE TIMERS
C–D RESERVED FOR FUTURE USE
E RESET TIMER2
F START A/D CONVERSION
4 INTERRUPT / $\overline{\text{NO INTERRUPT}}$
5 SET / $\overline{\text{CLEAR}}$
6 TIMER 2 / $\overline{\text{TIMER 1}}$
7 LOCK CAM

270590-41

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings*

Ambient Temperature
Under Bias .... 0°C to +70°C

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

Voltage On Any Pin to V_{SS} .... −0.5V to +7.0V

Power Dissipation 1.5W

**Notice: Stresses 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 specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

NOTICE: Specifications contained within the following tables are subject to change.

Operating Conditions

Symbol	Description	Min	Max	Units
T_A	Ambient Temperature Under Bias	0	+70	°C
V_{CC}	Digital Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	3.5	12	MHz

NOTE:
ANGND and V_{SS} should be nominally at the same potential.

D.C. Characteristics (Over specified operating conditions)

Symbol	Description	Min	Max	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage (Note 1)	0.2 V_{CC} + 0.9	V_{CC} + 0.5	V	
V_{IH1}	Input High Voltage on XTAL 1	0.7 V_{CC}	V_{CC} + 0.5	V	
V_{IH2}	Input High Voltage on RESET	2.2	V_{CC} + 0.5	V	
V_{OL}	Output Low Voltage		0.3 0.45 1.5	V V V	I_{OL} = 200 μA I_{OL} = 3.2 mA I_{OL} = 7 mA
V_{OH}	Output High Voltage (Standard Outputs)	V_{CC} − 0.3 V_{CC} − 0.7 V_{CC} − 1.5		V V V	I_{OH} = −200 μA I_{OH} = −3.2 mA I_{OH} = −7 mA
V_{OH1}	Output High Voltage (Quasi-bidirectional Outputs)	V_{CC} − 0.3 V_{CC} − 0.7 V_{CC} − 1.5		V V V	I_{OH} = −10 μA I_{OH} = −30 μA I_{OH} = −60 μA
I_{LI}	Input Leakage Current (Std. Inputs)		±10	μA	$0 < V_{IN} < V_{CC} - 0.3V$
I_{LI1}	Input Leakage Current (Port 0)		+3	μA	$0 < V_{IN} < V_{REF}$
I_{TL}	1 to 0 Transition Current (QBD Pins)		−650	μA	V_{IN} = 2.0V
I_{IL}	Logical 0 Input Current (QBD Pins)		−50	μA	V_{IN} = 0.45V
I_{IL1}	Logical 0 Input Current in Reset (Note 2) (ALE, $\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, INST, P2.0, Port 3/4)		−950	μA	V_{IN} = 0.45 V

NOTE:
1. All pins except RESET and XTAL1.
2. Holding these pins below V_{IH} in Reset may cause the part to enter test modes.

D.C. Characteristics (Over specified operating conditions) (Continued)

Symbol	Description	Min	Typ(7)	Max	Units	Test Conditions
I_{CC}	Active Mode Current in Reset		40	55	mA	XTAL1 = 12 MHz $V_{CC} = V_{PP} = V_{REF} = 5.5V$
I_{REF}	A/D Converter Reference Current		2	5	mA	
I_{IDLE}	Idle Mode Current		10	22	mA	
I_{CC1}	Active Mode Current		15	22	mA	XTAL1 = 3.5 MHz
I_{PD}(8)	Powerdown Mode Current		5		μA	$V_{CC} = V_{PP} = V_{REF} = 5.5V$
R_{RST}	Reset Pullup Resistor	6K		50K	Ω	
C_S	Pin Capacitance (Any Pin to V_{SS})			10	pF	$f_{TEST} = 1.0$ MHz

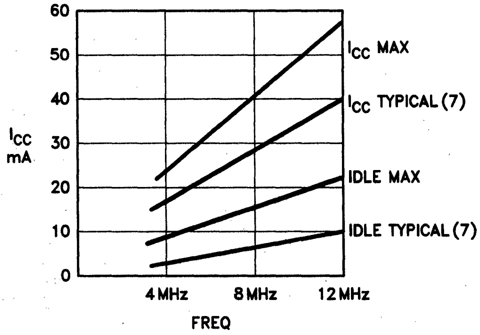
NOTES:

(Notes apply to all specifications)

1. QBD (Quasi-bidirectional) pins include Port 1, P2.6 and P2.7.
2. Standard Outputs include AD0–15, $\overline{RD}$, $\overline{WR}$, ALE, $\overline{BHE}$, INST, HSO pins, PWM/P2.5, CLKOUT, RESET, Ports 3 and 4, TXD/P2.0, and RXD (in serial mode 0). The V_{OH} specification is not valid for RESET. Ports 3 and 4 are open-drain outputs.
3. Standard Inputs include HSI pins, CDE, $\overline{EA}$, READY, BUSWIDTH, NMI, RXD/P2.1, EXTINT/P2.2, T2CLK/P2.3, and T2RST/P2.4.
4. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V or V_{OH} is held below V_{CC} − 0.7V:
   I_{OL} on Output pins: 10 mA
   I_{OH} on quasi-bidirectional pins: self limiting
   I_{OH} on Standard Output pins: 10 mA
5. Maximum current per bus pin (data and control) during normal operation is ±3.2 mA.
6. During normal (non-transient) conditions the following total current limits apply:

Port 1, P2.6	I_{OL}: 29 mA	I_{OH} is self limiting
HSO, P2.0, RXD, $\overline{RESET}$	I_{OL}: 29 mA	I_{OH}: 26 mA
P2.5, P2.7, $\overline{WR}$, $\overline{BHE}$	I_{OL}: 13 mA	I_{OH}: 11 mA
AD0–AD15	I_{OL}: 52 mA	I_{OH}: 52 mA
$\overline{RD}$, ALE, INST–CLKOUT	I_{OL}: 13 mA	I_{OH}: 13 mA

7. Typicals are based on a limited number of samples and are not guaranteed. The values listed are at room temperature and $V_{REF} = V_{CC} = 5V$.
8. I_{PD} is not guaranteed on the standard 87C196KB part and may exceed 100 μA on some parts. Customers whose applications use the power down mode and require a guaranteed maximum value of I_{PD} should order the S-spec part, contact an Intel Field Representative.



270590–20

I_{CC} Max = 3.88 × FREQ + 8.43
I_{IDLE} Max = 1.65 × FREQ + 2.2

Figure 5. I_{CC} and I_{IDLE} vs. Frequency

A.C. Characteristics (Over specified operating conditions)

Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, f_{OSC} = 12 MHz

The system must meet these specifications to work with the 87C196KB:

Symbol	Description	Min	Max	Units	Notes
T_{AVYV}	Address Valid to READY Setup 87C196KB10 87C196KB12		 $2T_{OSC} - 90$ $2T_{OSC} - 85$	 ns ns	
T_{LLYV}	ALE Low to READY Setup: 87C196KB10 87C196KB12		 $T_{OSC} - 80$ $T_{OSC} - 72$	 ns ns	
T_{YLYH}	NonREADY Time	No upper limit		ns	
T_{CLYX}	READY Hold after CLKOUT Low	0	$T_{OSC} - 30$	ns	(Note 1)
T_{LLYX}	READY Hold after ALE Low	$T_{OSC} - 15$	$2T_{OSC} - 40$	ns	(Note 1)
T_{AVGV}	Address Valid to Buswidth Setup		$2T_{OSC} - 85$	ns	
T_{LLGV}	ALE Low to Buswidth Setup		$T_{OSC} - 70$	ns	
T_{CLGX}	Buswidth Hold after CLKOUT Low	0		ns	
T_{AVDV}	Address Valid to Input Data Valid 87C196 KB10 87C196 KB12		 $3T_{OSC} - 70$ $3T_{OSC} - 67$	 ns ns	(Note 2)
T_{RLDV}	$\overline{RD}$ Active to Input Data Valid 87C196KB10 87C196KB12		 $T_{OSC} - 30$ $T_{OSC} - 23$	 ns ns	(Note 2)
T_{CLDV}	CLKOUT Low to Input Data Valid		$T_{OSC} - 50$	ns	
T_{RHDZ}	End of $\overline{RD}$ to Input Data Float		$T_{OSC} - 20$	ns	
T_{RXDX}	Data Hold after $\overline{RD}$ Inactive	0		ns	

NOTES:

1. If max is exceeded, additional wait states will occur.
2. When using wait states, add $2T_{OSC}$ × n where n = number of wait states.

A.C. Characteristics (Over specified operating conditions) (Continued)

Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, f_{OSC} = 12 MHz

The 87C196KB will meet these specifications:

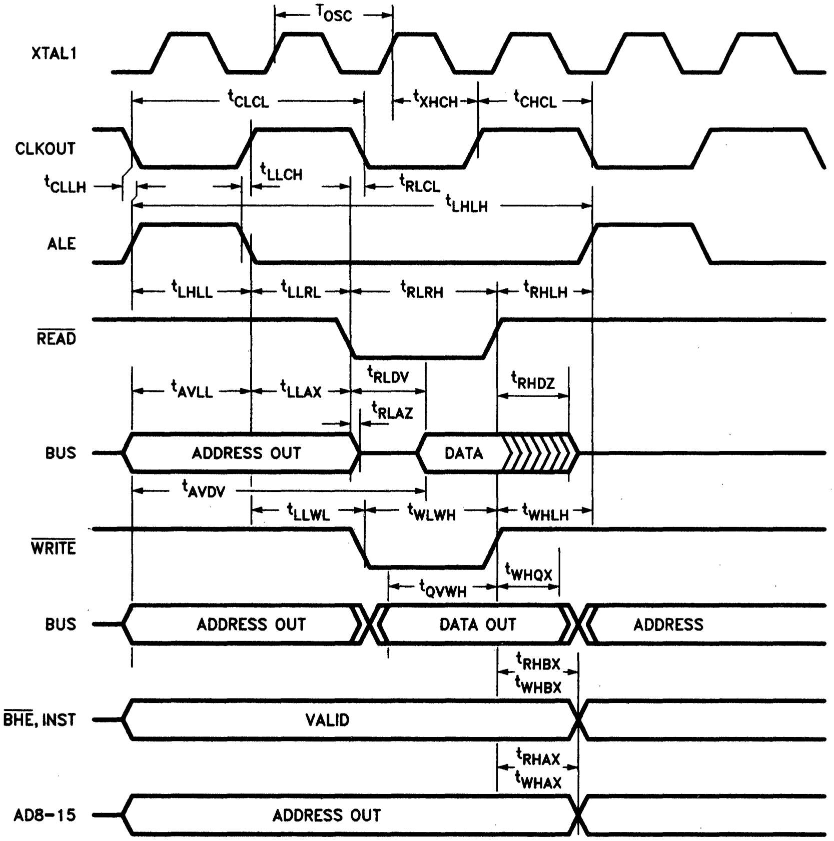
Symbol	Description	Min	Max	Units	Notes
F_{XTAL}	Frequency on XTAL1: 87C196KB10 87C196KB12	 3.5 3.5	 10.0 12.0	 MHz MHz	(Note 3)
T_{OSC}	$1/F_{XTAL}$: 87C196KB10 87C196KB12	 100 83	 286 286	 ns ns	
T_{XHCH}	XTAL1 High to CLKOUT High or Low	40	110	ns	(Note 1)
T_{CLCL}	CLKOUT Cycle Time	$2T_{OSC}$		ns	
T_{CHCL}	CLKOUT High Period	$T_{OSC} - 10$	$T_{OSC} + 10$	ns	
T_{CLLH}	CLKOUT Falling Edge to ALE Rising	−5	15	ns	
T_{LLCH}	ALE Falling Edge to CLKOUT Rising	−15	15	ns	
T_{LHLH}	ALE Cycle Time	$4T_{OSC}$		ns	(Note 4)
T_{LHLL}	ALE High Period	$T_{OSC} - 10$	$T_{OSC} + 10$	ns	
T_{AVLL}	Address Setup to ALE Falling Edge	$T_{OSC} - 20$		ns	
T_{LLAX}	Address Hold after ALE Falling Edge	$T_{OSC} - 40$		ns	
T_{LLRL}	ALE Falling Edge to $\overline{RD}$ Falling Edge	$T_{OSC} - 40$		ns	
T_{RLCL}	$\overline{RD}$ Low to CLKOUT Falling Edge	5	30	ns	
T_{RLRH}	$\overline{RD}$ Low Period	$T_{OSC} - 5$	$T_{OSC} + 25$	ns	(Note 4)
T_{RHLH}	$\overline{RD}$ Rising Edge to ALE Rising Edge	T_{OSC}	$T_{OSC} + 25$	ns	(Note 2)
T_{RLAZ}	$\overline{RD}$ Low to Address Float		10	ns	
T_{LLWL}	ALE Falling Edge to $\overline{WR}$ Falling Edge	$T_{OSC} - 10$		ns	
T_{CLWL}	CLKOUT Low to $\overline{WR}$ Falling Edge	0	25	ns	
T_{QVWH}	Data Stable to $\overline{WR}$ Rising Edge: 87C196KB10 87C196KB12	 $T_{OSC} - 30$ $T_{OSC} - 23$		 ns ns	(Note 4)
T_{CHWH}	CLKOUT High to $\overline{WR}$ Rising Edge	−10	10	ns	
T_{WLWH}	$\overline{WR}$ Low Period	$T_{OSC} - 30$	$T_{OSC} + 5$	ns	(Note 4)
T_{WHQX}	Data Hold after $\overline{WR}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{WHLH}	$\overline{WR}$ Rising Edge to ALE Rising Edge	$T_{OSC} - 10$	$T_{OSC} + 15$	ns	(Note 2)
T_{WHBX}	$\overline{BHE}$, INST HOLD after $\overline{WR}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{RHBX}	$\overline{BHE}$, INST HOLD after $\overline{RD}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{WHAX}	AD8–15 hold after $\overline{WR}$ Rising Edge	$T_{OSC} - 50$		ns	
T_{RHAX}	AD8–15 hold after $\overline{RD}$ Rising Edge	$T_{OSC} - 25$		ns	

NOTES:

T_{OSC} = 83.3 ns at 12 MHz; T_{OSC} = 100 ns at 10 MHz.

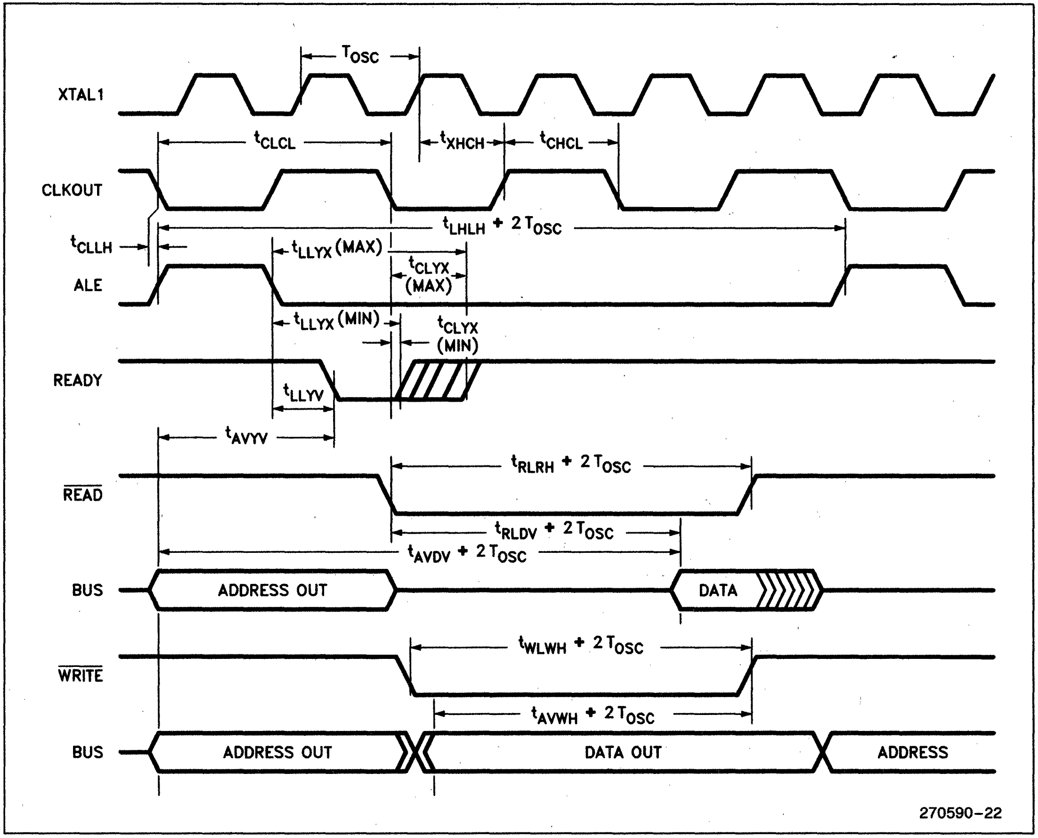
1. Typical specification, not guaranteed.
2. Assuming back-to-back bus cycles.
3. Testing performed at 3.5 MHz, however, the device is static by design and will typically operate below 1 Hz.
4. When using wait states, all $2T_{OSC}$ + n where n = number of wait states.

System Bus Timings



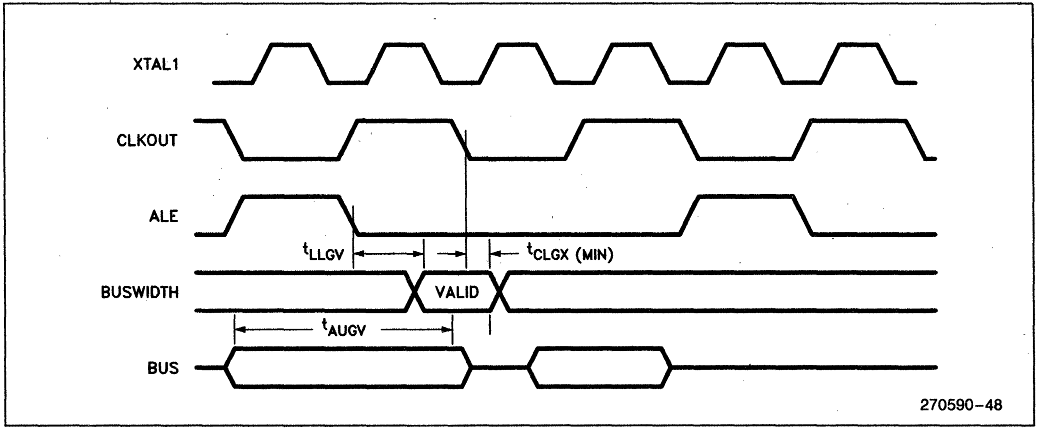
270590-21

READY Timings (One Wait State)



270590-22

Buswidth Bus Timings



270590-48

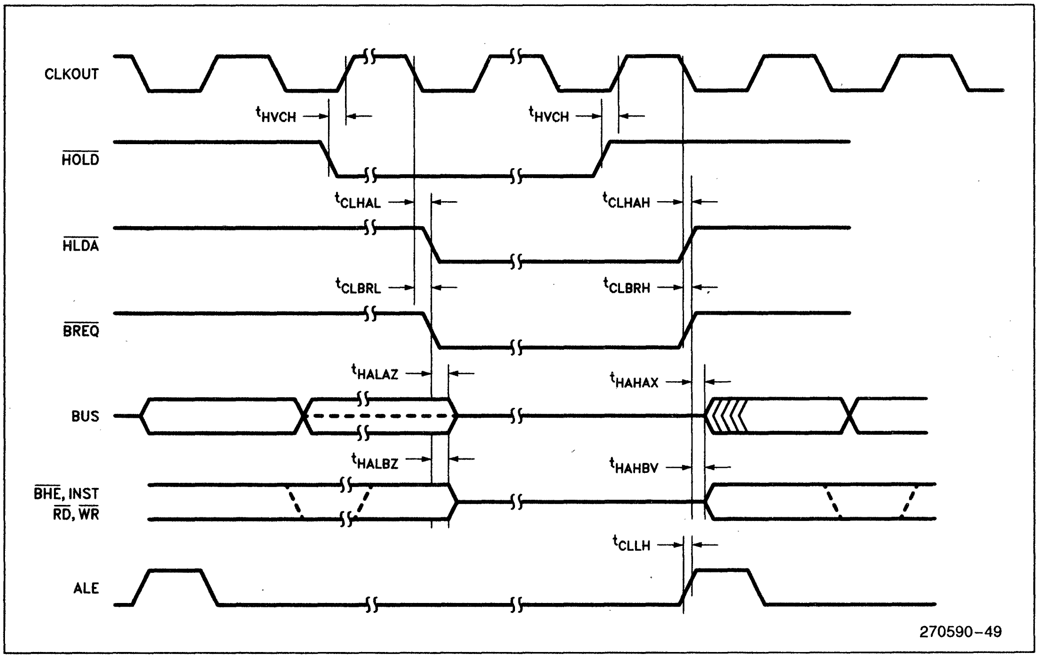
$\overline{\text{HOLD}}/\overline{\text{HLDA}}$ Timings

Symbol	Description	Min	Max	Units	Notes
T_{HVCH}	$\overline{\text{HOLD}}$ Setup	85		ns	(Note 1)
T_{CLHAL}	CLKOUT Low to $\overline{\text{HLDA}}$ Low	−15	15	ns	
T_{CLBRL}	CLKOUT Low to $\overline{\text{BREQ}}$ Low	−15	15	ns	
T_{HALAZ}	$\overline{\text{HLDA}}$ Low to Address Float		20	ns	
T_{HALBZ}	$\overline{\text{HLDA}}$ Low to $\overline{\text{BHE}}$, INST, $\overline{\text{RD}}$, $\overline{\text{WR}}$ Float		25	ns	
T_{CLHAH}	CLKOUT Low to $\overline{\text{HLDA}}$ High	−15	15	ns	
T_{CLBRH}	CLKOUT Low to $\overline{\text{BREQ}}$ High	−15	15	ns	
T_{HAHAX}	$\overline{\text{HLDA}}$ High to Address No Longer Float	−5		ns	
T_{HAHBV}	$\overline{\text{HLDA}}$ High to $\overline{\text{BHE}}$, INST, $\overline{\text{RD}}$, $\overline{\text{WR}}$ Valid	−20		ns	
T_{CLLH}	CLKOUT Low to ALE High	−5	15	ns	

NOTE:
1. To guarantee recognition at next clock.

Maximum Hold Latency

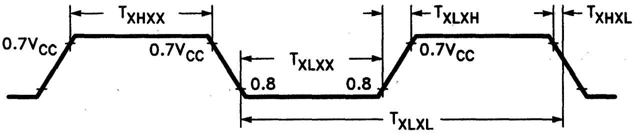
	Max Hold Latency
Internal Access	1.5 States
16-Bit External Execution	2.5 States
8-Bit External	4.5 States



EXTERNAL CLOCK DRIVE

Symbol	Parameter	Min	Max	Units
$1/T_{XLXL}$	Oscillator Frequency 87C196KB10 87C196KB12	 3.5 3.5	 10 12	 MHz MHz
T_{XLXL}	Oscillator Frequency 87C196KB10 87C196KB12	 100 83	 286 286	 ns ns
T_{XHXX}	High Time	32		ns
T_{XLXX}	Low Time	32		ns
T_{XLXH}	Rise Time		10	ns
T_{XHXL}	Fall Time		10	ns

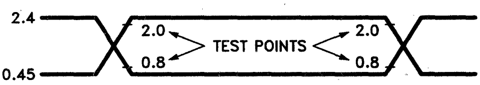
EXTERNAL CLOCK DRIVE WAVEFORMS



270590-23

An external oscillator may encounter as much as a 100 pF load at XTAL1 when it starts-up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{IL} and V_{IH} specifications, the capacitance will not exceed 20 pF.

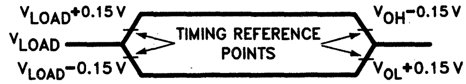
A.C. TESTING INPUT, OUTPUT WAVEFORM



270590-24

A.C. Testing inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0" Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

FLOAT WAVEFORM



270590-25

For Timing Purposes a Port Pin is no Longer Floating when a 100 mV change from Load Voltage Occurs and Begins to Float when a 100 mV change from the Loaded V_{OH}/V_{OL} Level occurs I_{OL}/I_{OH} = ±15 mA.

EXPLANATION OF AC SYMBOLS

Each symbol is two pairs of letters prefixed by "T" for time. The characters in a pair indicate a signal and its condition, respectively. Symbols represent the time between the two signal/condition points.

Conditions:

H	- High
L	- Low
V	- Valid
X	- No Longer Valid
Z	- Floating

Signals:

A	- Address	HA	- $\overline{\text{HLDA}}$
B	- $\overline{\text{BHE}}$	L	- ALE/$\overline{\text{ADV}}$
BR	- $\overline{\text{BREQ}}$	Q	- DATA OUT
C	- CLKOUT	R	- $\overline{\text{RD}}$
D	- DATA IN	W	- $\overline{\text{WR}}/\overline{\text{WRH}}/\overline{\text{WRL}}$
G	- Buswidth	X	- XTAL1
H	- $\overline{\text{HOLD}}$	Y	- READY

EPROM SPECIFICATIONS

A.C. EPROM Programming Characteristics

Operating Conditions: Load Capacitance = 150 pF, T_A = +25°C ±5°C, V_{CC}, V_{REF} = 5V, V_{SS}, ANGND = 0V, V_{PP} = 12.75V ± 0.25V, EA = 12.75V ± 0.25

Symbol	Description	Min	Max	Units
T_{SHLL}	Reset High to First $\overline{\text{PALE}}$ Low	1100		Tosc
T_{LLLH}	$\overline{\text{PALE}}$ Pulse Width	40		Tosc
T_{AVLL}	Address Setup Time	0		Tosc
T_{LLAX}	Address Hold Time	50		Tosc
T_{LLVL}	$\overline{\text{PALE}}$ Low to PVER Low		60	Tosc
T_{PLDV}	$\overline{\text{PROG}}$ Low to Word Dump Valid		50	Tosc
T_{PHDX}	Word Dump Data Hold		50	Tosc
T_{DVPL}	Data Setup Time	0		Tosc
T_{PLDX}	Data Hold Time	50		Tosc
T_{PLPH}	$\overline{\text{PROG}}$ Pulse Width	40		Tosc
T_{PHLL}	$\overline{\text{PROG}}$ High to Next $\overline{\text{PALE}}$ Low	120		Tosc
T_{LHPL}	$\overline{\text{PALE}}$ High to $\overline{\text{PROG}}$ Low	220		Tosc
T_{PHPL}	$\overline{\text{PROG}}$ High to Next $\overline{\text{PROG}}$ Low	120		Tosc
T_{PHIL}	$\overline{\text{PROG}}$ High to AINC Low	0		Tosc
T_{ILIH}	$\overline{\text{AINC}}$ Pulse Width	40		Tosc
T_{ILVH}	PVER Hold after $\overline{\text{AINC}}$ Low	50		Tosc
T_{ILPL}	$\overline{\text{AINC}}$ Low to $\overline{\text{PROG}}$ Low	170		Tosc
T_{PHVL}	$\overline{\text{PROG}}$ High to PVER Low		90	Tosc

NOTES:

1. Run Time Programming is done with Fosc = 6.0 MHz to 12.0 MHz, V_{REF} = 5V ± 0.65V. T_A = +25°C to ±5°C and V_{PP} = 12.75V. For run-time programming over a full operating range, contact the factory.

D.C. EPROM Programming Characteristics

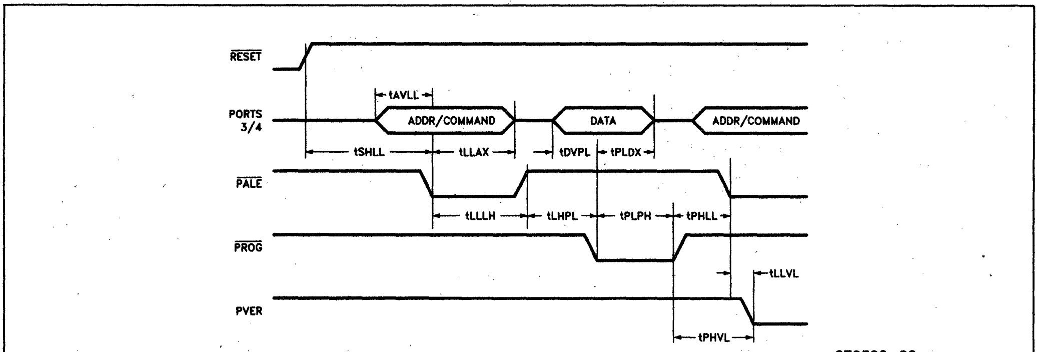
Symbol	Description	Min	Max	Units
I_{PP}	V_{PP} Supply Current (When Programming)		100	mA

NOTE:

V_{PP} must be within 1V of V_{CC} while $V_{CC} < 4.5V$. V_{PP} must not have a low impedance path to ground or V_{SS} while $V_{CC} > 4.5V$.

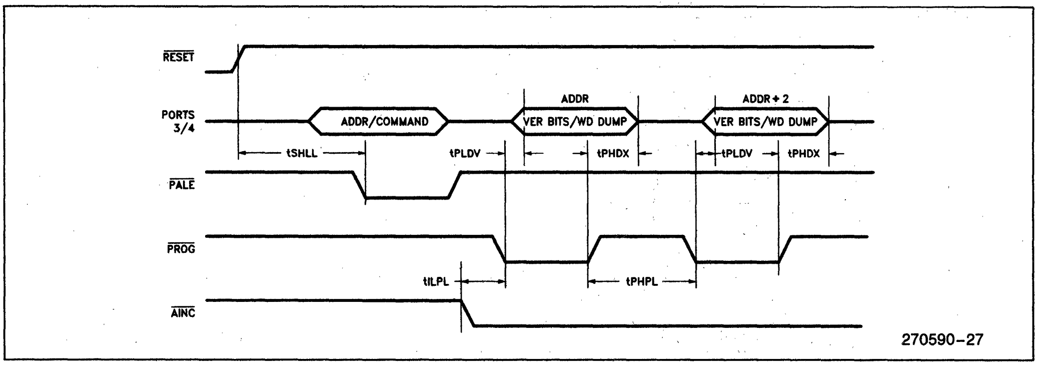
EPROM PROGRAMMING WAVEFORMS

SLAVE PROGRAMMING MODE DATA PROGRAM MODE WITH SINGLE PROGRAM PULSE



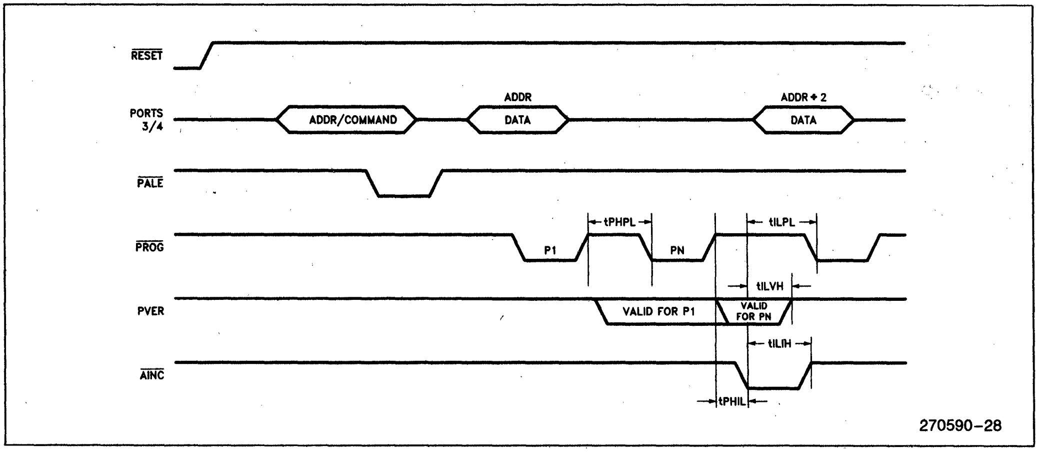
270590-26

SLAVE PROGRAM MODE IN WORD DUMP OR DATA VERIFY MODE WITH AUTO INCREMENT



270590-27

SLAVE PROGRAMMING MODE TIMING IN DATA PROGRAM MODE WITH REPEATED PROG PULSE AND AUTO INCREMENT



270590-28

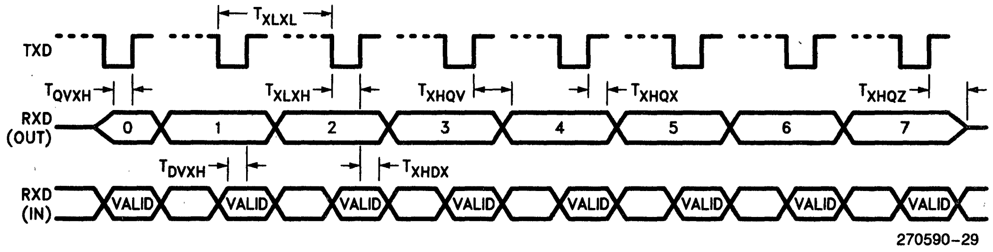
A.C. CHARACTERISTICS—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT TIMING—SHIFT REGISTER MODE

Symbol	Parameter	Min	Max	Units
T_{XLXL}	Serial Port Clock Period (BRR ≥ 8002H)	6 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR ≥ 8002H)	4T_{OSC} ±50		ns
T_{XLXL}	Serial Port Clock Period (BRR = 8001H)	4 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR = 8001H)	2 T_{OSC} ±50		ns
T_{QVXH}	Output Data Setup to Clock Rising Edge	2 T_{OSC} −50		ns
T_{XHQX}	Output Data Hold after Clock Rising Edge	2 T_{OSC} −50		ns
T_{XHQV}	Next Output Data Valid after Clock Rising Edge		2 T_{OSC} +50	ns
T_{DVXH}	Input Data Setup to Clock Rising Edge	T_{OSC} +50		ns
T_{XHDX}	Input Data Hold after Clock Rising Edge	0		ns
T_{XHQZ}	Last Clock Rising to Output Float		1 T_{OSC}	ns

WAVEFORM—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT WAVEFORM—SHIFT REGISTER MODE



A TO D CHARACTERISTICS

There are two modes of A/D operation: with or without clock prescaler. The modes are shown in the table below. The maximum frequency with the clock prescaler off is 8 MHz.

The converter is ratiometric, so the absolute accuracy is directly dependent on the accuracy and stability of V_{REF}. V_{REF} must be close to V_{CC} since it supplies both the resistor ladder and the digital section of the converter.

A/D CONVERTER SPECIFICATIONS

The specifications given below assume adherence to the Operating Conditions section of this data sheet. Testing is performed with $V_{REF} = 5.12V$.

Clock Prescaler On **IOC2.4 = 0**	**Clock Prescaler Off** **IOC2.4 = 1**
158 States 26.33 μs @ 12 MHz	91 States 22.75 μs @ 8 MHz

Parameter	Typical(1)	Minimum	Maximum	Units*	Notes
Resolution		512 9	1024 10	Levels Bits	
Absolute Error		0	±4	LSBs	
Full Scale Error				LSBs	
Zero Offset Error				LSBs	
Non-Linearity Error		0	±4	LSBs	
Differential Non-Linearity Error		> −1	+2	LSBs	
Channel-to-Channel Matching		0	±1	LSBs	
Repeatability	±0.25			LSBs	
Temperature Coefficients: Offset Full Scale Differential Non-Linearity	 0.009 0.009 0.009			 LSB/°C LSB/°C LSB/°C	
Off Isolation		−60		dB	2, 3
Feedthrough	−60			dB	2
V_{CC} Power Supply Rejection	−60			dB	2
Input Resistance		1K	5K	Ω	
D.C. Input Leakage		0	3.0	μA	
Sample Time: Prescaler On Prescaler Off	 15 8			 States States	 4 4
Sample Capacitive	3			pF	

NOTES:
*An "LSB", as used here, has a value of approximately 5 mV.
1. These values are expected for most parts at 25°C but are not tested or guaranteed.
2. DC to 100 KHz.
3. Multiplexer Break-Before-Make Guaranteed.
4. One state = 167 ns at 12 MHz, 250 ns at 8 MHz.

A/D GLOSSARY OF TERMS

ABSOLUTE ERROR—The maximum difference between corresponding actual and ideal code transitions. Absolute Error accounts for all deviations of an actual converter from an ideal converter.

ACTUAL CHARACTERISTIC—The characteristic of an actual converter. The characteristic of a given converter may vary over temperature, supply voltage, and frequency conditions. An actual characteristic rarely has ideal first and last transition locations or ideal code widths. It may even vary over multiple conversions under the same conditions.

BREAK BEFORE MAKE—The property of a multiplexer which guarantees that a previously selected channel is selected (i.e., the converter will not short inputs together).

CHANNEL-TO-CHANNEL MATCHING—The difference between corresponding code transitions of actual characteristics taken from different channels under the same temperature, voltage and frequency conditions.

CHARACTERISTIC—A graph of input voltage versus the resultant output code for an A/D converter. It describes the transfer function of the A/D converter.

CODE—The digital value output by the converter.

CODE TRANSITION—The point at which the converter changes from an output code of Q, to a code of Q + 1. The input voltage corresponding to a code transition is defined to be that voltage which is equally likely to produce either of two adjacent codes.

CODE WIDTH—The voltage corresponding to the difference between two adjacent code transitions.

D.C. INPUT LEAKAGE—Leakage current to ground from an analog input pin.

DIFFERENTIAL NON-LINEARITY—The difference between the ideal and actual code widths of the terminal based characteristic.

FEEDTHROUGH—Attenuation of a voltage applied on the selected channel of the A/D Converter after the sample window closes.

FULL SCALE ERROR—The difference between the expected and actual input voltage corresponding to the full scale code transition.

IDEAL CHARACTERISTIC—A characteristic with its first code transition at V_{IN} = 0.5 LSB, its last code transition at V_{IN} = (V_{REF} − 1.5 LSB) and all code widths equal to one LSB.

INPUT RESISTANCE—The effective series resistance from the analog input pin to the sample capacitor.

LSB—Least Significant Bit: The voltage corresponding to the full scale voltage divided by 2^n, where n is the number of bits of resolution of the converter. For an 8-bit converter with a reference voltage of 5.12V, one LSB is 20 mV. Note that this is different than digital LSBs, since an uncertainty of two LSB, when referring to an A/D converter, equals 40 mV. (This has been confused with an uncertainty of two digital bits, which would mean four counts, or 80 mV.)

NON-LINEARITY—The maximum deviation of code transitions of the terminal based characteristic from the corresponding code transitions of the ideal characteristic.

OFF-ISOLATION—Attenuation of a voltage applied on a deselected channel of the A/D converter. (Also referred to as Crosstalk.)

REPEATABILITY—The difference between corresponding code transitions from different actual characteristics taken from the same converter on the same channel at the same temperature, voltage and frequency conditions.

RESOLUTION—The number of input voltage levels that the converter can unambiguously distinguish between. Also defines the number of useful bits of information which the converter can return.

SAMPLE TIME—The time that the sample window is open.

SAMPLE WINDOW—Begins when the sample capacitor is attached to a selected channel and ends when the sample capacitor is disconnected from the selected channel.

TEMPERATURE COEFFICIENTS—Change in the stated variable per degree centigrade temperature change. Temperature coefficients are added to the typical values of a specification to see the effect of temperature drift.

TERMINAL BASED CHARACTERISTIC—An actual characteristic which has been rotated and translated to remove zero offset and full scale error.

V_{CC} REJECTION—Attenuation of noise on the V_{CC} line to the A/D converter.

ZERO OFFSET—The difference between the expected and actual input voltage corresponding to the first code transition.

87C196KB FUNCTIONAL DEVIATIONS

The 87C196KB has the following errata.

1. Interrupts will not occur between an untaken conditional jump and the next instruction. A series of untaken jumps will hold off interrupts until after the last untaken jump.
2. The DJNZW instruction is not guaranteed to be functional. The instruction, if encountered, will not cause an unimplemented opcode. The DJNZ (byte instruction) works correctly and should be used instead.
3. The serial port only tolerates a +1.25%, -7.5% baud rate error between transmitter and receiver. If the serial port fails on the receiver, increase the baud rate.
4. The CDE function is not guaranteed to work. In order to ensure proper 87C196KB operation, the pin must be directly connected to V_{SS}.
5. The HSI unit has two errata: one dealing with resolution and the other with first entries into the FIFO.

   The HSI resolution is 9 states instead of 8 states. Events on the same line may be lost if they occur faster than once every 9 state times.

   There is a mismatch between the 9 state time HSI resolution and the 8 state time timer. This causes one time value to be unused every 9 timer counts. Events may receive a time-tag one count later than expected because of this "skipped" time value.

   If the first two events into an empty FIFO (not including the Holding Register) occur in the same internal phase, both are recorded with one time-tag. Otherwise, if the second event occurs within 9 states after the first, its time-tag is one count later than the first's. If this is the "skipped" time value, the second event's time-tag is 2 counts later than the first's.

   If the FIFO and Holding Register are empty, the first event will transfer into the Holding Register after 8 state times, leaving the FIFO empty again. If the second event occurs after this time, it will act as a new first event into an empty FIFO.
6. The serial port Framing Error flag fails to indicate an error if the bit preceding the stop bit is a 1. This is the case in both the 8-bit and 9-bit modes. False framing errors are never generated.

DIFFERENCES BETWEEN THE 80C196KA AND THE 80C196KB

The 8XC196KB is identical to 8XC196KA except for the following differences.

1. ALE is high after reset on the 87C196KB instead of low like the 80C196KA.
2. The DJNZW instruction is not guaranteed to work on the 87C196KB.
3. The $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ bus protocol is available on the 87C196KB.

CONVERTING FROM OTHER 8096BH FAMILY PRODUCTS TO THE 8XC196KB

The following list of suggestions for designing an 879XBH system will yield a design that is easily converted to the 87C196KB.

1. Do not base critical timing loops on instruction or peripheral execution times.
2. Use equate statements to set all timing parameters, including the baud rate.
3. Do not base hardware timings on CLKOUT or XTAL1. The timings of the 87C196KB are different than those of the 8X9XBH, but they will function with standard ROM/EPROM/Peripheral type memory systems.
4. Verify that all inputs are driven high or low and not left floating.
5. Indexed and indirect operations relative to the stack pointer (SP) work differently on the 80C196KB than on the 8096BH. On the 8096BH, the address is calculated based on the un-updated version of the stack pointer. The 87C196KB uses the updated version. The offset for POP[SP] and POP nn[SP] instructions may need to be changed by a count of 2.
6. The 87C196KB does not support gang programming in auto programming mode. Gand programming in slave programming mode is supported.
7. $\overline{\text{PACT}}$ has changed from the HSO.O on the 8796BH to P2.7 on the 87C196KB.
8. The V_{PD} on the 8096BH has changed to a V_{SS} pin on the 87C196KB.

DATA SHEET REVISION HISTORY

This data sheet (version -003) is valid for devices with a "B" suffix on the topside tracking number.

The following differences exist between this and the -002 version of the 87C196KB data sheet.

1. The Instruction Summary Table, Execution State Times Table, Baud Rate Calculations, HOLD/HLDA Section, and EPROM Programming Section were deleted. This information is available in the 80C196KB User's Guide.
2. The CDE (Clock Detect Enable) pin was changed to a V_{SS} pin.
3. The PLCC package was added.
4. Thermal Characteristics for the PLCC and LCC packages were added.
5. The I_{IL1} max. specification was changed from −950 μA to −1.2 mA.
6. The following A.C. Timings were added: TRLRH Max, TWLWH Max, TRHBX, TWHAX, TRHAX.
7. The following A.C. Timings were changed:

Symbol	Description	87C196KB (Rev -002)	87C196KB (Rev -003)
T_{AVYV}	Address Valid to READY Setup 87C196KB10	$2T_{OSC} - 85$	$2T_{OSC} - 90$
T_{AVLL}	Address Valid to ALE Falling Edge	$T_{OSC} - 15$	$T_{OSC} - 20$
T_{RLCL}	$\overline{RD}$ Low to CLKOUT Falling Edge	Min = 10	Min = 5

8. The figures for System Bus Timings were redrawn to include the new timings and to more accurately describe the READY and BUSWIDTH timings.
9. $\overline{HOLD}/\overline{HLDA}$ timings were added.
10. The following A/D Converter Specifications were changed:

Parameter	87C196KB (Rev -002)	87C196KB (Rev -003)
Absolute Error	Max = ±8 LSBs	Max = ±4 LSBs
Full Scale Error	Typ = 0/−2 LSBs	0.25 ± 0.50
Zero Offset Error	Typ = −0.5/2.0 LSBs	-0.25 ± 0.50
Non-Linearity Error	Max = ±8 LSBs	Max = ±4 LSBs

Differential Non-Linearity Error	Min = 0 LSBs
	Max = ±2 LSBs
		Min = > −1 LSBs
		Max = +2 LSBs

D.C. Input Leakage	Max = ±3 μA	Min = 0 μA
		Max = +3 μA

11. Errata was added for the HSI resolution and first event anomalies.
12. Errata was added for the serial port Framing Error anomaly.

The following differences exist between the -002 Preliminary version and the -001 Advanced version of the 87C196KB data sheet.

1. The I_{IL1} Logical 0 input current in Reset changed from −500 μA to −850 μA.
2. New current specifications have been added to reflect typical current consumption at room temperature.
3. Current curves for I_{IDLE} (Max) and I_{IDLE} (Typical) have been added.
4. The following AC Characteristics have changed:

Symbol	Description	87C196KB (Rev -001)	87C196KB (Rev -002)
T_{AVYV}	Address Valid to READY Setup	$2T_{OSC} - 70$	$2T_{OSC} - 85$
T_{LLYV}	ALE Low to READY Setup 87C196KB10 87C196KB12	$T_{OSC} - 75$	$T_{OSC} - 80$ $T_{OSC} - 72$
T_{AVGV}	Address Valid to Buswidth Setup	$2T_{OSC} - 70$	$2T_{OSC} - 85$
T_{AVDV}	Address Valid to Input Data Valid 87C196KB10 87C196KB12	$3T_{OSC} - 60$	$3T_{OSC} - 70$ $3T_{OSC} - 67$
T_{RLAZ}	RD Low to Address Float	TBD	10
T_{QVWH}	Data Stable to WR Rising Edge 87C196KB10 87C196KB12	$T_{OSC} - 20$	$T_{OSC} - 30$ $T_{OSC} - 23$
T_{WHBX}	BHE, INST Hold after WR Rising Edge	$T_{OSC} - 5$	$T_{OSC} - 10$

5. The A/D no Sample and Hold feature is no longer available.
6. With the clock prescaler disabled, the A/D has a larger absolute error for frequencies greater than 8 MHz. The clock prescaler should be enabled for frequencies greater than 8 MHz.
7. A/D Sample Time and Sample Capacitance specifications have been added.
8. The following A/D Characteristics have changed:

Parameter	87C196KB (Rev -001)	87C196KB (Rev -002)
Absolute Error	±4 LSBs	±8 LSBs
Full Scale Error	−0.5/±0.5 LSBs	0/−2 LSBs
Zero Offset Error	±0.5 LSBs	−0.5/−2 LSBs
Non-Linearity Error	±4 LSBs	±8 LSBs

9. The minimum instruction times for some of the indirect instructions which write to external memory have changed.
10. Serial Port Timings have been added.
11. The upper four bits of the A/D Command register should be set to zero for future compatibility.
12. HSO Commands 0C and 0D should not be used to ensure future compatibility.
13. The Modified Quick Pulse™ algorithm has been recommended to ensure programming of the EPROM over the life of the device for both run-time and slave programming modes.

ADVANCE INFORMATION

MCS®-96 87C196KB/83C196KB/80C196KB *Express*

- **Extended Temperature Range (−40°C to +85°C)**
- **Burn-In**

The Intel EXPRESS system offers enhancements to the operational specifications of the MCS®-96 family of microcontrollers. These EXPRESS products are designed to meet the needs of those applications whose operating requirements exceed commercial standards.

The EXPRESS program includes an extended temperature range with or without burn-in, depending on the package type.

With the commercial standard temperature range operational characteristics are guaranteed over the temperature range of 0°C to +70°C. With the extended temperature range option, operational characteristics are guaranteed over the range of −40°C to +85°C.

The optional burn-in is dynamic, for a minimum time of 160 hours at 125°C with V_{CC} = 5.5V ±0.5V, following guidelines in MIL-STD-883, Method 1015.

Package types and EXPRESS versions are identified by a one- or two-letter prefix to the part number. The prefixes are listed in Table 1.

This data sheet specifies the parameters for the extended temperature range option. The commercial temperature range data sheets including functional deviations are applicable otherwise.

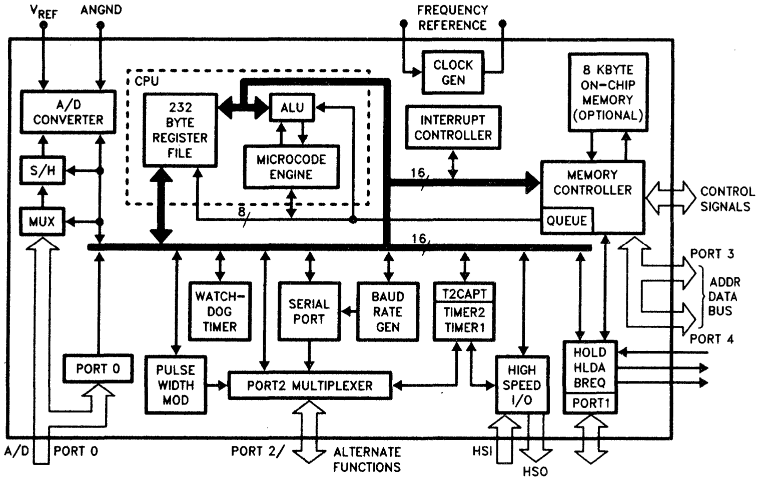


Figure 1. 8XC196KB Block Diagram

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied. Information contained herein supersedes previously published specifications on these devices from Intel.

August 1989
Order Number: 270780-002

Table 1. Express Prefix Identification

Product	Prefix	Package	Temperature	Burn-in
80C196KB12/80C196TB12	LA TN	PGA PLCC	Extended Extended	Yes No
83C196KB12/83C196TB12	TN	PLCC	Extended	No
87C196KB10	LR	LCC	Extended	Yes

Table 2. Thermal Characteristics

Package Type	θja	θjc
PGA	28°C/W	3.5°C/W
PLCC	35°C/W	12°C/W
LCC	28°C/W	3.5°C/W

EXPRESS PACKAGING

The 80C196KB/TB and 83C196KB/TB are available in a 68-pin PLCC package. In addition, the 80C196KB/TB is available in a 68-pin PGA package. The 87C196KB is only available in a 68-pin LCC package.

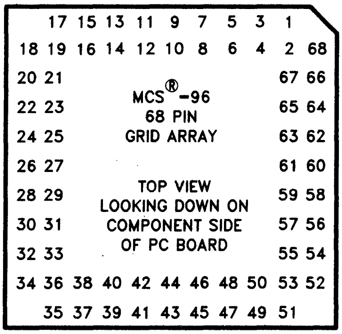
PGA/ LCC	PLCC	Description	PGA/ LCC	PLCC	Description	PGA/ LCC	PLCC	Description
1	9	ACH7/P0.7	24	54	AD6/P3.6	47	31	P1.6/$\overline{\text{HLDA}}$
2	8	ACH6/P0.6	25	53	AD7/P3.7	48	30	P1.5/$\overline{\text{BREQ}}$
3	7	ACH2/P0.2	26	52	AD8/P4.0	49	29	HSO.1
4	6	ACH0/P0.0	27	51	AD9/P4.1	50	28	HSO.0
5	5	ACH1/P0.1	28	50	AD10.P4.2	51	27	HSO.5/HSI.3
6	4	ACH3/P0.3	29	49	AD11/P4.3	52	26	HSO.4/HSI.2
7	3	NMI	30	48	AD12/P4.4	53	25	HSI.1
8	2	$\overline{\text{EA}}$	31	47	AD13/P4.5	54	24	HSI.0
9	1	V_{CC}	32	46	AD14/P4.6	55	23	P1.4
10	68	V_{SS}	33	45	AD15/P4.7	56	22	P1.3
11	67	XTAL1	34	44	T2CLK/P2.3	57	21	P1.2
12	66	XTAL2	35	43	READY	58	20	P1.1
13	65	CLKOUT	36	42	T2RST/P2.4/$\overline{\text{AINC}}$	59	19	P1.0
14	64	BUSWIDTH	37	41	BHE/WRH	60	18	TXD/P2.0
15	63	INST	38	40	WR/WRL	61	17	RXD/P2.1
16	62	ALE/$\overline{\text{ADV}}$	39	39	PWM/P2.5	62	16	$\overline{\text{RESET}}$
17	61	$\overline{\text{RD}}$	40	38	P2.7/T2CAPTURE/$\overline{\text{PACT}}$	63	15	EXTINT/P2.2
18	60	AD0/P3.0	41	37	V_{PP}	64	14	V_{SS}[1]
19	59	AD1/P3.1	42	36	V_{SS}	65	13	V_{REF}
20	58	AD2/P3.2	43	35	HSO.3/SID3	66	12	ANGND
21	57	AD3/P3.3	44	34	HSO.2/SID2	67	11	ACH4/P0.4
22	56	AD4/P3.4	45	33	P2.6/T2UP-DN	68	10	ACH5/P0.5
23	55	AD5/P3.5	46	32	P1.7/$\overline{\text{HOLD}}$			

Figure 2. Pin Definitions

NOTE:

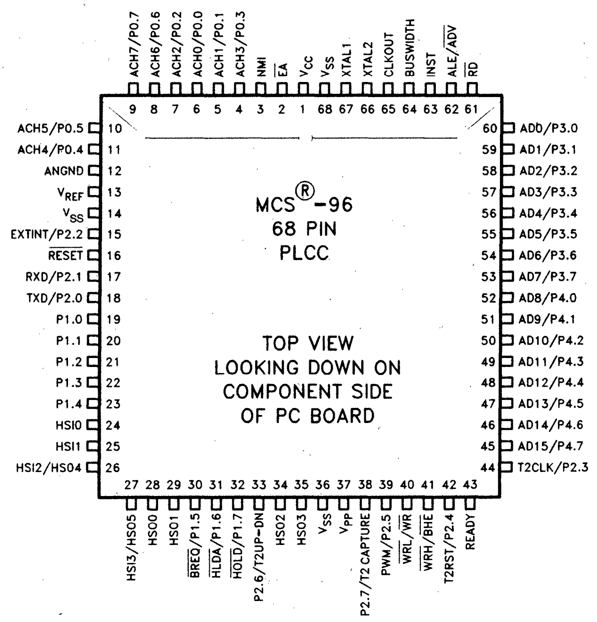
1. This pin was formerly the Clock Detect Enable Pin. The CDE function is not guaranteed to work, therefore this pin must be directly connected to V_{SS}.

Pins Facing Down



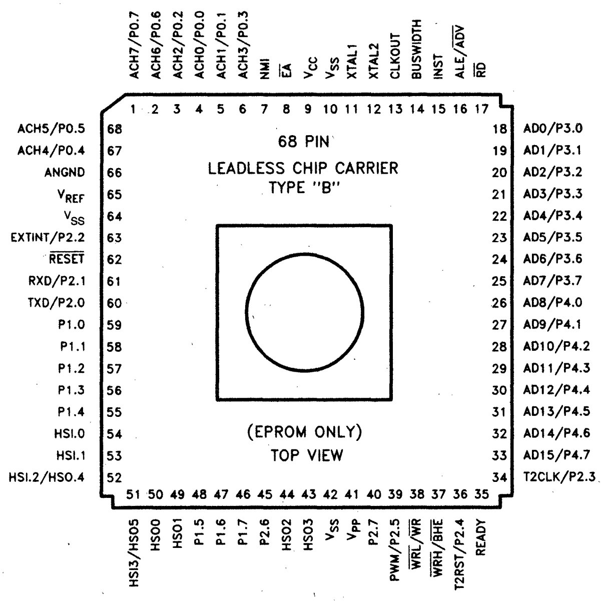
270634–23

Figure 3. 68-Pin Package (Pin Grid Array — Top View) 80C196KB Only



270634-2

Figure 4. 68-Pin Package (PLCC–Top View) 83C196KB/80C196KB



270680-2

(Top view looking down on component side of PC boards)

Figure 5. 68-Pin Package (LCC–Top View) 87C196KB Only

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings*

Ambient Temperature Under Bias	−40°C to +85°C
Storage Temperature	−65°C to +150°C
Voltage On Any Pin to V_{SS}	−0.5V to +7.0V
Power Dissipation	1.5W

**Notice: Stresses 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 specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

NOTICE: Specifications contained within the following tables are subject to change.

Operating Conditions

Symbol	Description	Min	Max	Units
T_A	Ambient Temperature Under Bias	−40	+85	°C
V_{CC}	Digital Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	3.5	12	MHz

NOTE:
ANGND and V_{SS} should be nominally at the same potential.

This is an Advance Data Sheet. It is expected that parameters may change before Intel releases this product for sale. Contact your local sales office before finalizing the Timing and D.C. Characteristics section of a design to verify you have the latest information.

D.C. Characteristics (Over specified operating conditions)

Symbol	Description	Min	Max	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage (Note 1)	$0.2\ V_{CC} + 0.9$	$V_{CC} + 0.5$	V	
V_{IH1}	Input High Voltage on XTAL 1	$0.7\ V_{CC}$	$V_{CC} + 0.5$	V	
V_{IH2}	Input High Voltage on RESET	2.4	$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage		0.3 0.45 1.5	V V V	$I_{OL} = 200\ \mu A$ $I_{OL} = 3.2\ mA$ $I_{OL} = 7\ mA$
V_{OH}	Output High Voltage (Standard Outputs)	$V_{CC} - 0.3$ $V_{CC} - 0.7$ $V_{CC} - 1.5$		V V V	$I_{OH} = -200\ \mu A$ $I_{OH} = -3.2\ mA$ $I_{OH} = -7\ mA$
V_{OH1}	Output High Voltage (Quasi-bidirectional Outputs)	$V_{CC} - 0.3$ $V_{CC} - 0.7$ $V_{CC} - 1.5$		V V V	$I_{OH} = -7\ \mu A$ $I_{OH} = -30\ \mu A$ $I_{OH} = -60\ \mu A$
I_{LI}	Input Leakage Current (Std. Inputs)		±10	μA	$0 < V_{IN} < V_{CC} - 0.3V$
I_{LI1}	Input Leakage Current (Port 0)		±3	μA	$0 < V_{IN} < V_{REF}$
I_{TL}	1 to 0 Transition Current (QBD Pins)		−650	μA	$V_{IN} = 2.0V$
I_{IL}	Logical 0 Input Current (QBD Pins)		−50	μA	$V_{IN} = 0.45V$
I_{IL1}	Logical 0 Input Current in Reset (Note 2) (ALE, $\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, INST, P2.0)		−1.2	mA	$V_{IN} = 0.45\ V$

NOTE:
1. All pins except RESET and XTAL1.
2. Holding these pins below V_{IH} in Reset may cause the part to enter test modes.

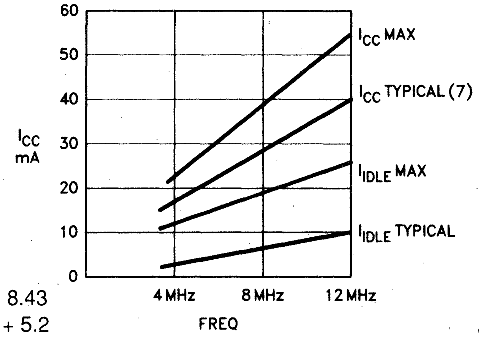
D.C. Characteristics (Over specified operating conditions) (Continued)

Symbol	Description	Min	Typ(7)	Max	Units	Test Conditions
I_{CC}	Active Mode Current in Reset		40	55	mA	XTAL1 = 12 MHz $V_{CC} = V_{PP} = V_{REF} = 5.5V$
I_{REF}	A/D Converter Reference Current		2	5	mA	
I_{IDLE}	Idle Mode Current		10	25	mA	
I_{CC1}	Active Mode Current in Reset		15	22	mA	XTAL1 = 3.5 MHz
I_{PD}(8)	Powerdown Mode Current		5		μA	$V_{CC} = V_{PP} = V_{REF} = 5.5V$
R_{RST}	Reset Pullup Resistor	6K		100K	Ω	
C_S	Pin Capacitance (Any Pin to V_{SS})			10	pF	f_{TEST} = 1.0 MHz

NOTES:
(Notes apply to all specifications)
1. QBD (Quasi-bidirectional) pins include Port 1, P2.6 and P2.7.
2. Standard Outputs include AD0-15, $\overline{RD}$, $\overline{WR}$, ALE, $\overline{BHE}$, INST, HSO pins, PWM/P2.5, CLKOUT, RESET, Ports 3 and 4, TXD/P2.0, and RXD (in serial mode 0). The V_{OH} specification is not valid for RESET. Ports 3 and 4 are open-drain outputs.
3. Standard Inputs include HSI pins, CDE, $\overline{EA}$, READY, BUSWIDTH, NMI, RXD/P2.1, EXTINT/P2.2, T2CLK/P2.3, and T2RST/P2.4.
4. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V or V_{OH} is held below V_{CC} − 0.7V:
   I_{OL} on Output pins: 10 mA
   I_{OH} on quasi-bidirectional pins: self limiting
   I_{OH} on Standard Output pins: 10 mA
5. Maximum current per bus pin (data and control) during normal operation is ±3.2 mA.
6. During normal (non-transient) conditions the following total current limits apply:

Port 1, P2.6	I_{OL}: 29 mA	I_{OH} is self limiting
HSO, P2.0, RXD, $\overline{RESET}$	I_{OL}: 29 mA	I_{OH}: 26 mA
P2.5, P2.7, $\overline{WR}$, $\overline{BHE}$	I_{OL}: 13 mA	I_{OH}: 11 mA
AD0-AD15	I_{OL}: 52 mA	I_{OH}: 52 mA
$\overline{RD}$, ALE, INST-CLKOUT	I_{OL}: 13 mA	I_{OH}: 13 mA

7. Typicals are based on a limited number of samples and are not guaranteed. The values listed are at room temperature and $V_{REF} = V_{CC} = 5V$.
8. I_{PD} is not guaranteed on the standard 80C196KB part and may exceed 100 μA on some parts. Customers whose applications use the powerdown mode and require a guaranteed maximum value of I_{PD} should contact an Intel Field Sales Representative.



I_{CC} Max = 3.88 X FREQ + 8.43
I_{IDLE} Max = 1.65 X FREQ + 5.2

270634-15

Figure 6. I_{CC} and I_{IDLE} vs Frequency

A.C. CHARACTERISTICS

For use over specified operating conditions
Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, fosc = 12 Mhz

The system must meet these specifications:

Symbol	Description	Min	Max	Notes
T_{AVYV}	Address valid to READY setup 80C196KB12/83C196KB12 87C196KB10		 2Tosc–85 2Tosc–105	
T_{LLYV}	ALE low to READY setup 80C196KB12/83C196KB12 87C196KB10		 Tosc–65 Tosc–95	
T_{YLYH}	Non READY time	No Upper Limit		
T_{CLYX}	READY hold after CLKOUT low	0	Tosc–30	Note 1
T_{LLYX}	READY hold after ALE low	Tosc–15	2Tosc–40	Note 1
T_{AVGV}	Address valid to BUSWIDTH setup 80C196KB12/83C196KB12 87C196KB10		 2Tosc–85 2Tosc–95	
T_{LLGV}	ALE low to BUSWIDTH setup 80C196KB12/83C196KB12 87C196KB10		 Tosc–60 Tosc–85	
T_{CLGX}	BUSWIDTH hold after CLKOUT low	0		
T_{AVDV}	Address valid to input data valid 80C196KB12/83C196KB12 87C196KB10		 3Tosc–60 3Tosc–80	 Note 2
T_{RLDV}	$\overline{RD}$ low to input data valid 80C196KB12/83C196KB12 87C196KB10		 Tosc–25 Tosc–30	 Note 2
T_{CLDV}	CLKOUT low to input data valid 80C196KB12/83C196KB12 87C196KB10		 Tosc–50 Tosc–60	
T_{RHDZ}	$\overline{RD}$ high to input data float		Tosc–20	
T_{RXDX}	Data hold after $\overline{RD}$ inactive	0		

NOTE:

1. If max is exceeded, additional wait states will occur.
2. When using wait states, add 2Tosc x n where n = number of wait states.

A.C. CHARACTERISTICS

For ·ıse over specified operating conditions
Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, fosc = 12 Mhz

The 8XC196KB will meet these specifications:

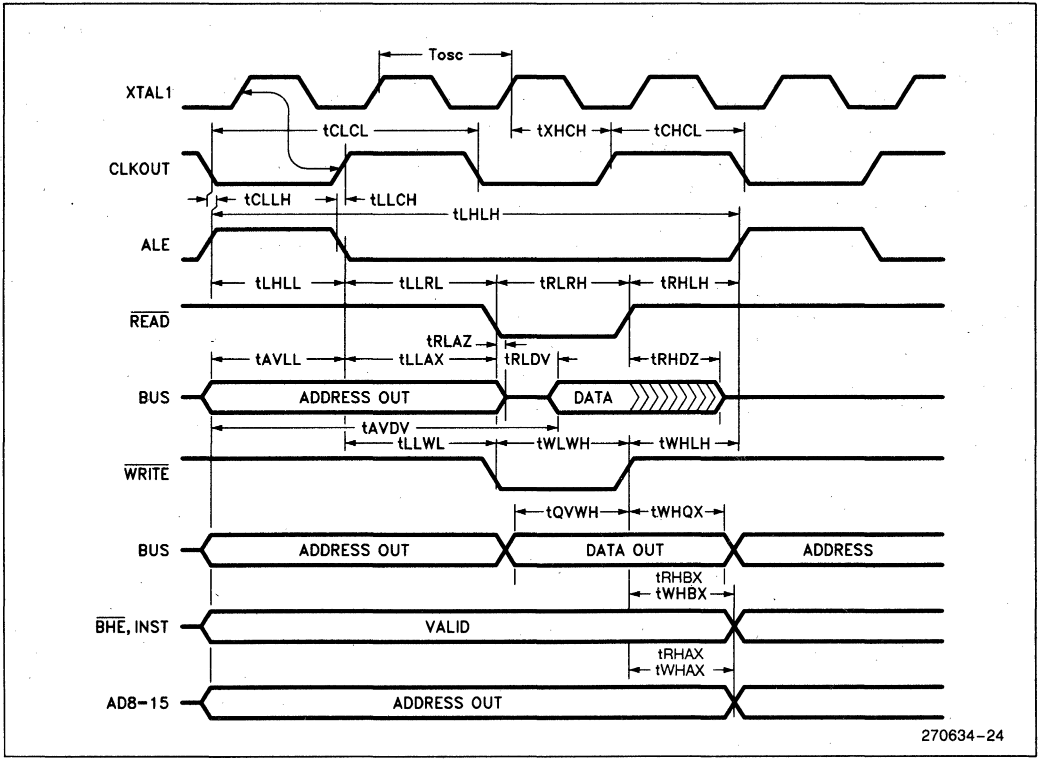
Symbol	Description	Min	Max	Notes
F_{XTAL}	Frequency on XTAL1 80C196KB12/83C196KB12 87C196KB10	 3.5 3.5	 12 10	 Note 1 Note 1
T_{OSC}	$1/F_{XTAL}$ 80C196KB12/83C196KB12 87C196KB10	 83 100	 286 286	
T_{XHCH}	XTAL1 high to CLKOUT high or low 80C196KB12/83C196KB12 87C196KB10	 35 40	 110 130	Note 2
T_{CLCL}	CLKOUT cycle time	2Tosc		
T_{CHCL}	CLKOUT high time	Tosc –10	Tosc+10	
T_{CLLH}	CLKOUT falling edge to ALE rising	– 5	15	
T_{LLCH}	ALE falling edge to CLKOUT rising	–15	15	
T_{LHLH}	ALE cycle time	4Tosc		Note 4
T_{LHLL}	ALE high period	Tosc –12	Tosc+12	
T_{AVLL}	Address setup to ALE falling	Tosc – 20		
T_{LLAX}	Address hold after ALE falling 80C196KB12/83C196KB12 87C196KB10	 Tosc – 40 Tosc – 45		
T_{LLRL}	ALE falling edge to $\overline{RD}$ falling 80C196KB12/83C196KB12 87C196KB10	 Tosc – 40 Tosc – 45		
T_{RLCL}	$\overline{RD}$ low to CLKOUT falling	5	30	
T_{RLRH}	$\overline{RD}$ low period	Tosc – 5	Tosc+25	Note 4
T_{RHLH}	$\overline{RD}$ rising edge to ALE rising 80C196KB12/83C196KB12 87C196KB10	 Tosc Tosc	 Tosc+25 Tosc+30	Note 3
T_{RLAZ}	$\overline{RD}$ low to address float		10	
T_{LLWL}	ALE falling edge to $\overline{WR}$ falling	Tosc –10		
T_{CLWL}	CLKOUTlow to $\overline{WR}$ falling 80C196KB12/83C196KB12 87C196KB10	 0 0	 25 30	
T_{QVWH}	Data stable to $\overline{WR}$ rising 80C196KB12/83C196KB12 87C196KB10	 Tosc – 23 Tosc – 30		Note 4
T_{CHWH}	CLKOUT high to $\overline{WR}$ rising	–10	+10	
T_{WLWH}	$\overline{WR}$ low period	Tosc – 30	Tosc+5	Note 4

A.C. CHARACTERISTICS

Symbol	Description	Min	Max	Notes
T_{WHQX}	Data hold after $\overline{WR}$ rising	Tosc–10		
T_{WHLH}	$\overline{WR}$ rising edge to ALE rising	Tosc–10	Tosc+15	Note 3
T_{WHBX}	$\overline{BHE}$, INST hold after $\overline{WR}$ rising	Tosc–10		
T_{WHAX}	AD8–15 hold after $\overline{WR}$ rising	Tosc–50		
T_{RHBX}	$\overline{BHE}$, INST hold after $\overline{RD}$ rising	Tosc–10		
T_{RHAX}	AD8–15 hold after $\overline{RD}$ rising	Tosc–50		

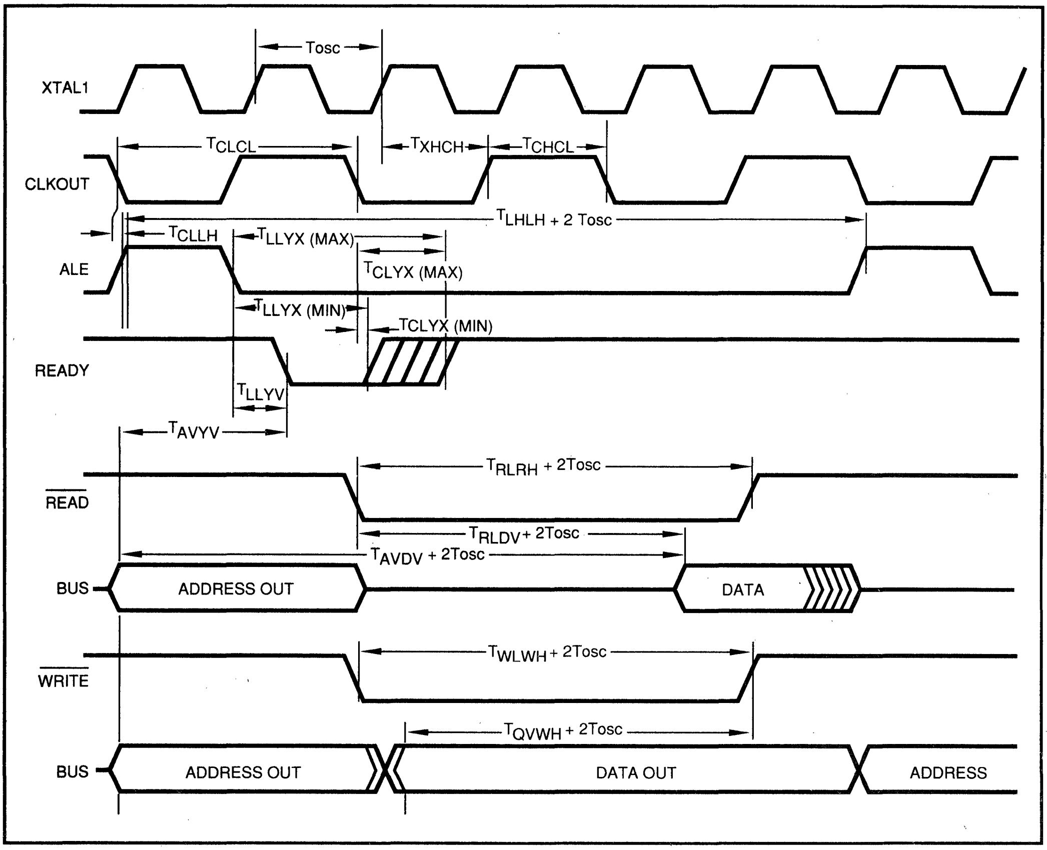
NOTES:
1. Testing performed at 3.5 MHz. However, the device is static by design and will typically operate below 1 Hz.
2. Typical specification, not guaranteed.
3. Assuming back-to-back bus cycles.
4. When using wait states, add 2 Tosc x n where n = number of wait states.

System Bus Timings

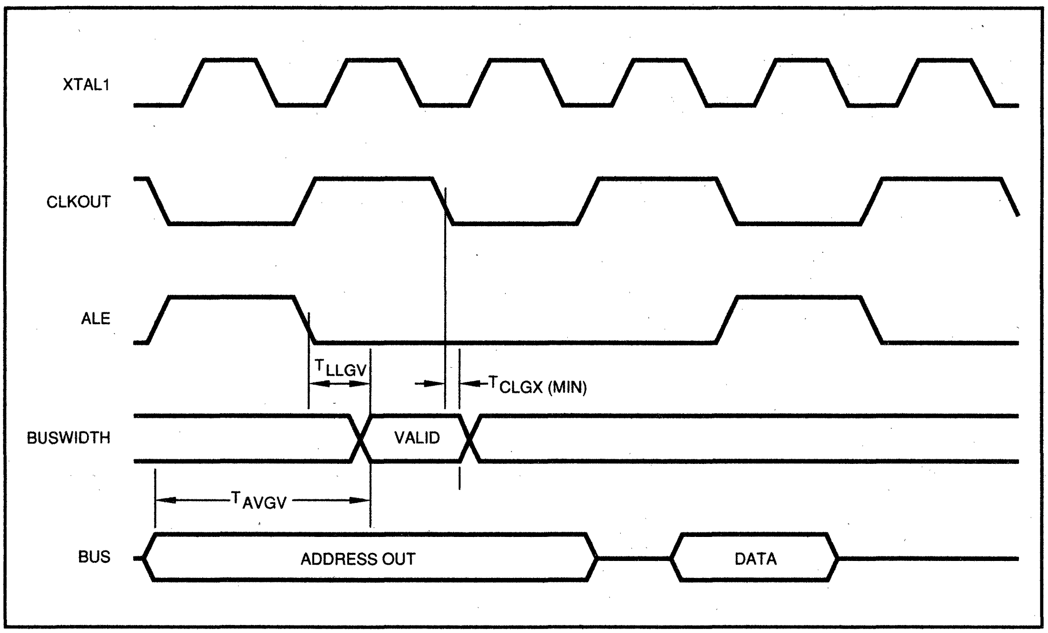


270634-24

READY Timings (One Waitstate)



Buswidth Timings

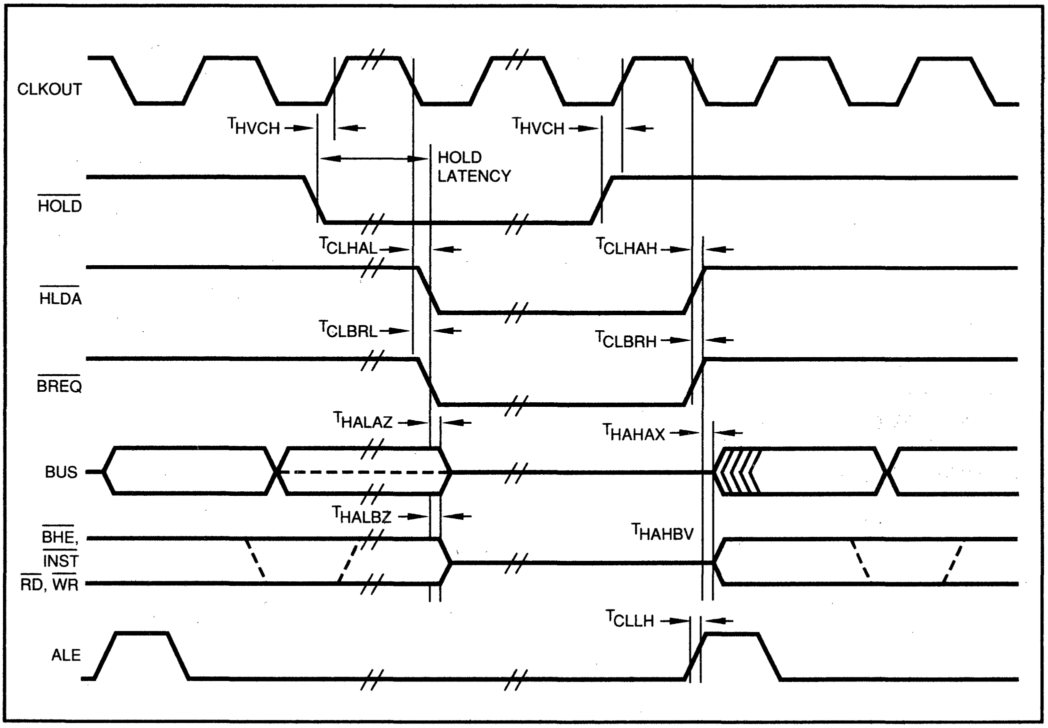


HOLD/HLDA Timings

Symbol	Description	Min	Max	Units	Notes
T_{HVCH}	$\overline{HOLD}$ Setup	90		ns	1
T_{CLHAL}	CLKOUT Low to $\overline{HLDA}$ Low	-15	15	ns	
T_{CLBRL}	CLKOUT Low to $\overline{BREQ}$ Low	-15	15	ns	
T_{HALAZ}	$\overline{HLDA}$ Low to Address Float		-25	ns	
T_{HALBZ}	$\overline{HLDA}$ Low to $\overline{BHE}$, $\overline{INST}$, $\overline{RD}$, $\overline{WR}$ Float		-30	ns	
T_{CLHAH}	CLKOUT Low to $\overline{HLDA}$ High	-15	15	ns	
T_{CLBRH}	CLKOUT Low to $\overline{BREQ}$ High	-15	15	ns	
T_{HAHAX}	$\overline{HLDA}$ High to Address No Longer Float	-5		ns	
T_{HAHBV}	$\overline{HLDA}$ High to $\overline{BHE}$, INST, $\overline{RD}$, $\overline{WR}$ Valid	-25		ns	
T_{CLLH}	CLKOUT Low to ALE High	-5	15	ns	

NOTES:

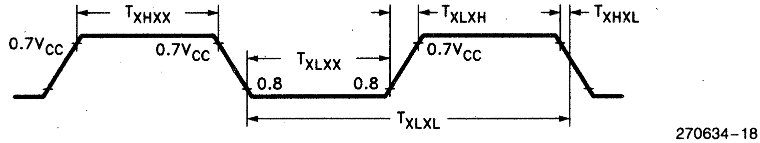
1. To guarantee recognition at next clock.



EXTERNAL CLOCK DRIVE

Symbol	Parameter	Min	Max	Units
$1/T_{XLXL}$	Oscillator Frequency			
	80C196KB12/83C196KB12	3.5	12.0	MHz
	87C196KB10	3.5	10.0	MHz
T_{XLXL}	Oscillator Frequency			
	80C196KB12/83C196KB12	83	286	ns
	87C196KB10	100	286	ns
T_{XLXX}	High Time	32		ns
T_{XLXX}	Low Time	32		ns
T_{XLXH}	Rise Time		10	ns
T_{XHXL}	Fall Time		10	ns

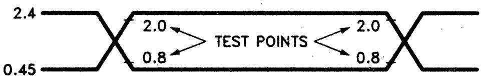
EXTERNAL CLOCK DRIVE WAVEFORMS



270634-18

An external oscillator may encounter as much as a 100 pf load at XTAL1 when it starts-up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{il} and V_{ih} specifications, the capacitance will not exceed 20pf.

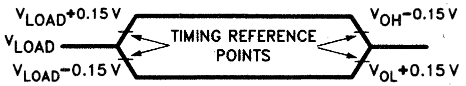
A.C. TESTING INPUT, OUTPUT WAVEFORM



270634-19

A.C. Testing inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0" Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

FLOAT WAVEFORM



270634-20

For Timing Purposes a Port Pin is no Longer Floating when a 100 mV change from Load Voltage Occurs and Begins to Float when a 100 mV change from the Loaded V_{OH}/V_{OL} Level occurs $I_{OL}/I_{OH} = \pm 15$ mA.

EXPLANATION OF AC SYMBOLS

Each symbol is two pairs of letters prefixed by "T" for time. The characters in a pair indicate a signal and its condition, respectively. Symbols represent the time between the two signal/condition points.

Conditions:	Signals:		
H – High	A – Address	G – Buswidth	R – $\overline{\text{RD}}$
L – Low	B – $\overline{\text{BHE}}$	H – $\overline{\text{HOLD}}$	W – $\overline{\text{WR}}/\overline{\text{WRH}}/\overline{\text{WRL}}$
V – Valid	BR – $\overline{\text{BREQ}}$	HA – $\overline{\text{HLDA}}$	X – XTAL1
X – No Longer Valid	C – CLKOUT	L – ALE/$\overline{\text{ADV}}$	Y – READY
Z – Floating	D – DATA IN	Q – DATA OUT	

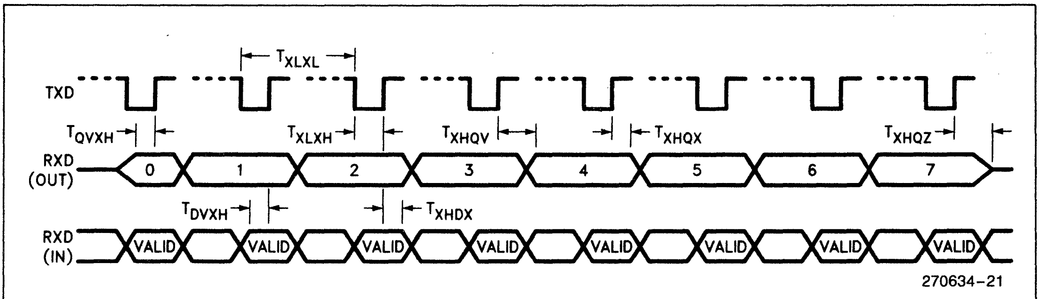
A.C. CHARACTERISTICS—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT TIMING—SHIFT REGISTER MODE

Symbol	Parameter	Min	Max	Units
T_{XLXL}	Serial Port Clock Period (BRR ≥ 8002H)	6 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR ≥ 8002H)	4 T_{OSC} ±50		ns
T_{XLXL}	Serial Port Clock Period (BRR = 8001H)	4 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR = 8001H)	2 T_{OSC} ±50		ns
T_{QVXH}	Output Data Setup to Clock Rising Edge	2 T_{OSC} −50		ns
T_{XHQX}	Output Data Hold after Clock Rising Edge	2 T_{OSC} −50		ns
T_{XHQV}	Next Output Data Valid after Clock Rising Edge		2 T_{OSC} +50	ns
T_{DVXH}	Input Data Setup to Clock Rising Edge	T_{OSC} +50		ns
T_{XHDX}	Input Data Hold after Clock Rising Edge	0		ns
T_{XHQZ}	Last Clock Rising to Output Float		1 T_{OSC}	ns

WAVEFORM—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT WAVEFORM—SHIFT REGISTER MODE



A TO D CHARACTERISTICS

There are two modes of A/D operation: with or without clock prescaler. The speed of the A/D converter can be adjusted by setting a clock prescaler on or off. At high frequencies more time is needed for the comparator to settle. The maximum frequency with the clock prescaler disabled is 6 MHz. The conversion times with the prescaler turned on or off is shown in the table below.

The converter is ratiometric, so the absolute accuracy is directly dependent on the accuracy and stability of V_{REF}. V_{REF} must be close to V_{CC} since it supplies both the resistor ladder and the digital section of the converter.

A/D CONVERTER SPECIFICATIONS

The specifications given below assume adherence to the Operating Conditions section of this data sheet. Testing is performed with V_{REF} = 5.12V.

Clock Prescaler On IOC2.4 = 0	Clock Prescaler Off IOC2.4 = 1
158 States 26.33 μs @ 12 MHz	91 States 30.3 μs @ 6 MHz

Parameter	Typical(1)	Minimum	Maximum	Units*	Notes
Resolution		512 9	1024 10	Levels Bits	
Absolute Error		0	±6	LSBs	
Full Scale Error	0.25±0.50			LSBs	
Zero Offset Error	-0.25±0.50			LSBs	
Non-Linearity Error	1.5±2.5	0	±4	LSBs	
Differential Non-Linearity		>-1	+2	LSBs	
Channel-to-Channel Matching	±0.1	0	±1	LSBs	
Repeatability	±0.25			LSBs	
Temperature Coefficients: Offset Full Scale Differential Non-Linearity	 0.009 0.009 0.009			 LSB/°C LSB/°C LSB/°C	
Off Isolation		−60		dB	2, 3
Feedthrough	−60			dB	2
V_{CC} Power Supply Rejection	−60			dB	2
Input Resistance		1K	5K	Ω	
D.C. Input Leakage		0	3.0	μA	
Sample Time: Prescaler On Prescaler Off	15 8			States States	4 4
Input Capacitance	3			pF	

NOTES:

*An "LSB", as used here, has a value of approximately 5 mV.

1. Typical values are expected for most devices at 25°C but are not tested or guaranteed.
2. DC to 100 KHz.
3. Multiplexer Break-Before-Make Guaranteed.
4. One state = 167 ns at 12 MHz, 333 ns at 6 MHz.

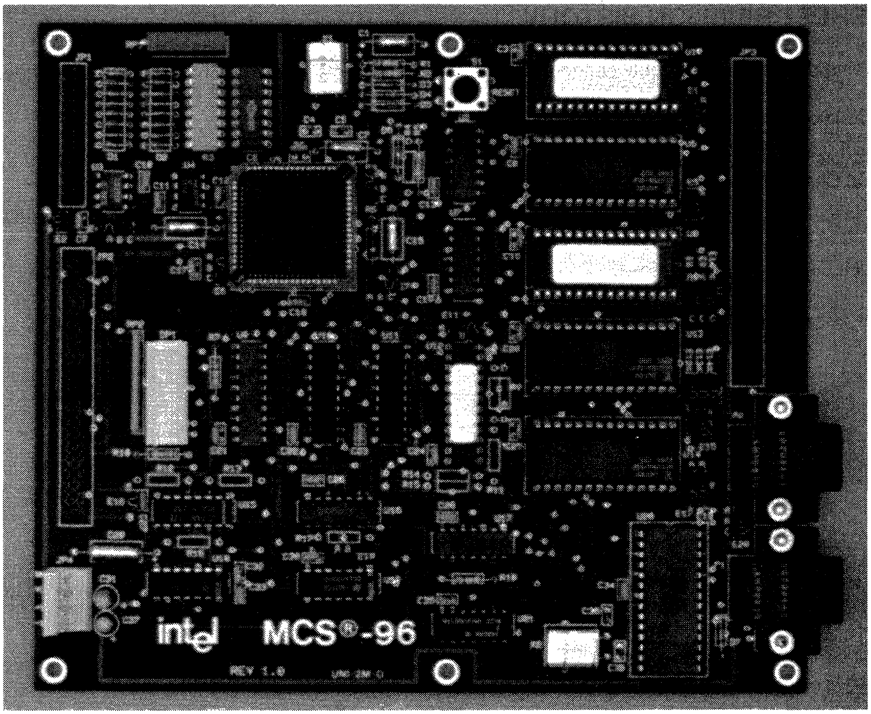
DATA SHEET REVISION HISTORY

The following differences exist between this and the -001 version of the 8XC196KB Express data sheet.

1. The 83C196KB ROM device was added.
2. The CDE (Clock Detect Enable) pin was changed to a V_{SS} pin.
3. Thermal Characteristics for all packages were added.
4. The figures for System Bus Timings were redrawn to include missing timings and to more accurately describe a wait state condition.
5. $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ timings were added.

EV80C196KB EVALUATION BOARD

PRELIMINARY



EV80C196KB FEATURES

- **Zero Wait-State 12 MHz Execution Speed**
- **24K Bytes of ROMsim**
- **Flexible Wait-State, Buswidth, Chip-Select Controller**
- **Totally CMOS, Low Power Board**
- **Concurrent Interrogation of Memory and Registers**
- **Sixteen Software Breakpoints**
- **Two Single Step Modes**
- **High-Level Language Support**
- **Symbolic Debug**
- **RS-232-C Communication Link**

LOW COST CODE EVALUATION TOOL

Intel's EV80C196KB evaluation board provides a hardware environment for code execution and software debugging at a relatively low cost. The board features the 80C196KB advanced, CHMOS*, 16-bit microcontroller, the newest member of the industry standard MCS®-96 family. The board allows the user to take full advantage of the power of the MCS-96. The EV80C196KB provides zero wait-state, 12 MHz execution of a user's code. Plus, its memory (ROMsim) can be reconfigured to match the user's planned memory system, allowing for exact analysis of code execution speeds in a particular application.

*CHMOS is a patented Intel process.
**IBM PC, XT, AT and DOS are registered trademarks of International Business Machines Corporation.

intel®

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodies in an Intel product. No other circuit patent licenses are implied. Information contained herein supersedes previously published specifications on these devices from Intel.

FEBRUARY 1989
Order number: 270729-001

Popular features such as a symbolic single line assembler/disassembler, single-step program execution, and sixteen software breakpoints are standard on the EV80C196KB. Intel provides a complete code development environment using assembler (ASM-96) as well as high-level languages such as Intel's iC-96 or PL/M-96 to accelerate development schedules.

The evaluation board is hosted on an IBM PC** or BIOS-compatible clone, already a standard development solution in most of today's engineering environments. The source code for the on-board monitor (written in ASM-96) is public domain. The program is about 1K, and can be easily modified to be included in the user's target hardware. In this way, the provided PC host software can be used throughout the development phase.

FULL SPEED EXECUTION

The EV80C196KB executes the user's code from on-board ROMsim at 12 MHz with zero wait-states. By changing crystals on the 80C196KB any slower execution speed can be evaluated. The boards host interface timing is not affected by this crystal change.

24K BYTES OF ROMSIM

The board comes with 24K bytes of SRAM to be used as ROMsim for the user's code and as data memory if needed. 16K bytes of this memory are configured as sixteen bits wide, and 8K bytes are configured as eight bits wide. The user can therefore evaluate the speed of the part executing from either buswidth.

FLEXIBLE MEMORY DECODING

By changing the Programable Logic Device (PLD) on the board, the memory on the board can be made to look like the memory system planned for the user's hardware application. The PLD controls the buswidth of the 80C196KB and the chip-select inputs on the board. It also controls the number of wait states (zero to three) generated by the 80C196KB during a memory cycle. These features can all be selected with 256 byte boundaries of resolution.

TOTALLY CMOS BOARD

The EV80C196KB board is built totally with CMOS components. Its power consumption is therefore very low, requiring 5 volts at only 300 mA. If the on board LED's are disabled, the current drops to only 165 mA. The board also requires +/- 12 volts at 15 mA.

CONCURRENT INTERROGATION OF MEMORY AND REGISTERS

The monitor for the EV80C196KB allows the user to read and modify internal registers and external memory while the user's code is running in the board.

SIXTEEN SOFTWARE BREAKPOINTS

There are sixteen breakpoints available which automatically substitute a TRAP instruction for a user's instruction at the breakpoint location. The substitution occurs when execution is started. If the code is halted or a breakpoint is reached, the user's code is restored in the ROMsim.

TWO STEP MODES

There are two single-step modes available. The first stepping mode locks out all interrupts which might occur during the step. The second mode enables interrupts, and treats subroutine calls and interrupt routines as one indivisible instruction.

HIGH LEVEL LANGUAGE SUPPORT

The host software for the EV80C196KB board is able to load absolute object code generated by ASM-96, iC-96, PL/M-96 or RL-96 all of which are available from Intel.

SYMBOLIC DEBUG

The host has a Single Line Assembler, and a Disassmbler, which recognize symbolics generated by Intel software tools.

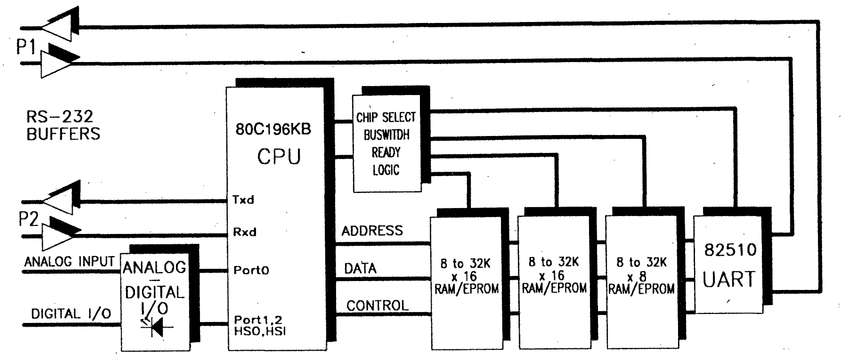
RS-232-C COMMUNICATION LINK

The EV80C196KB communicates with the host using an Intel 82510 UART provided on board. This frees the on-chip UART of the 80C196KB for the user's application.

PERSONAL COMPUTER REQUIREMENTS

The EV80C196KB Evaluation Board is hosted on an IBM PC, XT, AT** or BIOS compatible clone. The PC must meet the following minimum requirements:

- 512K Bytes of Memory
- One 360K Byte floppy Disk Drive
- PC DOS** 3.1 or Later
- A Serial Port (COM1 or COM2) at 9600 Baud
- ASM-96, iC-96 or PL/M-96
- A text editor such as AEDIT



Block Diagram of the 80C196KB Board

80C196KC User's Guide and Data Sheets

5

The 80C196KC family is a CHMOS branch of the MCS®-96 family of high performance, 16-bit microcontrollers. Other members of the MCS-96 family include the 8096BH, 8098 and 80C196KB. All of the MCS-96 components share a common instruction set and architecture. However the CHMOS components have enhancements to provide higher performance and lower power consumptions. The 80C196KC has twice the memory of any previous MCS-96 family member with 488 bytes of RAM and 16K of ROM/EPROM, and at 16 MHz, is 33% faster than an 80C196KB at 12 MHz and at least twice as fast as an 8096BH at 12 MHz. Because some instructions operate in fewer clock cycles than an NMOS device, the 80C196KC can be as much as 2.5–3X the performance of an NMOS device.

The MCS-96 family is a register-to-register architecture, so no accumulator is needed, and most operations can be quickly performed from or to any of the 256 registers. Using the Vertical Windowing scheme, the additional 256 bytes of RAM can also be addressed as registers. In addition, the register operations control the many peripherals which are available on the chips. These peripherals include a serial port, A/D converter, three PWM outputs, up to 48 I/O lines and a High-Speed I/O subsystem which has two 16-bit timer/counters, an 8-level input capture FIFO and an 8-entry programmable output generator.

Typical applications for MCS-96 products are closed-loop control and mid-range digital signal processing. MCS-96 products are being used in modems, motor controls, printers, engine controls, photocopiers, anti-lock brakes, AC motor control, disk drives, and medical instrumentation.

There are many members of the 80C196KC family; to provide easier reading this manual will refer to the family generically as the 80C196KC. Where information applies only to specific components it will be clearly indicated.

This document was written to be a standalone users' guide for anyone wishing to implement a design with the 80C196KC. Those customers who are already familiar with the 80C196KB architecture can proceed to section 18 for a description of the additional features of the 80C196KC. Section 18 also contains the information needed to convert an 80C196KB design to a 80C196KC.

1.0 CPU OPERATION

The major components of the CPU on the 80C196KC are the Register File and the Register/Arithmetic Logic Unit (RALU). Communication with the outside world is done through either the Special Function Registers (SFRs) or the Memory Controller. The RALU does not use an accumulator. Instead, it operates directly on the 256-byte register space made up of the Register File and the SFRs. Efficient I/O operations are possible by directly controlling the I/O through the SFRs. The main benefits of this structure are the ability to quickly change context, absence of accumulator bottleneck, and fast throughput and I/O times.

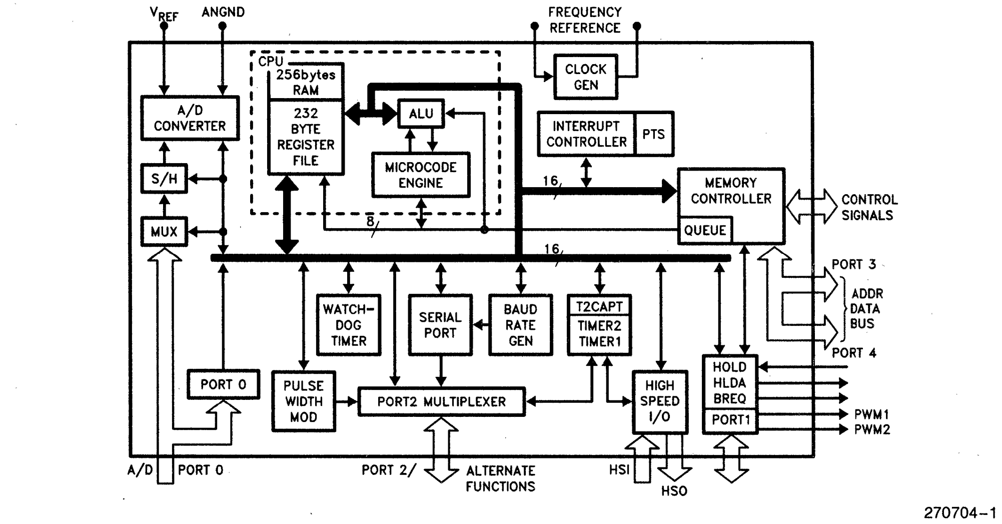


Figure 1-1. 80C196KC Block Diagram

The CPU on the 80C196KC is 16 bits wide and connects to the interrupt controller and the memory controller by a 16-bit bus. In addition, there is an 8-bit bus which transfers instruction bytes from the memory controller to the CPU. An extension of the 16-bit bus connects the CPU to the peripheral devices.

1.1 Memory Controller

The RALU accesses the memory, except for the locations in the register file and SFR space, through the memory controller. Within the memory controller is a bus controller, a four byte prefetch queue and a Slave Program Counter (Slave PC). Both the internal ROM/EPROM bus and the external memory bus are driven by the bus controller. Memory access requests to the bus controller can come from either the RALU or the queue, with queue accesses having priority. Requests from the queue are always for data at the address in the slave PC.

By having program fetches from memory referenced to the slave PC, the processor saves time as addresses seldom have to be sent to the memory controller. If the address sequence changes because of a jump, interrupt, call, or return, the slave PC is loaded with a new value, the queue is flushed, and processing continues.

Execution speed is increased by using a queue since it usually keeps the next instruction byte available. The instruction execution times shown in Section 3 show the normal execution times with no wait states added and the 16-bit bus selected. Reloading the slave PC and fetching the first byte of the new instruction stream takes 4 state times. This is reflected in the jump taken/not-taken times shown in the table.

When debugging code using a logic analyzer, one must be aware of the queue. It is not possible to determine when an instruction will begin executing by simply watching when it is fetched, since the queue is filled in advance of instruction execution.

1.2 CPU Control

A microcode engine controls the CPU, allowing it to perform operations with any byte, word or double word in the 256 byte register space. By using the VWindowing scheme discussed in Section 3-3, the additional 256 bytes of RAM can also be used as registers. Instructions to the CPU are taken from the queue and stored temporarily in the instruction register. The microcode engine decodes the instructions and generates the correct sequence of events to have the RALU perform the desired function. Figure 1-2 shows the memory controller, RALU, instruction register and the control unit.

REGISTER/ALU (RALU)

Most calculations performed by the 80C196KC take place in the RALU. The RALU, shown in Figure 1-2, contains a 17-bit ALU, the Program Status Word (PSW), the Program Counter (PC), a loop counter, and three temporary registers. All of the registers are 16-bits or 17-bits (16+ sign extension) wide. Some of the registers have the ability to perform simple operations to off-load the ALU.

A separate incrementor is used for the Program Counter (PC) as it accesses operands. However, PC changes due to jumps, calls, returns and interrupts must be handled through the ALU. Two of the temporary registers have their own shift logic. These registers are used for the operations which require logical shifts, including Normalize, Multiply, and Divide. The "Lower Word" and "Upper Word" are used together for the 32-bit instructions and as temporary registers for many instructions. Repetitive shifts are counted by the 6-bit "Loop Counter".

A third temporary register stores the second operand of two operand instructions. This includes the multiplier during multiplications and the divisor during divisions. To perform subtractions, the output of this register can be complemented before being placed into the "B" input of the ALU.

Several constants, such as 0, 1 and 2 are stored in the RALU to speed up certain calculations. (e.g. making a 2's complement number or performing an increment or decrement instruction.) In addition, single bit masks for bit test instructions are generated in the constant register based on the 3-bit Bit Select register.

1.3 Internal Timing

The 80C196KC requires an input clock on XTAL1 to function. Since XTAL1 and XTAL2 are the input and output of an inverter a crystal can be used to generate the clock. Details of the circuit and suggestions for its use can be found in Section 14.

Internal operation of the 80C196KC is based on the crystal or external oscillator frequency divided by 2. Every 2 oscillator periods is referred to as one "state time", the basic time measurement for all 80C196KC operations. With a 16 MHz oscillator, a state time is 125 nanoseconds. Since the 80C196KC will be run at many frequencies, the times given throughout this chapter will be in state times or "states", unless otherwise specified. A clock out (CLKOUT) signal, shown in Figure 1-3, is provided as an indication of the internal machine state. Details on timing relationships can be found in Section 14.

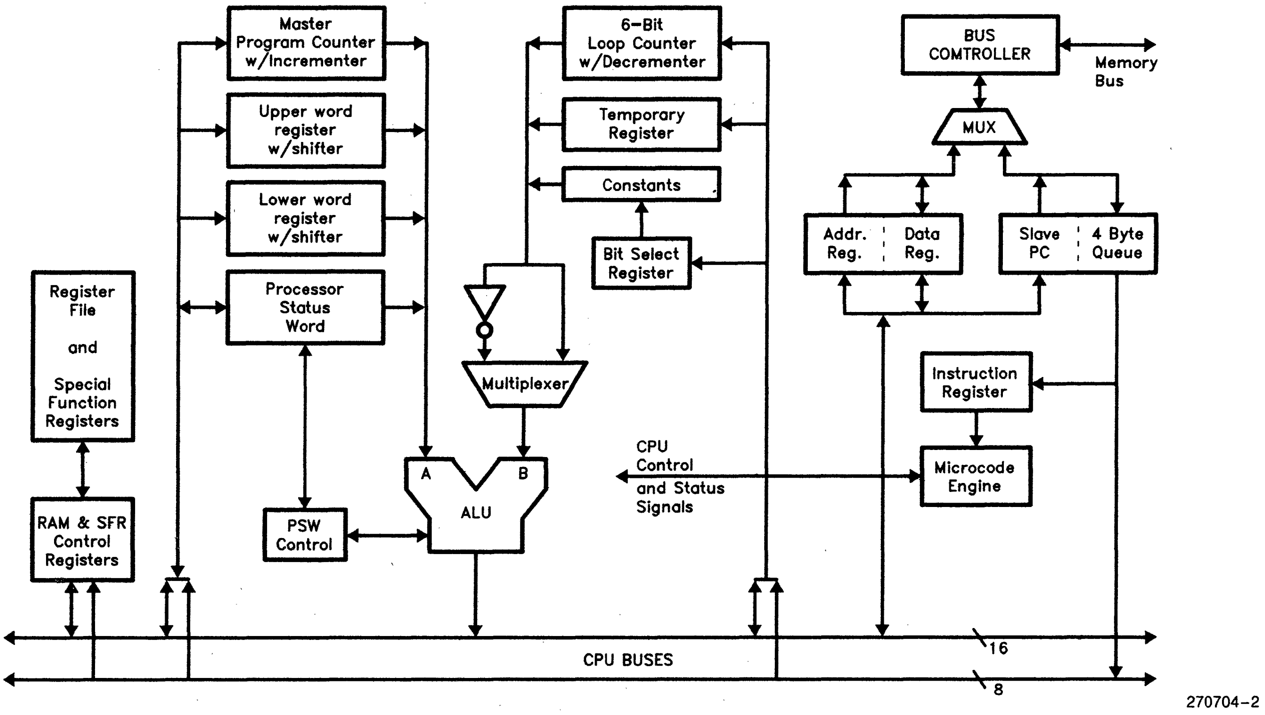


Figure 1-2. RALU and Memory Controller Block Diagram

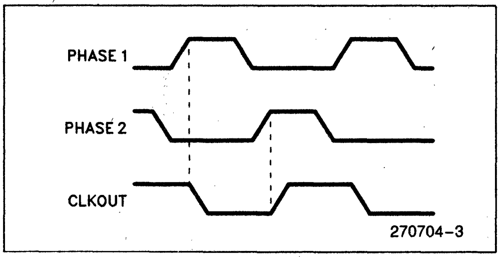


Figure 1-3. Internal Clock Waveforms

2.0 MEMORY SPACE

The addressable memory space on the 80C196KC consists of 64K bytes, most of which is available to the user for program or data memory. Locations which have special purposes are 0000H through 01FFH and 1FFEH through 2080H. All other locations can be used for either program or data storage or for memory mapped peripherals. A memory map is shown in Figure 2-1. Those locations marked "Reserved" must be filled with 0FFHs for future compatibility.

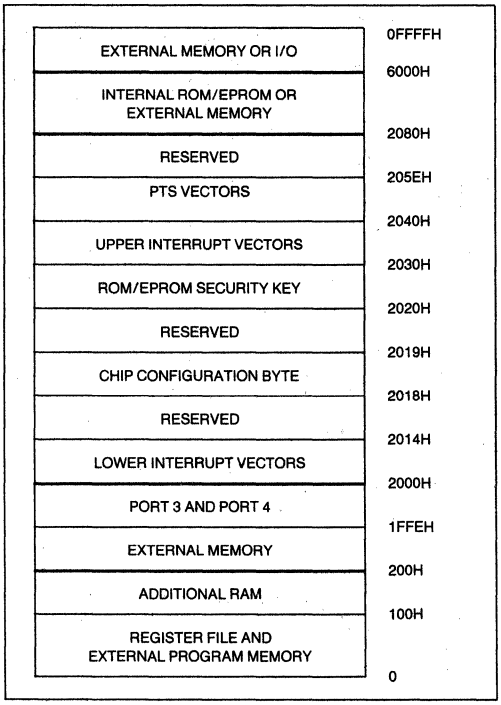


Figure 2-1. 80C196KC Memory Map

2.1 Register File

Locations 00H through 1FFH contain the Register File, Special Function Registers (SFRs), and 256 bytes of additional RAM. If an attempt to execute instructions from locations 000H through 1FFH is made, the instructions will be fetched from *external* memory. This section of external memory is reserved for use by Intel development tools. This memory region, as well as the status of the majority of the chip, is kept intact while the chip is in the Powerdown Mode. Details on the Powerdown Mode are discussed in Section 15.

The internal RAM from location 018H (24 decimal) to 0FFH is the Register File. It contains 232 bytes of RAM which can be accessed as bytes (8 bits), words (16 bits), or double-words (32 bits). Since each of these locations can be used by the RALU, there are essentially 232 "accumulators". Also, the extra 256 bytes of RAM from 100H–1FFH can be accessed as registers by the RALU with Vertical Register Windowing. For more on Register Windowing, see Section 3.3.

Locations 18H and 19H contain the stack pointer. These are not SFRs and may be used as standard RAM if stack operations are not being performed. Since the stack pointer is in this area, the RALU can easily operate on it. The stack pointer must be initialized by the user program and can point anywhere in the 64K memory space. Stack operations cause it to build down, so the stack pointer should be initialized to 2 bytes above the highest stack location. The stack must be word aligned.

2.2 Special Function Registers

Locations 00H through 17H are the I/O control registers or SFRs. All of the peripheral devices on the 80C196KC (except Ports 3 and 4) are controlled through the SFRs. As shown in Figure 2-2, three horizontal windows (HWindows) are provided on the 80C196KC to increase SFR space while remaining upward compatible with earlier MCS-96 products. Switching between Horizontal Windows is discussed in Section 3.3.

HWindow 0 is a superset of the SFR space on the 8096BH and identical to the 80C196KB. As depicted in Figure 2-2, it has 24 registers, some of which have different functions when read than when written.

HWindow 1 contains the additional SFRs needed to support the added functionality of the 80C196KC. These SFRs support the Peripheral Transaction Server (PTS), the two new PWMs, Timer2, and the new functions of the A/D converter. These registers are not needed to remain compatible with the 80C196KB. All SFRs are read/writable in this window.

In register HWindow 15, the operation of the SFRs is changed, so that those which were read-only in HWindow 0 space are write-only and vice versa. The only major exception is Timer2 is read/write in HWindow 0, and T2 Capture is read/write in HWindow 15. Timer2 was read-only on the 8096BH.

Figure 2-3 contains brief descriptions of the SFR registers. Detailed descriptions are contained in the section which discusses the peripheral controlled by the register. Figure 2-4 contains a description of the alternate function in HWindow 15.

Within the SFR space are several registers and bit locations labeled "RESERVED". A reserved bit location must always be written with 0 to maintain compatibility with future parts. Registers and bits which are not labeled should be treated as reserved registers and bits. Note that the default state of internal registers is 0, while that for external memory is 1. This is because SFR functions are typically disabled with a zero, while external memory is typically erased to all 1s.

Caution must be taken when using the SFRs as sources of operations or as base or index registers for indirect or indexed operations. It is possible to get undesired results, since external events can change SFRs and some SFRs clear when read. The potential for an SFR to change value must be taken into account when operating on these registers. This is particularly important when high level languages are used as they may not make allowances for SFR-type registers. SFRs can be operated on as bytes or words unless otherwise specified.

	HWINDOW 0 when Read		HWINDOW 0 when Written		HWINDOW 1 Read/Write		HWINDOW 15
19H	SP (HI)	19H	SP (HI)	19H	SP (HI)	19H	SP (HI)
18H	SP (LO)	18H	SP (LO)	18H	SP (LO)	18H	SP (LO)
17H	IOS2	17H	PWM0_CONTROL	17H	PWM1_CONTROL	17H	
16H	IOS1	16H	IOC1	16H	PWM2_CONTROL	16H	
15H	IOS0	15H	IOC0	15H	RESERVED	15H	
14H	WSR	14H	WSR	14H	WSR	14H	WSR
13H	INT_MASK_1	13H	INT_MASK_1	13H	INT_MASK_1	13H	INT_MASK_1
12H	INT_PEND_1	12H	INT_PEND_1	12H	INT_PEND_1	12H	INT_PEND_1
11H	SP_STAT	11H	SP_CON	11H	RESERVED	11H	
10H	PORT2	10H	PORT2	10H	RESERVED	10H	RESERVED
0FH	PORT1	0FH	PORT1	0FH	RESERVED	0FH	RESERVED
0EH	PORT0	0EH	BAUD RATE	0EH	RESERVED	0EH	RESERVED
0DH	TIMER2 (HI)	0DH	TIMER2 (HI)	0DH	RESERVED	0DH	T2CAP (LO)
0CH	TIMER2 (LO)	0CH	TIMER2 (LO)	0CH	T2CONTROL	0CH	T2CAP (HI)
0BH	TIMER1 (HI)	0BH	IOC2	0BH	RESERVED	0BH	
0AH	TIMER1 (LO)	0AH	WATCHDOG	0AH	RESERVED	0AH	
09H	INT_PEND	09H	INT_PEND	09H	INT_PEND	09H	INT_PEND
08H	INT_MASK	08H	INT_MASK	08H	INT_MASK	08H	INT_MASK
07H	SBUF (RX)	07H	SBUF (TX)	07H	PTSSRV (HI)	07H	
06H	HSI_STATUS	06H	HSO_COMMAND	06H	PTSSRV (LO)	06H	
05H	HSI_TIME (HI)	05H	HSO_TIME (HI)	05H	PTSSEL(HI)	05H	
04H	HSI_TIME (LO)	04H	HSO_TIME (LO)	04H	PTSSEL(LO)	04H	
03H	AD_RESULT (HI)	03H	HSI_MODE	03H	A/D_TIME	03H	
02H	AD_RESULT (LO)	02H	AD_COMMAND	02H	RESERVED	02H	
01H	ZERO_REG (HI)	01H	ZERO_REG (HI)	01H	ZERO_REG (HI)	01H	ZERO_REG (HI)
00H	ZERO_REG (LO)	00H	ZERO_REG (LO)	00H	ZERO_REG (LO)	00H	ZERO_REG (LO)

Figure 2-2. Multiple Register Windows

Register	Description
R0	Zero Register - Always reads as a zero, useful for a base when indexing and as a constant for calculations and compares.
AD_RESULT	A/D Result Hi/Low - Low and high order results of the A/D converter
AD_COMMAND	A/D Command Register - Controls the A/D
HSI_MODE	HSI Mode Register - Sets the mode of the High Speed Input unit.
HSI_TIME	HSI Time Hi/Lo - Contains the time at which the High Speed Input unit was triggered.
HSO_TIME	HSO Time Hi/Lo - Sets the time or count for the High Speed Output to execute the command in the Command Register.
HSO_COMMAND	HSO Command Register - Determines what will happen at the time loaded into the HSO Time registers.
HSI_STATUS	HSI Status Registers - Indicates which HSI pins were detected at the time in the HSI Time registers and the current state of the pins.
SBUF(TX)	Transmit buffer for the serial port, holds contents to be outputted.
SBUF(RX)	Receive buffer for the serial port, holds the byte just received by the serial port.
INT_MASK	Interrupt Mask Register - Enables or disables the individual interrupts.
INT_PEND	Interrupt Pending Register - Indicates that an interrupt signal has occurred on one of the sources and has not been serviced. (also INT_PENDING)
WATCHDOG	Watchdog Timer Register - Written periodically to hold off automatic reset every 64K state times.
TIMER1	Timer 1 Hi/Lo - Timer1 high and low bytes.
TIMER2	Timer 2 Hi/Lo - Timer2 high and low bytes.
IOPORT0	Port 0 Register - Levels on pins of Port 0.
BAUD_RATE	Register which determines the baud rate, this register is loaded sequentially.
IOPORT1	Port 1 Register - Used to read or write to Port 1.
IOPORT2	Port 2 Register - Used to read or write to Port 2.
SP_STAT	Serial Port Status - Indicates the status of the serial port.
SP_CON	Serial Port Control - Used to set the mode of the serial port.
IOS0	I/O Status Register 0 - Contains information on the HSO status.
IOS1	I/O Status Register 1 - Contains information on the status of the timers and of the HSI.
IOC0	I/O Control Register 0 - Controls alternate functions of HSI pins, Timer 2 reset sources and Timer 2 clock sources.
IOC1	I/O Control Register 1 - Controls alternate functions of Port 2 pins, timer interrupts and HSI interrupts.
PWM_CONTROL	Pulse Width Modulation Control Register - Sets the duration of the PWM pulse.

Figure 2-3. Special Function Register Description

Register	Description
INT_PEND1	Interrupt Pending register for the 8 new interrupt vectors (also INT_PENDING1)
INT_MASK1	Interrupt Mask register for the 8 new interrupt vectors
IOC2	I/O Control Register 2
IOS2	I/O Status Register 2 - Contains information on HSO events
WSR	Window Select Register - Selects register window
AD_TIME	Determines A/D Conversion Time
T2CNTC	New 80C196KC features (T2 internal clocking, PWMs)
PTSSEL	Individually enables PTS channels
PTSSRV	End-of-PTS Interrupt Pending Flags

Figure 2-3. Special Function Register Description (Continued)

AD_COMMAND (02H)	— Read the last written command
AD_RESULT (02H, 03H)	— Write a value into the result register
HSI_MODE (03H)	— Read the value in HSI_MODE
HSI_TIME (04H, 05H)	— Write to FIFO Holding register
HSO_TIME (04H, 05H)	— Read the last value placed in the holding register
HSI_STATUS (06H)	— Write to status bits but not to HSI inputs bits. (Inputs bits are 1, 3, 5, 7)
HSO_COMMAND (06H)	— Read the last value placed in the holding register
SBUF(RX) (07H)	— Write a value into the receive buffer
SBUF(TX) (07H)	— Read the last value written to the transmit buffer
WATCHDOG (0AH)	— Read the value in the upper byte of the WDT
TIMER1 (0AH, 0BH)	— Write a value to Timer1
TIMER2 (0CH, 0DH)	— Read/Write the Timer2 capture register. (Timer2 read/write is done with WSR = 0)
IOC2 (0BH)	— Last written value is readable, except bit 7 (Note 1)
BAUD_RATE (0EH)	— No function, cannot be read
PORT0 (0EH)	— No function, no output drivers on the pins
SP_STAT (11H)	— Set the status bits, TI and RI can be set, but it will not cause an interrupt
SP_CON (11H)	— Read the current control byte
IOS0 (15H)	— Writing to this register controls the HSO pins. Bits 6 and 7 are inactive for writes.
IOC0 (15H)	— Last written value is readable, except bit 1 (Note 1)
IOS1 (16H)	— Writing to this register will set the status bits, but not cause interrupts. Bits 6 and 7 are not functional.
IOC1 (16H)	— Last written value is readable
IOS2 (17H)	— Writing to this register will set the status bits, but not cause interrupts.
PWM_CONTROL (17H)	— Read the duty cycle value written to PWM_CONTROL

NOTE:
1. IOC2.7 (CAM CLEAR) and IOC0.1 (T2RST) are not latched and will read as a 1 (precharged bus).

Being able to write to the read-only registers and vice-versa provides a lot of flexibility. One of the most useful advantages is the ability to set the timers and HSO lines for initial conditions other than zero.

Figure 2-4. Alternate SFR Function in HWindow 15

2.3 Internal ROM and EPROM

For a ROM and EPROM part, the internal memory locations 2080H through 5FFFH are user specified, as are the interrupt vectors, and PTS (Peripheral Transaction Server) vectors, Chip Configuration Register and Security Key in locations 2000H through 207FH.

Instruction and data fetches from the internal ROM or EPROM occur only if $\overline{\mathrm{EA}}$ is tied high, and the address is between 2000H and 5FFFH. At all other times data is accessed from either the internal RAM space or external memory and instructions are fetched from external memory. The $\overline{\mathrm{EA}}$ pin is latched on $\overline{\mathrm{RESET}}$ rising. Information on programming EPROMs can be found in Section 17 of this manual.

A security feature can lock the chip against reading and/or writing the internal memory. In order to maintain security, code can not be executed out of the last four locations of internal ROM/EPROM if the lock is enabled. Details on this feature are in Section 17.

2.4 System Bus

There are several modes of system bus operation on the 80C196KC. The standard bus mode uses a 16-bit multiplexed address/data bus. Other bus modes include an 8-bit mode and a mode in which the bus size can dynamically be switched between 8-bits and 16-bits.

Hold/Hold Acknowledge ($\overline{\mathrm{HOLD}}/\overline{\mathrm{HLDA}}$) and Ready signals are available to create a variety of memory systems. The READY line extends the width of the $\overline{\mathrm{RD}}$ (read) and $\overline{\mathrm{WR}}$ (write) pulses to allow access of slow memories. Multiple processor systems with shared memory can be designed using $\overline{\mathrm{HOLD}}/\overline{\mathrm{HLDA}}$. Details on the System Bus are in Sections 15 and 16.

3.0 SOFTWARE OVERVIEW

This section provides information on writing programs to execute in the 80C196KC. Additional information can be found in the following documents:

MCS®-96 MACRO ASSEMBLER USER'S GUIDE
Order Number 122048 (Intel Systems)
Order Number 122351 (DOS Systems)

MCS®-96 UTILITIES USER'S GUIDE
Order Number 122049 (Intel Systems)
Order Number 122356 (DOS Systems)

PL/M-96 USER'S GUIDE
Order Number 122134 (Intel Systems)
Order Number 122361 (DOS Systems)

C-96 USER'S GUIDE
Order Number 167632 (DOS Systems)

Throughout this chapter short sections of code are used to illustrate the operation of the device. For these sections it is assumed that the following set of temporary registers has been declared:

AX, BX, CX, and DX are 16-bit registers.

AL is the low byte of AX, AH is the high byte.

BL is the low byte of BX

CL is the low byte of CX

DL is the low byte of DX

These are the same as the names for the general data registers used in the 8086. In the 80C196KC these are not dedicated registers but merely the symbolic names assigned by the programmer to an eight byte region within the on-board register file.

3.1 Operand Types

The MCS-96 architecture supports a variety of data types likely to be useful in a control application. To avoid confusion, the name of an operand type is capitalized. A "BYTE" is an unsigned eight bit variable; a "byte" is an eight bit unit of data of any type.

BYTES

BYTES are unsigned 8-bit variables which can take on the values between 0 and 255. Arithmetic and relational operators can be applied to BYTE operands but the result must be interpreted in modulo 256 arithmetic. Logical operations on BYTES are applied bitwise. Bits within BYTES are labeled from 0 to 7, with 0 being the least significant bit.

WORDS

WORDS are unsigned 16-bit variables which can take on the values between 0 and 65535. Arithmetic and relational operators can be applied to WORD operands but the result must be interpreted modulo 65536. Logical operations on WORDS are applied bitwise. Bits within words are labeled from 0 to 15 with 0 being the least significant bit. WORDS must be aligned at even byte boundaries in the MCS-96 address space. The least significant byte of the WORD is in the even byte address and the most significant byte is in the next higher (odd) address. The address of a word is the address of its least significant byte. Word operations to odd addresses are not guaranteed to operate in a consistent manner.

SHORT-INTEGERS

SHORT-INTEGERS are 8-bit signed variables which can take on the values between −128 and +127. Arithmetic operations which generate results outside of the range of a SHORT-INTEGER will set the overflow indicators in the program status word. The actual numeric result returned will be the same as the equivalent operation on BYTE variables.

INTEGERS

INTEGERS are 16-bit signed variables which can take on the values between −32,768 and +32,767. Arithmetic operations which generate results outside of the range of an INTEGER will set the overflow indicators in the program status word. The actual numeric result returned will be the same as the equivalent operation on WORD variables. INTEGERS conform to the same alignment and addressing rules as do WORDS.

BITS

BITS are single-bit operands which can take on the Boolean values of true and false. In addition to the normal support for bits as components of BYTE and WORD operands, the 80C196KC provides for the direct testing of any bit in the register file. The MCS-96 architecture requires that bits be addressed as components of BYTES or WORDS, it does not support the direct addressing of bits that can occur in the MCS-51 architecture.

DOUBLE-WORDS

DOUBLE-WORDS are unsigned 32-bit variables which can take on the values between 0 and 4,294,967,295. The MCS-96 architecture provides direct support for this operand type only for shifts, as the dividend in a 32 by 16 divide, the product of a 16 by 16 multiply, and for double-word compares. For these operations a DOUBLE-WORD variable must reside in the on-board register file of the 80C196KC and be aligned at an address which is evenly divisible by 4. A DOUBLE-WORD operand is addressed by the address of its least significant byte. DOUBLE-WORD operations which are not directly supported can be easily implemented with two WORD operations. The CMPL instruction views the zero register as a 32-bit value of zero. This allows it to be used for comparison to zero. For consistency with Intel provided software the user should adopt the conventions for addressing DOUBLE-WORD operands which are discussed in Section 3.6.

LONG-INTEGERS

LONG-INTEGERS are 32-bit signed variables which can take on the values between −2,147,483,648 and +2,147,483,647.

LONG-INTEGERS can also be normalized. For these operations a LONG-INTEGER variable must reside in the onboard register file of the 80C196KC and be aligned at an address which is evenly divisible by 4. A LONG-INTEGER is addressed by the address of its least significant byte.

LONG-INTEGER operations which are not directly supported can be easily implemented with two INTEGER operations.

3.2 Operand Addressing

Operands are accessed within the address space of the 80C196KC with one of six basic addressing modes. Some of the details of how these addressing modes work are hidden by the assembly language. If the programmer is to take full advantage of the architecture, it is important that these details be understood. This section will describe the addressing modes as they are handled by the hardware. At the end of this section the addressing modes will be described as they are seen through the assembly language. The six basic address modes which will be described are termed register-direct, indirect, indirect with auto-increment, immediate, short-indexed, and long-indexed. Several other useful addressing operations can be achieved by combining these basic addressing modes with specific registers such as the ZERO register or the stack pointer.

REGISTER-DIRECT REFERENCES

The register-direct mode is used to directly access a register from the 256 byte on-board register file. The register is selected by an 8-bit field within the instruction and the register address must conform to the operand type's alignment rules. Depending on the instruction, up to three registers can take part in the calculation.

Examples

```
ADD    AX,BX,CX      ; AX:=BX+CX
MUL    AX,BX         ; AX:=AX*BX
INCB   CL            ; CL:=CL+1
```

INDIRECT REFERENCES

The indirect mode is used to access an operand by placing its address in a WORD variable in the register file. The calculated address must conform to the alignment rules for the operand type. Note that the indirect address can refer to an operand anywhere within the address space of the 80C196KC, including the register file. The register which contains the indirect address is selected by an eight bit field within the instruction. An instruction can contain only one indirect reference and the remaining operands of the instruction (if any) must be register-direct references.

Examples

```
LD     AX,[AX]      ; AX:=MEM_WORD(AX)
ADDB   AL,BL,[CX]   ; AL:=BL+MEM_BYTE(CX)
POP    [AX]         ; MEM_WORD(AX)
                    ;  :=MEM_WORD(SP)
                    ;  SP:=SP+2
```

INDIRECT WITH AUTO-INCREMENT REFERENCES

This addressing mode is the same as the indirect mode except that the WORD variable which contains the indirect address is incremented *after* it is used to address the operand. If the instruction operates on BYTES or SHORT-INTEGERS the indirect address variable will be incremented by one, if the instruction operates on WORDS or INTEGERS the indirect address variable will be incremented by two.

Examples

```
LD     AX,[BX]+      ; AX:=MEM_WORD(BX) ; BX:=BX+2
ADDB   AL,BL,[CX]+   ; AL:=BL+MEM_BYTE(CX) ; CX:=CX+1
PUSH   [AX]+         ; SP:=SP-2;
                     ;   MEM_WORD(SP):=MEM_WORD(AX)
                     ;   AX:=AX+2
```

IMMEDIATE REFERENCES

This addressing mode allows an operand to be taken directly from a field in the instruction. For operations on BYTE or SHORT-INTEGER operands this field is eight bits wide, for operations on WORD or INTEGER operands the field is 16 bits wide. An instruction can contain only one immediate reference and the remaining operand(s) must be register-direct references.

Examples

```
ADD  AX,#340  ; AX:=AX+340
PUSH #1234H   ; SP:=SP-2; MEM_WORD(SP):=1234H
DIVB AX,#10   ; AL:=AX/10; AH:=AX MOD 10
```

SHORT-INDEXED REFERENCES

In this addressing mode an eight bit field in the instruction selects a WORD variable in the register file which contains an address. A second eight bit field in the instruction stream is sign-extended and summed with the WORD variable to form the address of the operand which will take part in the calculation.

Since the eight bit field is sign-extended, the effective address can be up to 128 bytes before the address in the WORD variable and up to 127 bytes after it. An instruction can contain only one short-indexed reference and the remaining operand(s) must be register-direct references.

Examples

```
LD     AX,12[BX]      ; AX:=MEM_WORD(BX+12)
MULB   AX,BL,3[CX]    ; AX:=BL*MEM_BYTE(CX+3)
```

LONG-INDEXED REFERENCES

This addressing mode is like the short-indexed mode except that a *16-bit* field is taken from the instruction and added to the WORD variable to form the address of the operand. No sign extension is necessary. An instruction can contain only one long-indexed reference and the remaining operand(s) must be register-direct references.

Examples

```
AND  AX,BX,TABLE[CX]     ; AX:=BX AND MEM_WORD(TABLE+CX)
ST   AX,TABLE[BX]        ; MEM_WORD(TABLE+BX):=AX
ADDB AL,BL,LOOKUP[CX]    ; AL:=BL+MEM_BYTE(LOOKUP+CX)
```

ZERO REGISTER ADDRESSING

The first two bytes in the register file are fixed at zero by the 80C196KC hardware. In addition to providing a fixed source of the constant zero for calculations and comparisons, this register can be used as the WORD variable in a long-indexed reference. This combination of register selection and address mode allows any location in memory to be addressed directly.

Examples

```
ADD  AX,1234[0]   ; AX:=AX+MEM_WORD(1234)
POP  5678[0]      ; MEM_WORD(5678):=MEM_WORD(SP)
                  ;  SP:=SP+2
```

STACK POINTER REGISTER ADDRESSING

The system stack pointer in the 80C196KC is accessed as register 18H of the internal register file. In addition to providing for convenient manipulation of the stack pointer, this also facilitates the accessing of operands in the stack. The top of the stack, for example, can be accessed by using the stack pointer as the WORD variable in an indirect reference. In a similar fashion, the stack pointer can be used in the short-indexed mode to access data within the stack.

Examples

```
PUSH  [SP]       ; DUPLICATE TOP_OF_STACK
LD    AX,2[SP]   ; AX:=NEXT_TO_TOP
```

ASSEMBLY LANGUAGE ADDRESSING MODES

The MCS-96 assembly language simplifies the choice of addressing modes to be used in several respects:

Direct Addressing. The assembly language will choose between register-direct addressing and long-indexed with the ZERO register depending on where the operand is in memory. The user can simply refer to an operand by its symbolic name; if the operand is in the register file, a register-direct reference will be used, if the operand is elsewhere in memory, a long-indexed reference will be generated.

Indexed Addressing. The assembly language will choose between short and long indexing depending on the value of the index expression. If the value can be expressed in eight bits then short indexing will be used, if it cannot be expressed in eight bits then long indexing will be used.

These features of the assembly language simplify the programming task and should be used wherever possible.

3.3 Register Windowing

One of the biggest advantages of the MSC-96 architecture is its ability to perform operations directly on the Register File without using accumulators. The Register Direct Addressing Mode makes for efficient code that is easy to write. The RALU accesses the Register file using eight bit addressing, making 256 bytes available to the RALU at one time. Register Windowing was implemented so the RALU could have access to more than 256 bytes of Registers by simply switching windows. There are two types of Windows: Horizontal Windows and Vertical Windows. Switching between Windows is controlled by the WSR (Window Select Register) shown in Figure 3-1. The 7 LSBs of the WSR control the Windowing and the MSB (HLDEN) enables the $\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ function.

	7							0
WSR	HLDEN	W6	W5	W4	W3	W2	W1	W0
								14H

Figure 3-1. Window Select Register

HORIZONTAL WINDOWS

Horizontal Windows (HWindows) contain the extra SFRs for the 80C196KC. Three HWindows are supported on the 80C196KC, 0, 1, and 15. Figures 2-3 and 2-4 in Section 2 show the HWindows and their contents. HWindow 15 is compatible with HWindow 15 on the 80C196KB and HWindow 1contains extra SFRs to support the additional functionality of the 80C196KC. Switching Horizontal Windows maps the 24 bytes of the HWindow into the lowest 24 bytes of the Register File. There is no other way to access a Horizontal Window. To switch between HWindows, simply write the number of the HWindow into the four LSBs of the WSR. Bits 4–6 of the WSR must be written as 0s when accessing Horizontal Windows.

VERTICAL WINDOWS

Vertical Windows (VWindows) can be used to map sections of the 512 bytes of RAM from 00H–1FFH into the upper section of the Register File. An important difference between Horizontal and Vertical Windows is VWindows reside directly in the 80C196KC addressing space. Therefore, 100H–1FFH can be addressed directly with 16-bit addressing using an indexed or indirect addressing mode, or as registers using Vertical Windows.

Vertical Windowing allows, 32-, 64-, or 128-byte "windows" from 00–1FFH to be mapped onto the top 32-, 64-, or 128-byte block of the Register File. Figure 3-2 shows all the available VWindows on the 80C196KC. Switching between VWindows is done by setting bit 6, 5, or 4 in the WSR and writing the number of the VWindow into the 4 LSBs. Figure 3-3 shows how to use the WSR to switch between VWindows.

For an example, let's map the 32-byte block from 120H–13FH into the upper part of the Register File from 0E0H–0FFH. Figure 3-4 shows the VWindow being switched as well as the correct value to load into the WSR. Now any access to locations 0E0H–0FFH using a register direct reference will actually access the memory at 120H–13FH. However the two locations can still be accessed directly with 16-bit addressing. The section of code in Figure 3-4 should clarify this.

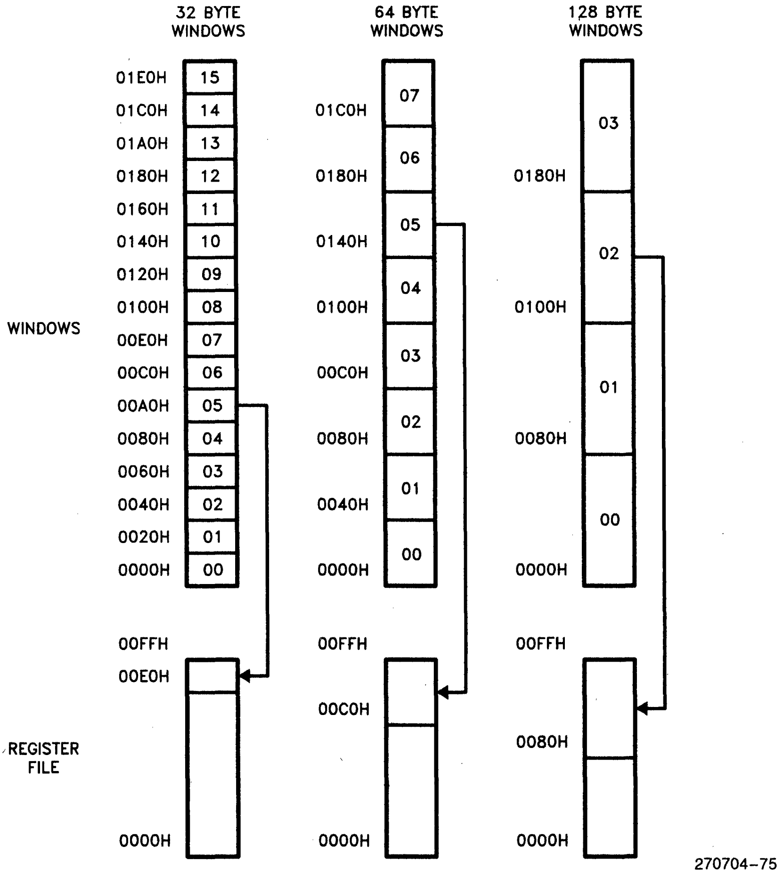


Figure 3-2. Vertical Windows

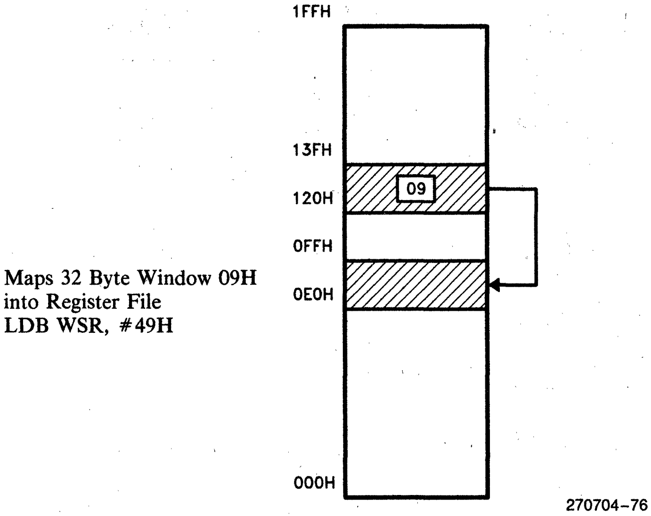
Forms A 9 Bit Address

	7						0	
WSR	HLDEN	W6	W5	W4	W3	W2	W1	W0	14H

WWW WWWW = 000 XXXX: Select Horizontal Window (3 Possible)
= 100 XXXX: Select 32-Byte Window (16 Possible)
= 010 0XXX: Select 64-Byte Window (8 Possible)
= 001 00XX: Select 128-Byte Window (4 Possible)

32-Byte Window Addresses:	W3	W2	W1	W0	A4	A3	A2	A1	A0
64-Byte Window Addresses:	W2	W1	W0	A5	A4	A3	A2	A1	A0
128-Byte Window Addresses:	W1	W0	A6	A5	A4	A3	A2	A1	A0

Figure 3-3. Accessing a VWindow



Maps 32 Byte Window 09H
into Register File
LDB WSR, #49H

```
ldb WSR_IMAGE, WSR
ldb WSR, #49H         ;switch windows
add 40H, 0F0H         ;adds the value at 130H to the
                      ;value at 40H
add 40H, 0F0H[0]      ;adds the value at 0F0H to the
                      ;value at 40H
add 40H, 130H [0]     ;adds the value at 130H to the
                      ;value at 40H
ldb WSR, WSR_IMAGE    ;switch back to previous window
```

Figure 3-4. VWindow Example

VWindowing provides for fast context switching of register sets. For example, an Interrupt Service Routine could have its own set of local registers in a VWindow, and pass results to a main routine through global registers in the Register File.

3.4 Program Status Word

The program status word (PSW) is a collection of Boolean flags which retain information concerning the state of the user's program. There are two bytes in the PSW; the actual status word and the low byte of the interrupt mask. Figure 3-5 shows the status bits of the PSW. The PSW can be saved in the system stack with a single operation (PUSHF) and restored in a like manner (POPF). Only the interrupt section of the PSW can be accessed directly. There is no SFR for the PSW status bits.

CONDITION FLAGS

The PSW bits on the 80C196KC are set as follows:

	7	6	5	4	3	2	1	0
PSW:	Z	N	V	VT	C	PSE	I	ST

Figure 3-5. PSW Register

Z: The Z (Zero) flag is set to indicate that the operation generated a result equal to zero. For the add-with-carry (ADDC) and subtract-with-borrow (SUBC) operations the Z flag is cleared if the result is non-zero but is never set. These two instructions are normally used in conjunction with the ADD and SUB instructions to perform multiple precision arithmetic. The operation of the Z flag for these instructions leaves it indicating the proper result for the entire multiple precision calculation.

N: The Negative flag is set to indicate that the operation generated a negative result. Note that the N flag will be in the algebraically correct state even if an overflow occurs. For shift operations, including the normalize operation and all three forms (SHL, SHR, SHRA) of byte, word and double word shifts, the N flag will be set to the same value as the most significant bit of the result. This will be true even if the shift count is 0.

V: The oVerflow flag is set to indicate that the operation generated a result which is outside the range for the destination data type. For the SHL, SHLB and SHLL instructions, the V flag will be set if the most significant bit of the operand changes at any time during the shift. For divide operations, the following conditions are used to determine if the V flag is set:

For the operation:	V is set if Quotient is:
UNSIGNED BYTE DIVIDE	> 255(0FFH)
UNSIGNED WORD DIVIDE	> 65535(0FFFFH)
SIGNED BYTE DIVIDE	< −127(81H) or > 127(7FH)
SIGNED WORD DIVIDE	< −32767(8001H) or > 32767(7FFFH)

VT: The oVerflow Trap flag is set when the V flag is set, but it is only cleared by the CLRVT, JVT and JNVT instructions. The operation of the VT flag allows for the testing for a possible overflow condition at the end of a sequence of related arithmetic operations. This is normally more efficient than testing the V flag after each instruction.

C: The Carry flag is set to indicate the state of the arithmetic carry from the most significant bit of the ALU for an arithmetic operation, or the state of the last bit shifted out of an operand for a shift. Arithmetic Borrow after a subtract operation is the complement of the C flag (i.e. if the operation generated a borrow then C=0.)

PSE: The Peripheral Transaction Server Enable bit. Globally enables the PTS when set. Manipulated by the EPTS and DPTS instructions.

I: The global Interrupt disable bit disables all interrupts except NMI, TRAP, and unimplemented opcode when cleared.

ST: The ST (STicky bit) flag is set to indicate that during a right shift a 1 has been shifted first into the C flag and then been shifted out. The ST flag is undefined after a multiply operation. The ST flag can be used along with the C flag to control rounding after a right shift. Consider multiplying two eight bit quantities and then scaling the result down to 12 bits:

```
MULUB  AX,CL,DL    ;AX:=CL*DL
SHR    AX,#4       ;Shift right 4
                    places
```

If the C flag is set after the shift, it indicates that the bits shifted off the end of the operand were greater-than or equal-to one half the least significant bit (LSB) of the result. If the C flag is clear after the shift, it indicates that the bits shifted off the end of the operand were less than half the LSB of the result. Without the ST flag, the rounding decision must be made on the basis of the C flag alone. (Normally the result would be rounded up if the C flag is set.) The ST flag allows a finer resolution in the rounding decision:

C	ST	Value of the Bits Shifted Off
0	0	Value = 0
0	1	0 < Value < ½ LSB
1	0	Value = ½ LSB
1	1	Value > ½ LSB

Figure 3-6. Rounding Alternatives

Imprecise rounding can be a major source of error in a numerical calculation; use of the ST flag improves the options available to the programmer.

INTERRUPT FLAGS

The lower eight bits of the PSW individually mask the lowest 8 sources of interrupt to the 80C196KC. These mask bits can be accessed as an eight bit byte (INT__MASK—address 8) in the register file. A separate register (INT__MASK1—address 13H) contains the control bits for the higher 8 interrupts. A logical '1' in these bit positions enables the servicing of the corresponding interrupt. Bit 9 in the PSW is the global interrupt disable. If this bit is cleared then interrupts will be locked out. Further information on the interrupt structure of the 80C196KC can be found in Section 5.

3.5 Instruction Set

The MCS-96 instruction set contains a full set of arithmetic and logical operations for the 8-bit data types BYTE and SHORT INTEGER and for the 16-bit data types WORD and INTEGER. The DOUBLE-WORD and LONG data types (32 bits) are supported for shifts, products of 16 by 16 multiplies, dividends of 32-by-16 divides, and for 32-bit compares. The remaining operations on 32-bit variables can be implemented by combinations of 16-bit operations. As an example the sequence:

```
ADD     AX,CX
ADDC    BX,DX
```

performs a 32-bit addition, and the sequence

```
SUB     AX,CX
SUBC    BX,DX
```

performs a 32-bit subtraction. Operations on REAL (i.e. floating point) variables are not supported directly by the hardware but are supported by the floating point library for the 80C196KC (FPAL-96) which implements a single precision subset of Draft 10 of the IEEE Standard for Floating Point Arithmetic. The performance of this software is significantly improved by the 80C196KC NORML instruction which normalizes a 32-bit variable and by the existence of the ST flag in the PSW.

In addition to the operations on the various data types, the 80C196KC supports conversions between these types. LDBZE (load byte zero extended) converts a BYTE to a WORD and LDBSE (load byte sign extended) converts a SHORT-INTEGER into an INTEGER. WORDS can be converted to DOUBLE-WORDS by simply clearing the upper WORD of the DOUBLE-WORD (CLR) and INTEGERS can be converted to LONGS with the EXT (sign extend) instruction.

The MCS-96 instructions for addition, subtraction, and comparison do not distinguish between unsigned words and signed integers. Conditional jumps are provided to allow the user to treat the results of these operations as either signed or unsigned quantities. As an example, the CMPB (compare byte) instruction is used to compare both signed and unsigned eight bit quantities. A JH (jump if higher) could be used following the compare if unsigned operands were involved or a JGT (jump if greater-than) if signed operands were involved.

Tables 3-7 and 3-8 summarize the operation of each of the instructions.

The execution times for the instruction set is given in Figure 3-8. These times are given for a 16-bit bus with no waitstates. On-chip EPROM/ROM space is a 16-bit, zero waitstate bus. When executing from an 8-bit external memory system or adding waitstates, the CPU becomes bus limited and must sometimes wait for the prefetch queue. The performance penalty for an 8-bit external bus is difficult to measure, but has shown to be between 10 and 30 percent based on the instruction mix. The best way to measure code performance is to actually benchmark the code and time it using an emulator or with TIMER1.

The indirect and indexed instruction timings are given for two memory spaces; SFR/Internal RAM space (0–1FFH), and a memory controller reference (200H–0FFFFH). Any instruction that uses an operand that is referenced thru the memory controller (ex. Add r1,5000H[0]) takes 2–3 states longer than if the operand was in the SFR/Internal RAM space. Any data access to on-chip ROM/EPROM is considered to be a memory controller reference.

Flag Settings. The modification to the flag setting is shown for each instruction. A checkmark (✓) means that the flag is set or cleared as appropriate. A hyphen means that the flag is not modified. A one or zero (1) or (0) indicates that the flag will be in that state after the instruction. An up arrow (↑) indicates that the instruction may set the flag if it is appropriate but will not clear the flag. A down arrow (↓) indicates that the flag can be cleared but not set by the instruction. A question mark (?) indicates that the flag will be left in an indeterminant state after the operation.

Table 3-7A. Instruction Summary

Mnemonic	Operands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
ADD/ADDB	2	D ← D + A	✓	✓	✓	✓	↑	–	
ADD/ADDB	3	D ← B + A	✓	✓	✓	✓	↑	–	
ADDC/ADDCB	2	D ← D + A + C	↓	✓	✓	✓	↑	–	
SUB/SUBB	2	D ← D – A	✓	✓	✓	✓	↑	–	
SUB/SUBB	3	D ← B – A	✓	✓	✓	✓	↑	–	
SUBC/SUBCB	2	D ← D – A + C – 1	↓	✓	✓	✓	↑	–	
CMP/CMPB	2	D – A	✓	✓	✓	✓	↑	–	
MUL/MULU	2	D,D + 2 ← D × A	–	–	–	–	–	–	2
MUL/MULU	3	D,D + 2 ← B × A	–	–	–	–	–	–	2
MULB/MULUB	2	D,D + 1 ← D × A	–	–	–	–	–	–	3
MULB/MULUB	3	D,D + 1 ← B × A	–	–	–	–	–	–	3
DIVU	2	D ← (D,D + 2) /A,D + 2 ← remainder	–	–	–	✓	↑	–	2
DIVUB	2	D ← (D,D + 1) /A,D + 1 ← remainder	–	–	–	✓	↑	–	3
DIV	2	D ← (D,D + 2) /A,D + 2 ← remainder	–	–	–	✓	↑	–	
DIVB	2	D ← (D,D + 1) /A,D + 1 ← remainder	–	–	–	✓	↑	–	
AND/ANDB	2	D ← D AND A	✓	✓	0	0	–	–	
AND/ANDB	3	D ← B AND A	✓	✓	0	0	–	–	
OR/ORB	2	D ← D OR A	✓	✓	0	0	–	–	
XOR/XORB	2	D ← D (ecxl. or) A	✓	✓	0	0	–	–	
LD/LDB	2	D ← A	–	–	–	–	–	–	
ST/STB	2	A ← D	–	–	–	–	–	–	
XCH/XCHB	2	D ← A, A ← D	–	–	–	–	–	–	
LDBSE	2	D ← A; D + 1 ← SIGN(A)	–	–	–	–	–	–	3,4
LDBZE	2	D ← A; D + 1 ← 0	–	–	–	–	–	–	3,4
PUSH	1	SP ← SP – 2; (SP) ← A	–	–	–	–	–	–	
POP	1	A ← (SP); SP + 2	–	–	–	–	–	–	
PUSHF	0	SP ← SP – 2; (SP) ← PSW; PSW ← 0000H; I ← 0	0	0	0	0	0	0	
POPF	0	PSW ← (SP); SP ← SP + 2; I ← ✓	✓	✓	✓	✓	✓	✓	
SJMP	1	PC ← PC + 11-bit offset	–	–	–	–	–	–	5
LJMP	1	PC ← PC + 16-bit offset	–	–	–	–	–	–	5
BR[indirect]	1	PC ← (A)	–	–	–	–	–	–	
TIJMP	3	PC ← [A] + 2 * ([B] AND C)	–	–	–	–	–	–	
SCALL	1	SP ← SP – 2; (SP) ← PC; PC ← PC + 11-bit offset	–	–	–	–	–	–	5
LCALL	1	SP ← SP – 2; (SP) ← PC; PC ← PC + 16-bit offset	–	–	–	–	–	–	5

Table 3-7B. Instruction Summary

Mnemonic	Operands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
RET	0	PC ← (SP); SP ← SP + 2	–	–	–	–	–	–	
J (conditional)	1	PC ← PC + 8-bit offset (if taken)	–	–	–	–	–	–	5
JC	1	Jump if C = 1	–	–	–	–	–	–	5
JNC	1	jump if C = 0	–	–	–	–	–	–	5
JE	1	jump if Z = 1	–	–	–	–	–	–	5
JNE	1	Jump if Z = 0	–	–	–	–	–	–	5
JGE	1	Jump if N = 0	–	–	–	–	–	–	5
JLT	1	Jump if N = 1	–	–	–	–	–	–	5
JGT	1	Jump if N = 0 and Z = 0	–	–	–	–	–	–	5
JLE	1	Jump if N = 1 or Z = 1	–	–	–	–	–	–	5
JH	1	Jump if C = 1 and Z = 0	–	–	–	–	–	–	5
JNH	1	Jump if C = 0 or Z = 1	–	–	–	–	–	–	5
JV	1	Jump if V = 0	–	–	–	–	–	–	5
JNV	1	Jump if V = 1	–	–	–	–	–	–	5
JVT	1	Jump if VT= 1; Clear VT	–	–	–	–	0	–	5
JNVT	1	Jump if VT = 0; Clear VT	–	–	–	–	0	–	5
JST	1	Jump if ST = 1	–	–	–	–	–	–	5
JNST	1	Jump if ST = 0	–	–	–	–	–	–	5
JBS	3	Jump if Specified Bit = 1	–	–	–	–	–	–	5,6
JBC	3	Jump if Specified Bit = 0	–	–	–	–	–	–	5,6
DJNZ/ DJNZW	1	D ← D – 1; If D ≠ 0 then PC ← PC + 8-bit offset	–	–	–	–	–	–	5
DEC/DECB	1	D ← D – 1	✓	✓	✓	✓	↑	–	
NEG/NEGB	1	D ← 0 – D	✓	✓	✓	✓	↑	–	
INC/INCB	1	D ← D + 1	✓	✓	✓	✓	↑	–	
EXT	1	D ← D; D + 2 ← Sign (D)	✓	✓	0	0	–	–	2
EXTB	1	D ← D; D + 1 ← Sign (D)	✓	✓	0	0	–	–	3
NOT/NOTB	1	D ← Logical Not (D)	✓	✓	0	0	–	–	
CLR/CLRB	1	D ← 0	1	0	0	0	–	–	
SHL/SHLB/SHLL	2	C ← msb - - - - - lsb ← 0	✓	✓	✓	✓	↑	–	7
SHR/SHRB/SHRL	2	0 → msb - - - - - lsb → C	✓	✓	✓	0	–	✓	7
SHRA/SHRAB/SHRAL	2	msb → msb - - - - - lsb → C	✓	✓	✓	0	–	✓	7
SETC	0	C ← 1	–	–	1	–	–	–	
CLRC	0	C ← 0	–	–	0	–	–	–	

Table 3-7C. Instruction Summary

Mnemonic	Operands	Operation (Note 1)	Flags						Notes
			Z	N	C	V	VT	ST	
CLRVT	0	VT ← 0	–	–	–	–	0	–	
RST	0	PC ← 2080H	0	0	0	0	0	0	8
DI	0	Disable All Interupts (I ← 0)	–	–	–	–	–	–	
EI	0	Enable All Interupts (I ← 1)	–	–	–	–	–	–	
DPTS	0	Disable all PTS Cycles (PSE = 0)	–	–	–	–	–	–	
EPTS	0	Enable all PTS Cycles (PSE = 1)	–	–	–	–	–	–	
NOP	0	PC ← PC + 1	–	–	–	–	–	–	
SKIP	0	PC ← PC + 2	–	–	–	–	–	–	
NORML	2	Left shift till msb = 1; D ← shift count	✓	✓	0	–	–	–	7
TRAP	0	SP ← SP – 2; (SP) ← PC; PC ← (2010H)	–	–	–	–	–	–	9

PUSHA	1	SP ← SP-2; (SP) ← PSW;
		PSW ← 0000H; SP ← SP-2;
		(SP) ← IMASK1/WSR; IMASK1 ← 00H
			0	0	0	0	0	0	

POPA	1	IMASK1/WSR ← (SP); SP ← SP+2 PSW ← (SP); SP ← SP+2	✓	✓	✓	✓	✓	✓	

IDLPD	1	IDLE MODE IF KEY=1;
		POWERDOWN MODE IF KEY =2;
		CHIP RESET OTHERWISE
			–	–	–	–	–	–	

CMPL	2	D-A	✓	✓	✓	✓	↑	–	
BMOV, BMOVi	2	[PTR_HI]+ ← [PTR_LOW]+ ; UNTIL COUNT=0	–	–	–	–	–	–	

NOTES:

1. If the mnemonic ends in "B" a byte operation is performed, otherwise a word operation is done. Operands D, B, and A must conform to the alignment rules for the required operand type. D and B are locations in the Register File; A can be located anywhere in memory.
2. D,D + 2 are consecutive WORDS in memory; D is DOUBLE-WORD aligned.
3. D,D + 1 are consecutive BYTES in memory; D is WORD aligned.
4. Changes a byte to word.
5. Offset is a 2's complement number.
6. Specified bit is one of the 2048 bits in the register file.
7. The "L" (Long) suffix indicates double-word operation.
8. Initiates a Reset by pulling $\overline{\text{RESET}}$ low. Software should re-initialize all the necessary registers with code starting at 2080H.
9. The assembler will not accept this mnemonic.

Table 3-7D. Instruction Length/Opcode

MNEMONIC	DIRECT	IMMED	INDIRECT		INDEXED	
			NORMAL*(1)	A-INC*(1)	SHORT*(1)	LONG*(1)
ADD (3-op)	4/44	5/45	4/46	4/46	5/47	6/47
SUB (3-op)	4/48	5/49	4/4A	4/4A	5/4B	6/4B
ADD (2-op)	3/64	4/65	3/66	3/66	4/67	5/67
SUB (2-op)	3/68	4/69	3/6A	3/6A	4/6B	5/6B
ADDC	3/A4	4/A5	3/A6	3/A6	4/A7	5/A7
SUBC	3/A8	4/A9	3/AA	3/AA	4/AB	5/AB
CMP	3/88	4/89	3/AB	3/AB	4/8B	5/8B
ADDB (3-op)	4/54	4/55	4/56	4/56	5/57	6/57
SUBB (3-op)	4/58	4/59	4/5A	4/5A	5/5B	6/5B
ADDB (2-op)	3/74	3/75	3/76	3/76	4/77	5/77
SUBB (2-op)	3/78	3/79	3/7A	3/7A	4/7B	5/7B
ADDCB	3/B4	3/B5	3/B6	3/B6	4/B7	5/B7
SUBCB	3/B8	3/B9	3/BA	3/BA	4/BB	5/BB
CMPB	3/98	3/99	3/9A	3/9A	4/9B	5/9B
MUL (3-op)	5/(2)	6/(2)	5/(2)	5/(2)	6/(2)	7/(2)
MULU (3-op)	4/4C	5/4D	4/4E	4/4E	5/4F	6/4F
MUL (2-op)	4/(2)	5/(2)	4/(2)	4/(2)	5/(2)	6/(2)
MULU (2-op)	3/6C	4/6D	3/6E	3/6E	4/6F	5/6F
DIV	4/(2)	5/(2)	4/(2)	4/(2)	5/(2)	6/(2)
DIVU	3/8C	4/8D	3/8E	3/8E	4/8F	5/8F
MULB (3-op)	5/(2)	5/(2)	5/(2)	5/(2)	6/(2)	7/(2)
MULUB (3-op)	4/5C	4/5D	4/5E	4/5E	5/5F	6/5F
MULB (2-op)	4/(2)	4/(2)	4/(2)	4/(2)	5/(2)	6/(2)
MULUB (2-op)	3/7C	3/7D	3/7E	3/7E	4/7F	5/7F
DIVB	4/(2)	4/(2)	4/(2)	4/(2)	5/(2)	6/(2)
DIVUB	3/9C	3/9D	3/9E	3/9E	4/9F	5/9F
AND (3-op)	4/40	5/41	4/42	4/42	5/43	6/43
AND (2-op)	3/60	4/61	3/62	3/62	4/63	5/63
OR (2-op)	3/80	4/81	3/82	3/82	4/83	5/83
XOR	3/84	4/85	3/86	3/86	4/87	5/87
ANDB (3-op)	4/50	4/51	4/52	4/52	5/53	5/53
ANDB (2-op)	3/70	3/71	3/72	3/72	4/73	4/73
ORB (2-op)	3/90	3/91	3/92	3/92	4/93	5/93
XORB	3/94	3/95	3/96	3/96	4/97	5/97
PUSH	2/C8	3/C9	2/CA	2/CA	3/CB	4/CB
POP	2/CC	—	2/CE	2/CE	3/CF	4/CF

NOTES:

1. Indirect and indirect + share the same opcodes, as do short and long indexed opcodes. If the second byte is even, use indirect or short indexed. If odd, use indirect or long indexed.
2. The opcodes for signed multiply and divide are the unsigned opcode with an "FE" prefix.

Table 3-7E. Instruction Length (in bytes)/Opcode

MNEMONIC	DIRECT	IMMED	INDIRECT		INDEXED	
			NORMAL	A-INC	SHORT	LONG
LD	3/A0	4/A1	3/A2	3/A2	4/A3	5/A3
LDB	3/B0	3/B1	3/B2	3/B2	4/B3	5/B3
ST	3/C0	—	3/C2	3/C2	4/C3	5/C3
STB	3/C4	—	3/C6	3/C6	4/C7	5/C7
XCH	3/04	—	—	—	4/0B	5/0B
XCHB	3/14	—	—	—	4/1B	5/1B
LDBSE	3/BC	3/BD	3/BE	3/BE	4/BF	5/BF
LBSZE	3/AC	3/AD	3/AE	3/AE	4/AF	5/AF

Mnemonic	Length/Opcode
PUSHF	1/F2
POPF	1/F3
PUSHA	1/F4
POPA	1/F5
TRAP	1/F7
LCALL	3/EF
SCALL	2/28-2F[3]
RET	1/F0
LJMP	3/E7
SJMP	2/20-27[3]
BR[ ]	2/E3
TIJMP	4/E2
JNST	1/D0
JST	1/D8
JNH	1/D1
JH	1/D9
JGT	1/D2
JLE	1/DA
JNC	1/B3
JC	1/D8
JNVT	1/D4
JVT	1/DC
JNV	1/D5
JV	1/DD
JGE	1/D6
JLT	1/DE
JNE	1/D7
JE	1/DF
JBC	3/30-37
JBS	3/38-3F

Mnemonic	Length/Opcode
DJNZ	3/E0
DJNZW	3/E1
NORML	3/0F
SHRL	3/0C
SHLL	3/0D
SHRAL	3/0E
SHR	3/08
SHRB	3/18
SHL	3/09
SHLB	3/19
SHRA	3/0A
SHRAB	3/1A
CLRC	1/F8
SETC	1/F9
DI	1/FA
EI	1/FB
DPTS	1/EC
EPTS	1/ED
CLRVT	1/FC
NOP	1/FD
RST	1/FF
SKIP	2/00
IDLPD	1/F6
BMOV	3/C1
BMOVi	3/CD

NOTES:

[3] The 3 least significant bits of the opcode are concatenated with the 8 bits to form an 11-bit, 2's complement offset.

Table 3-8A. Instruction Execution State Times[1]

MNEMONIC	DIRECT	IMMED	INDIRECT		INDEXED	
			NORMAL*	A-INC*	SHORT*	LONG*
ADD (3-op)	5	6	7/10	8/11	7/10	8/11
SUB (3-op)	5	6	7/10	8/11	7/10	8/11
ADD (2-op)	4	5	6/8	7/9	6/8	7/9
SUB (2-op)	4	5	6/8	7/9	6/8	7/9
ADDC	4	5	6/8	7/9	6/8	7/9
SUBC	4	5	6/8	7/9	6/8	7/9
CMP	4	5	6/8	7/9	6/8	7/9
ADDB (3-op)	5	5	7/10	8/11	7/10	8/11
SUBB (3-op)	5	5	7/10	8/11	7/10	8/11
ADDB (2-op)	4	4	6/8	7/9	6/8	7/9
SUBB (2-op)	4	4	6/8	7/9	6/8	7/9
ADDCB	4	4	6/8	7/9	6/8	7/9
SUBCB	4	4	6/8	7/9	6/8	7/9
CMPB	4	4	6/8	7/9	6/8	7/9
MUL (3-op)	16	17	18/21	19/22	19/22	20/23
MULU (3-op)	14	15	16/19	17/19	17/20	18/21
MUL (2-op)	16	17	18/21	19/22	19/22	20/23
MULU (2-op)	14	15	16/19	17/19	17/20	18/21
DIV	26	27	28/31	29/32	29/32	30/33
DIVU	24	25	26/29	27/30	27/30	28/31
MULB (3-op)	12	12	14/17	13/15	15/18	16/19
MULUB (3-op)	10	10	12/15	12/16	12/16	14/17
MULB (2-op)	12	12	14/17	15/18	15/18	16/19
MULUB (2-op)	10	10	12/15	13/15	12/16	14/17
DIVB	18	18	20/23	21/24	21/24	22/25
DIVUB	16	16	18/21	19/22	19/22	20/23
AND (3-op)	5	6	7/10	8/11	7/10	8/11
AND (2-op)	4	5	6/8	7/9	6/8	7/9
OR (2-op)	4	5	6/8	7/9	6/8	7/9
XOR	4	5	6/8	7/9	6/8	7/9
ANDB (3-op)	5	5	7/10	8/11	7/10	8/11
ANDB (2-op)	4	4	6/8	7/9	6/8	7/9
ORB (2-op)	4	4	6/8	7/9	6/8	7/9
XORB	4	4	6/8	7/9	6/8	7/9
LD/LDB	4	5	5/8	6/8	6/9	7/10
ST/STB	4	5	5/8	6/9	6/9	7/10
XCH/XCHB	5	—	—	—	8/13	9/14
LDBSE	4	4	5/8	6/8	6/9	7/10
LDBZE	4	4	5/8	6/8	6/9	7/10
BMOV	6+8 per word +3 for each memory controller reference					
BMOVi	7+8 per word + 14 for each interrupt +3 for each memory controller reference					
PUSH (int stack)	6	7	9/12	10/13	10/13	11/14
POP (int stack)	8	—	10/12	11/13	11/13	12/14
PUSH (ext stack)	8	9	11/14	12/15	12/15	13/16
POP (ext stack)	11	—	13/15	14/16	14/16	15/17

*Times for operands addressed as SFRs and internal RAM (0-1FFH)/memory controller references (200–0FFFFH)).

NOTE:

1. Execution times for memory controller references may be one to two states higher depending on the number of bytes in the prefetch queue.
2. INT stack is 0–1FFH and EXT stack is 200–0FFFFH.

Table 3-8B. Instruction Execution State Times

MNEMONIC		MNEMONIC	
PUSHF (int stack)	6	PUSHF (ext stack)	8
POPF (int stack)	7	POPF (ext stack)	10
PUSHA (int stack)	12	PUSHA (ext stack)	18
POPA (int stack)	12	POPA (ext stack)	18
TRAP (int stack)	16	TRAP (ext stack)	18
LCALL (int stack)	11	LCALL (ext stack)	13
SCALL (int stack)	11	SCALL (ext stack)	13
RET (int stack)	11	RET (ext stack)	14
CMPL	7	DEC/DECB	3
CLR/CLRB	3	EXT/EXTB	4
NOT/NOTB	3	INC/INCB	3
NEG/NEGB	3		
LJMP	7		
SJMP	7		
BR [indirect]	7		
TIJMP	15 + 3 for each memory controller reference		
JNST, JST	4/8 jump not taken/jump taken		
JNH, JH	4/8 jump not taken/jump taken		
JGT, JLE	4/8 jump not taken/jump taken		
JNC, JC	4/8 jump not taken/jump taken		
JNVT, JVT	4/8 jump not taken/jump taken		
JNV, JV	4/8 jump not taken/jump taken		
JGE, JLT	4/8 jump not taken/jump taken		
JNE, JE	4/8 jump not taken/jump taken		
JBC, JBS	5/9 jump not taken/jump taken		
DJNZ	5/9 jump not taken/jump taken		
DJNZW	6/10 jump not taken/jump taken		
NORML	8 + 1 per shift (9 for 0 shift)		
SHRL	7+ 1 per shift (8 for 0 shift)		
SHLL	7 + 1 per shift (8 for 0 shift)		
SHRAL	7 + 1 per shift (8 for 0 shift)		
SHR/SHRB	6 + 1 per shift (7 for 0 shift)		
SHL/SHLB	6 + 1 per shift (7 for 0 shift)		
SHRA/SHRAB	6 + 1 per shift (7 for 0 shift)		
CLRC	2		
SETC	2		
DI	2		
EI	2		
DPTS	2		
EPTS	2		
CLRVT	2		
NOP	2		
RST	20 (includes fetch of configuration byte)		
SKIP	3		
IDLPD	8/25 (proper key/improper key)		

Table 3-8C. Instruction Execution State Times

PTS CYCLES	
Single Transfer	18 (+ 3 for each memory controller reference)
Block Transfer	13 (+ 7 for each transfer, 1 minimum + 3 for each memory controller reference)
A/D Mode (SFRs/internal RAM)	21
(MEMORY CONT)	25
HSI MODE (SFRs/internal RAM)	12 (+ 10 for each transfer, 1 minimum)
(MEMORY CONT)	16 (+ 10 for each transfer, 1 minimum)
HSO MODE (SFRs/internal RAM)	11 (+ 10 for each transfer, 1 minimum)
(MEMORY CONT)	15 (+ 11 for each transfer, 1 minimum)

3.6 80C196KC Instruction Set Additions

For users already familiar with the 80C196KB, there are six instructions added to the standard MCS-96 instruction set to form the 80C196KC instruction set. All of the former instructions perform the same function. The new instructions and their descriptions are listed below:

DPTS Disables the Peripheral Transaction Server by clearing the PSE flag in the PSW

EPTS Enables the Peripheral Transaction Server by setting the PSE flag in the PSW

BMOVI Block move using 2 auto-incrementing pointers and a counter which can be interrupted. BMOV with interrupt. BMOVI updates the counter when interrupted

XCH/XCHB Exchanges the contents of two registers or a register and a memory location

TIJMP Jumps to an address selected out of a table of addresses. The table may have a maximum 128 entries

See Figure 3-9 for TIJMP address calculation.

```
        TIJMP   TBASE, [INDEX], INDEX_MASK

        TBASE = WORD REGISTER CONTAINING 16-BIT ADDRESS
                OF BEGINNING OF JUMP TABLE

        INDEX = WORD REGISTER CONTAINING 16-BIT ADDRESS
                OF 8-BIT INDEX INTO JUMP TABLE

   INDEX_MASK = 8-BIT IMMEDIATE DATA TO MASK (AND)
                WITH INDEX
```

ADDRESS CALCULATION

```
   [INDEX] AND INDEX_MASK = OFFSET
           (2 * OFFSET) + TBASE = DEST X
```

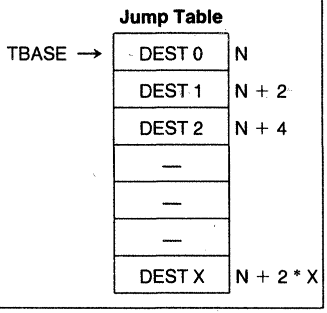


Figure 3-9. TIJMP Address Calculation

3.7 Software Standards and Conventions

For a software project of any size it is a good idea to modularize the program and to establish standards which control the communication between these modules. The nature of these standards will vary with the needs of the final application. A common component of all of these standards, however, must be the mechanism for passing parameters to procedures and returning results from procedures. In the absence of some overriding consideration which prevents their use, it is suggested that the user conform to the conventions adopted by the PLM-96 programming language for procedure linkage. It is a very usable standard for both the assembly language and PLM-96 environment and it offers compatibility between these environments. Another advantage is that it allows the user access to the same floating point arithmetics library that PLM-96 uses to operate on REAL variables.

REGISTER UTILIZATION

The MCS-96 architecture provides a 256 byte register file. Some of these registers are used to control register-mapped I/O devices and for other special functions such as the ZERO register and the stack pointer. The remaining bytes in the register file, some 230 of them, plus the extra 256 bytes of RAM, are available for allocation by the programmer. If these registers are to be used effectively, some overall strategy for their allocation must be adopted. PLM-96 adopts the simple and effective strategy of allocating the eight bytes between addresses 1CH and 23H as temporary storage. The starting address of this region is called PLMREG. The remaining area in the register file is treated as a segment of memory which is allocated as required.

ADDRESSING 32-BIT OPERANDS

These operands are formed from two adjacent 16-bit words in memory. The least significant word of the double word is always in lower address, even when the data is in the stack (which means that the most significant word must be pushed into the stack first). A double word is addressed by the address of its least significant byte. Note that the hardware supports some operations on double words.

SUBROUTINE LINKAGE

Parameters are passed to subroutines in the stack. Parameters are pushed into the stack in the order that they are encountered in the scanning of the source text. Eight-bit parameters (BYTES or SHORT-INTEGERS) are pushed into the stack with the high order byte undefined. Thirty-two bit parameters (LONG-INTEGERS, DOUBLE-WORDS, and REALS) are pushed onto the stack as two 16-bit values; the most significant half of the parameter is pushed into the stack first.

As an example, consider the following PLM-96 procedure:

```
example_procedure: PROCEDURE
(param1,param2,param3);
  DECLARE param1 BYTE,
          param2 DWORD,
          param3 WORD;
```

When this procedure is entered at run time the stack will contain the parameters in the following order:

?????? : param1	
high word of param2	
low word of param2	
param3	
return address	← Stack_pointer

Figure 3-9. Stack Image

If a procedure returns a value to the calling code (as opposed to modifying more global variables) then the result is returned in the variable PLMREG. PLMREG is viewed as either an 8-, 16- or 32-bit variable depending on the type of the procedure.

The standard calling convention adopted by PLM-96 has several key features:

a) Procedures can always assume that the eight bytes of register file memory starting at PLMREG can be used as temporaries within the body of the procedure.

b) Code which calls a procedure must assume that the eight bytes of register file memory starting at PLMREG are modified by the procedure.

c) The Program Status Word (PSW—see Section 3.4) is not saved and restored by procedures so the calling code must assume that the condition flags (Z, N, V, VT, C, and ST) are modified by the procedure.

d) Function results from procedures are always returned in the variable PLMREG.

PLM-96 allows the definition of INTERRUPT procedures which are executed when a predefined interrupt occurs. These procedures do not conform to the rules of a normal procedure. Parameters cannot be passed to these procedures and they cannot return results. Since they can execute essentially at any time (hence the term interrupt), these procedures must save the PSW and PLMREG when they are entered and restore these values before they exit.

3.8 Software Protection Hints

Several features to assist in recovery from hardware and software errors are available on the 80C196KC. Protection is also provided against executing unimplemented opcodes by the unimplemented opcode interrupt. In addition, the hardware reset instruction (RST) can cause a reset if the program counter goes out of bounds. This instruction has an opcode of 0FFH, so if the processor reads in bus lines which have been pulled high it will reset itself.

The Watchdog Timer (WDT) further helps protect against software and hardware errors. When using the WDT to protect software it is desirable to reset it from only one place in code, lessening the chance of an undesired WDT reset. The section of code that resets the WDT should monitor the other code sections for proper operation. This can be done by checking variables to make sure they are within reasonable values. Simply using a software timer to reset the WDT every 10 milliseconds will provide protection only for catastrophic failures.

4.0 PERIPHERAL OVERVIEW

There are five major peripherals on the 80C196KC: the Pulse-Width-Modulated outputs (PWM), Timer1 and Timer2, High Speed I/O Unit, Serial Port, and A/D Converter. A minor peripheral is the watchdog timer. With the exception of the high speed I/0 unit (HSIO), each of the peripherals is a single unit that can be discussed without further separation.

Four individual sections make up the HSIO and work together to form a very flexible timer/counter based I/0 system. Included in the HSIO are a 16-bit timer (Timer1), a 16-bit up/down counter (Timer2), a programmable high speed input unit (HSI), and a programmable high speed output unit (HSO). With very little CPU overhead the HSIO can measure pulse widths, generate waveforms, and create periodic interrupts. Depending on the application, the HSI/0 can perform the work of up to 18 timer/counters and capture/compare registers.

A brief description of the peripheral functions and interactions is included in this section. All of the details on control bits and precautions are in the individual sections for each peripheral starting with Section 7.

4.1 Pulse Width Modulation Output (D/A)

Digital to analog conversion can be done with the Pulse Width Modulation output. The 80C196KC has 3 PWM outputs, like the 1 PWM on the 80C196KB.

The output waveform is a variable duty cycle pulse which is selectable to repeat every 256 state times or 512 state times. Changes in the duty cycle are made by writing to the PWM registers. Several types of motors require a PWM waveform for most efficient operation. Additionally, if this waveform is filtered it will produce a DC level which can be changed in 256 steps by varying the duty cycle. Details on the PWM are in Section 7.0.

4.2 Timer1 and Timer2

Two 16-bit timers are available for use on the 80C196KC. The first is designated "Timer1", the second "Timer2". The timers are the time bases for the HSI and HSO units and can be considered an integral part of the HSI/O. Details on the Timers are in Section 8.0.

Timer1

Timer1 is a free-running timer which is incremented every eight state times. It can be read and written, but care must be taken when writing to it if the HSIO Subsystem is being used. The precautions necessary when writing to Timer1 are described in Section 8. Timer1 can cause an interrupt when it overflows.

Timer2

Timer2 counts transitions, both positive and negative, on its input which can be either the T2CLK pin or the HSI.1 pin. Also, the 80C196KC has added the capability to clock Timer2 internally every 1 or 8 state times. Timer2 can be read and written and can be reset by hardware, software or the HSO unit. It can be configured to count up or down based on Port 2.6 and it's value can be captured into the T2CAPture register on a rising edge on Port 2.7.

When clocking Timer2 externally, the maximum input transition speed is once every 8 state times or once per state time in the Fast Increment mode.

4.3 High Speed Inputs (HSI)

The High Speed Input (HSI) unit can capture the value of Timer1 when an event takes place on one of four input pins (HSI.0-HSI.3). Four types of events can trigger a capture: rising edges only, falling edges only, rising or falling edges, or every eighth rising edge. Each HSI pin can be independently programmed to look for any of these conditions. A block diagram of this unit is shown in Figure 4-1.

When events occur, the Timer1 value gets stored in the FIFO along with 4 status bits which indicate the input line(s) that caused the event. The next event ready to be unloaded from the FIFO is placed in the HSI Holding Register, so a total of 8 pieces of data can be stored in the FIFO. Data is taken off the FIFO by reading the HSI__STATUS register, followed by reading the HSI__TIME register. When the time register is read the next FIFO location is loaded into the holding register.

Independent of the HSI operation, the state of the HSI pins is indicated by 4 bits of the HSI__STATUS register so the pins can also be inputs. Also independent of the HSI operation is the HSI.0 pin interrupt, which can be used as an extra external interrupt. Details on the HSI are in Section 9.0

4.4 High Speed Outputs (HSO)

The High Speed Output (HSO) unit can generate events at specified values of Timer1 or Timer2 with minimal CPU overhead. A block diagram of the HSO unit is shown in Figure 4-2. Up to 8 pending events can be stored in the CAM (Content Addressable Memory) of the HSO unit at one time. Commands are placed into the HSO unit by first writing to HSO__COMMAND with the event to occur, and then to HSO__TIME with the timer match value.

Fourteen different types of events can be triggered by the HSO: 8 external and 7 internal. There are two interrupt vectors associated with the HSO, one for external events, and one for internal events. External events consist of switching one or more of the 6 HSO lines (HSO.0-HSO.5). HSO.4 and HSO.5 share pins with HSI.2 and HSI.3 and it is possible to have these pins enabled for both functions. Internal events include setting up 4 Software Timers, resetting Timer2, and starting an A/D conversion. The software timers are flags that can be set by the HSO and optionally cause interrupts.

4.5 Serial Port

The serial port on the 80C196KC is functionally compatible with the serial port on the MCS-51 and MCS-96 families of microcontrollers. One synchronous and three asynchronous modes are available. The asynchronous modes are full duplex. Double buffering is provided for the receiver so a second byte can be received before the first byte has been read. The transmitter is also double buffered, allowing bytes to be written 2 at a time.

The Serial Port STATus (SP__STAT) register contains bits to indicate receive overrun, parity, and framing errors, and transmit and receive interrupts.

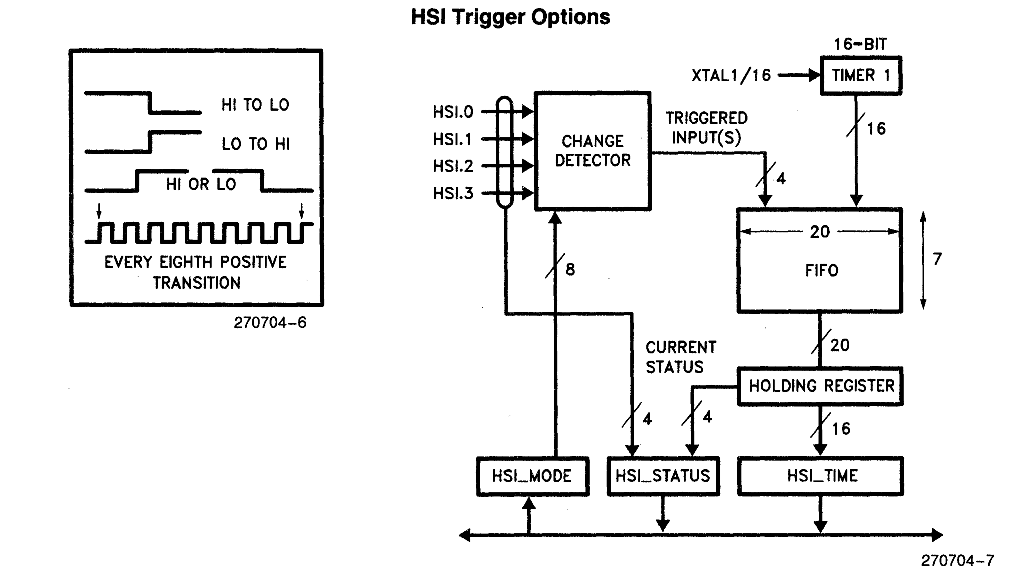


Figure 4-1. HSI Block Diagram

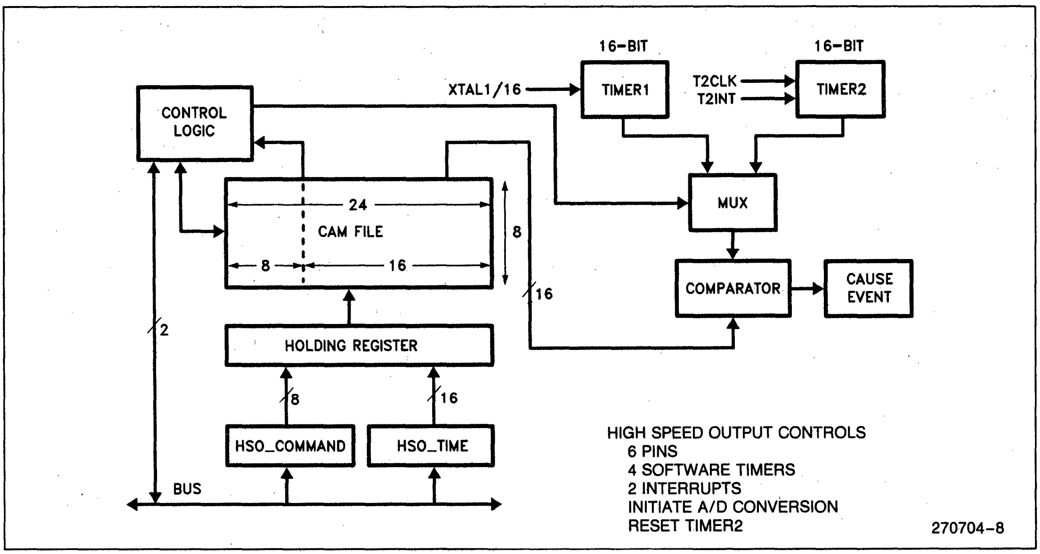


Figure 4-2. HSO Block Diagram

BAUD RATES

Baud rates are generated in an independent 15-bit counter based on either the T2CLK pin or XTAL1 pin. Common baud rates can be easily generated with standard crystal frequencies. A maximum baud rate of 1 Mbaud is available in the asynchronous modes with 16 MHz on XTAL1. The synchronous mode has a maximum rate of 4.0 Mbaud with a 16 MHz clock.

4.6 A/D Converter

The A/D Converter consists of a sample-and-hold, an 8-channel multiplexer, and a 8- or 10-bit successive approximation analog-to-digital converter.

Analog signals can be sampled by any of the 8 analog input pins (ACH0 through ACH7) which are shared with Port 0. An A/D conversion is performed on one input at a time using successive approximation with a result equal to the ratio of the input voltage divided by the analog supply voltage. If the ratio is 1.00, then the result will be all ones. A conversion can be started by writing to the A/D_COMMAND register or by an HSO command. For details on the A/D Converter, see Section 12.

4.7 I/O Ports

There are five 8-bit I/O ports on the 80C196KC. Some are input only, some are output only, some are bidirectional and some have multiple functions. In addition to these ports, the HSI/O lines can be used as standard I/O lines.

Port 0 is an input port which is also the analog input for the A/D converter. Port 1 is a quasi-bidirectional port. The three MSBs of Port 1 are multiplexed with the $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ functions. Also, the 2 extra PWM outputs are multiplexed on Port 1.4 and 1.3. Port 2 contains three types of pins: quasi-bidirectional, input and output. Its input and output lines are shared with other functions. Ports 3 and 4 are open-drain bidirectional ports which share their pins with the address/data bus.

Quasi-bidirectional pins can be used as input and output pins without the need for a data direction register. They output a strong low value and a weak high value. The weak high value can be externally pulled low providing an input function. A detailed explanation of these ports can be found in Section 13.

4.8 Watchdog Timer

The Watchdog Timer (WDT) provides a means to recover gracefully from a software upset. When the watchdog is enabled it will initiate a hardware reset unless the software clears it every 64K state times. Hardware resets on the 80C196KC pull the $\overline{\text{RESET}}$ pin low, providing a system reset to other components on the board.

5.0 INTERRUPTS

Twenty-eight (28) sources of interrupts are available on the 80C196KC. These sources are gathered into 15 vectors plus special vectors for NMI, the TRAP instruction, and Unimplemented Opcodes. Figure 5-1 shows the routing of the interrupt sources into their vectors as well as the control bits which enable some of the sources. The operation of the Peripheral Transaction Server (PTS) is intimately a part of interrupt operation and is discussed in Section 6.

Special Interrupts

Three special interrupts are available on the 80C196KC: NMI, TRAP and Unimplemented opcode. Although available for customer use, these interrupts may be used in Intel development tools or evaluation boards.

NMI

NMI, the external Non-Maskable Interrupt, is the highest priority interrupt. It vectors indirectly through location 203EH. For design symmetry, a mask bit exists in INT_MASK1 for the NMI. However, the bit does not function and will not stop an NMI from occurring. For future compatibility, the NMI mask bit must be set to zero.

NMI on the 8096BH vectored directly to location 0000H, so for the 80C196KC to be compatible with 8096BH software, which uses the NMI, location 203EH must be loaded with 0000H. The NMI interrupt vector and interrupt vector location is used by some Intel development tools. The NMI interrupt is rising edge triggered.

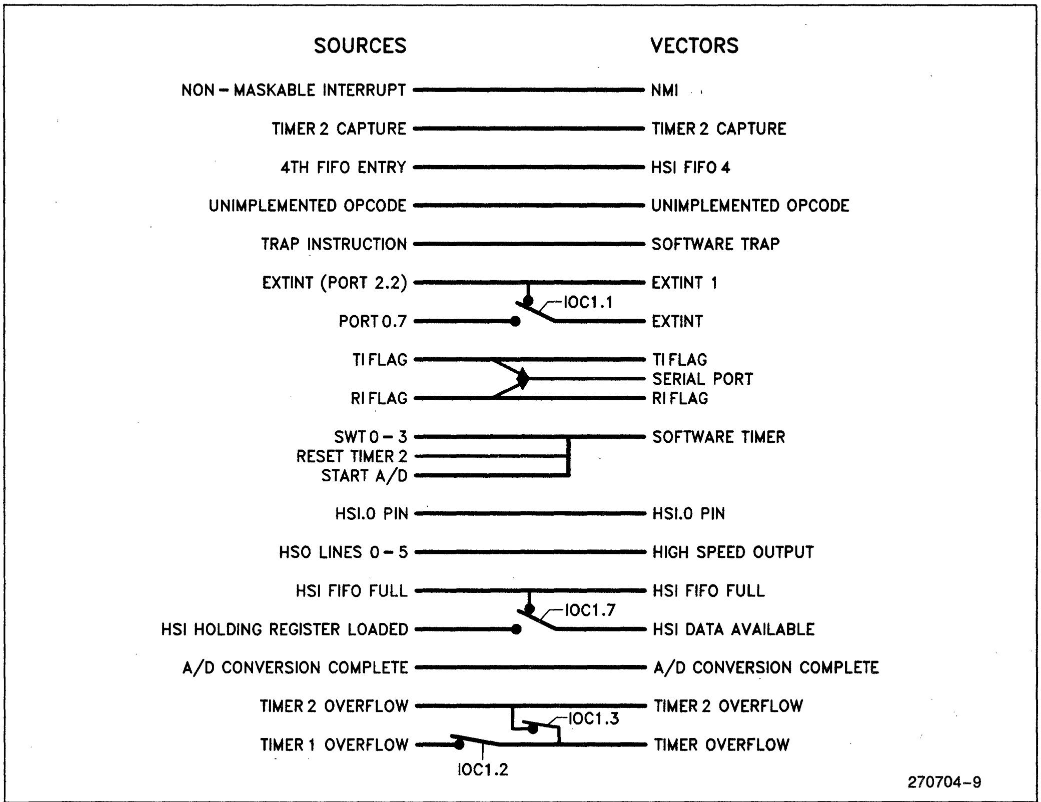


Figure 5-1. 80C196KC Interrupt Sources

TRAP

Opcode 0F7H, the TRAP instruction, causes an indirect vector through location 2010H. The TRAP instruction provides a single instruction interrupt useful in designing software debuggers. The TRAP instruction prevents the acknowledgement of interrupts until after execution of the next instruction.

Unimplemented Opcode

Opcodes which are not implemented on the 80C196KC will cause an indirect vector through location 2012H if executed.

The programmer must initialize the interrupt vector table with the starting addresses of the appropriate interrupt service routines. It is suggested that any unused interrupts be vectored to an error handling routine.

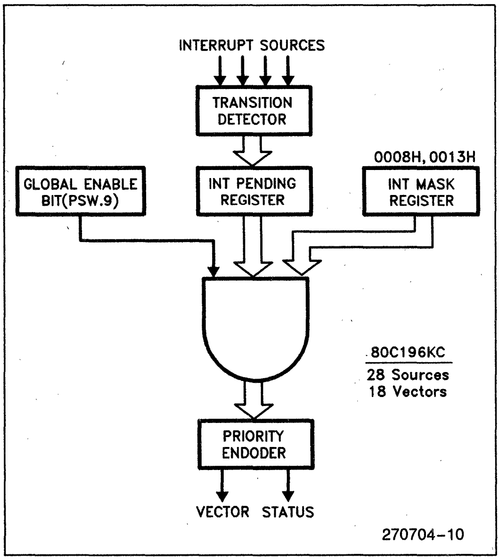


Figure 5-2. 80C196KC Interrupt Structure Block Diagram

Five registers control the operation of the interrupt system: INT_PEND, INT_PEND1, INT_MASK and INT_MASK1 and the PSW which contains a global disable bit. A block diagram of the system is shown in Figure 5-2. The transition detector looks for 0 to 1 transitions on any of the sources. External sources have a maximum transition speed of one edge every state time.

5.1 Interrupt Control

Interrupt Pending Register

When hardware detects one of the sixteen interrupts, it sets the corresponding bit in one of two pending interrupt registers (INT_PEND @ 09H and INT_PEND1 @ 12H). When the interrupt vector is taken, the pending bit is cleared. These registers, the formats of which are shown in Figure 5-3, can be read or modified as byte registers. They can be read to determine which of the interrupts are pending at any given time or modified to either clear pending interrupts or generate interrupts under software control. Any software which modifies the INT_PEND registers should ensure that the entire operation is inseparable. The easiest way to do this is to use the logical instructions in the two or three operand format, for example:

```
ANDB   INT_PEND,#11111101B
              ; Clears the A/D Interrupt
ORB    INT_PEND,#00000010B
              ; Sets the A/D Interrupt
```

Caution must be used when writing to the pending register to clear interrupts. If the interrupt has already been acknowledged when the bit is cleared, a 5 state time "partial" interrupt cycle will occur. This is because the 80C196KC will have to fetch the next instruction of the normal instruction flow, instead of proceeding with the interrupt processing. The effect on the program will be essentially that of an extra two NOPs. This can be prevented by clearing the bits using a 2 operand immediate logical, as the 80C196KC holds off acknowledging interrupts during these "read/modify/write" instructions.

Interrupt Mask Register

Individual interrupts can be enabled or disabled by setting or clearing bits in the interrupt mask registers (INT_MASK @ 08H and INT_MASK1 @ 13H). The format of these registers is the same as the Interrupt Pending Register shown in Figure 5-3.

The INT_MASK and INT_MASK1 registers can be read or written as byte registers. A one in any bit position will enable the corresponding interrupt source and a zero will disable the source. The hardware will save any interrupts that occur by setting bits in the pending register, even if the interrupt mask bit is cleared. The INT_MASK register is the lower eight bits of the PSW so the PUSHF and POPF instructions save and restore the INT_MASK register as well as the PSW. Both the INT_MASK and INT_MASK1 registers can be saved and restored with the PUSHA and POPA Instructions.

	7	6	5	4	3	2	1	0
12H IPEND1: 13H IMASK1:	NMI	FIFO FULL	EXT INT1	T2 OVF	T2 CAP	HSI4	RI	TI

	7	6	5	4	3	2	1	0
09H IPEND: 08H IMASK:	EXT INT	SER PORT	SOFT TIMER	HSI.0 PIN	HSO PIN	HSI DATA	A/D DONE	TIMER OVF

Figure 5-3. Interrupt Mask and Pending Registers

Global Disable

The processing of all interrupts except the NMI, TRAP and unimplemented opcode interrupts can be disabled by clearing the I bit in the PSW. Setting the I bit will enable interrupts that have mask register bits set. The I bit is controlled by the EI (Enable Interrupts) and DI (Disable Interrupts) instructions. Note that the I bit only controls the actual servicing of interrupts. Interrupts that occur when I is cleared will be held in the pending register and serviced on a prioritized basis when I is set.

5.2 Interrupt Priorities

The priority encoder looks at all of the interrupts which are both pending and enabled, and selects the one with the highest priority. The priorities are shown in Figure 5-4 (15 is highest, 0 is lowest). The interrupt generator then forces a call to the location in the indicated vector location. This location would be the starting location of the Interrupt Service Routine (ISR).

This priority selection controls the order in which pending interrupts are passed to the software via interrupt calls. The software can then implement its own priority structure by controlling the mask registers (INT_MASK and INT_MASK1). To see how this is done, consider the case of a serial I/O service routine which must run at a priority level which is lower than the HSI data available interrupt but higher than any other source. The "preamble" and exit code for this interrupt service routine would look like this:

```
serial_io_isr:
 PUSHA          ; Save the PSW, INT_MASK
                ; INT_MASK1, and WSR
 LDB    INT_MASK,#00000100B
 EI             ; Enable interrupts again
 ; ⎫
 ; ⎪
 ; ⎪
 ; ⎬      Service the interrupt
 ; ⎪
 ; ⎪
 ; ⎭
 POPA           ; Restore
 RET
```

Number	Source	Vector Location	Priority
INT15	NMI	203EH	15
INT14	HSI FIFO Full	203CH	14
INT13	EXTINT1	203AH	13
INT12	TIMER2 Overflow	2038H	12
INT11	TIMER2 Capture	2036H	11
INT10	4th Entry into HSI FIFO	2034H	10
INT09	RI	2032H	9
INT08	TI	2030H	8
SPECIAL	Unimplemented Opcode	2012H	N/A
SPECIAL	Trap	2010H	N/A
INT07	EXTINT	200EH	7
INT06	Serial Port	200CH	6
INT05	Software Timer	200AH	5
INT04	HSI.0 Pin	2008H	4
INT03	High Speed Outputs	2006H	3
INT02	HSI Data Available	2004H	2
INT01	A/D Conversion Complete	2002H	1
INT00	Timer Overflow	2000H	0

Figure 5-4. 80C196KC Interrupt Priorities

Note that location 200CH in the interrupt vector table would have to be loaded with the label serial_io_isr and the interrupt be enabled for this routine to execute.

There is an interesting chain of instruction side-effects which makes this (or any other) 80C196KC interrupt service routine execute properly:

a) After the Interrupt controller decides to process an interrupt, it executes a "CALL", using the location from the corresponding interrupt vector table entry as the destination. The return address is pushed onto the stack. Another interrupt cannot be serviced until after the first instruction following the interrupt call is executed.

b) The PUSHA instruction, which is guaranteed to execute, saves the PSW, INT__MASK, INT__MASK1, and the WSR on the stack as two words, and clears them. An interrupt cannot be serviced immediately following a PUSHA instruction. (If INT__MASK1 and the WSR register are not used, or 8096BH code is being executed, PUSHF, which saves only the PSW and INT__MASK, can be used in place of PUSHA).

c) LD INT__MASK, which is guaranteed to execute, enables those interrupts that are allowed to interrupt this ISR. This allows the software to establish its own priorities independent of the hardware.

d) The EI instruction reenables the processing of interrupts with the new priorities.

e) At the end of the ISR, the POPA instruction restores the PSW, INT__MASK, INT__MASK1, and WSR to their original state when the interrupt occurred. Interrupts cannot occur immediately following a POPA instruction so the RET instruction is guaranteed to execute. This prevents the stack from overflowing if interrupts are occurring at high frequency. (If INT__MASK1 and the WSR are not being used, or 8096BH code is being executed, POPF, which restores only the PSW and INT__MASK, can be used in place of POPA.

Notice that the "preamble" and exit code for the interrupt service routine does not include any code for saving or restoring registers. This is because it has been assumed that the interrupt service routine has been allocated its own private set of registers from the onboard register file. The availability of some 230 bytes of register storage makes this quite practical. By using the VWindowing Scheme, an additional 256 bytes become available (See Section 3.4).

5.3 Critical Regions

Interrupt service routines sometimes must share data with other routines. Whenever the programmer is coding those sections of a program which access these shared pieces of data, great care must be taken to ensure that the integrity of the data is maintained. Consider clearing a bit in the interrupt pending register as part of a non-interrupt routine:

```
LDB         AL,INT_PEND
ANDB        AL,#bit_mask
STB         AL,INT_PEND
```

This code works if no other routines are operating concurrently, but can cause occasional and serious problems if used in a concurrent environment. (All programs which make use of interrupts must be considered a concurrent environment.) For example, assume that the INT__PEND register contains 00001111B and bit 3 (HSO event interrupt pending) is to be reset. The code does work for this data pattern but what happens if an HSI interrupt occurs somewhere between the LDB and the STB instructions? Before the LDB instruction INT__PEND contains 00001111B and after the LDB instruction so does AL. If the HSI interrupt service routine executes at this point then INT__PEND will change to 00001011B. The ANDB changes AL to 00000111B and the STB changes INT__PEND to 00000111B. It should be 00000011B. This code sequence has managed to generate a false HSI interrupt. These problems can be avoided by assuring mutual exclusion which means that if more than one routine can change a variable, then the programmer must ensure exclusive access to the variable during the entire operation on the variable.

In many cases the instruction set of the 80C196KC allows the variable to be modified with a single instruction. The code in the above example can be implemented with a single instruction.

```
ANDB        INT_PEND,#bit_mask
```

Instructions are indivisible so mutual exclusion is ensured in this case. Changes to the INT__PEND or INT__PEND1 register must be made as a single instruction, since bits can be changed in this register even if interrupts are disabled. Depending on system configurations, several other SFRs might also need to be changed in a single instruction for the same reason.

When variables must be modified without interruption, and a single instruction can not be used, the programmer must create what is termed a critical region. One way to do this is to simply disable interrupts with a DI instruction, perform the modification, and then re-enable interrupts with an EI instruction. The problem with this approach is that it leaves the interrupts enabled even if they were not enabled at the start. A better solution is to enter the critical region with a PUSHF instruction which saves the PSW and also clears the interrupt enable flag. The region can then be terminated with a POPF instruction which returns the interrupt enable to the state it was in before the code sequence. It should be noted that some system configurations might require more protection to form a critical region. An example is a system in which more than one processor has access to a common resource such as memory or external I/O devices.

5.4 Interrupt Timing

The 80C196KC can be interrupted from four different external sources; NMI, P2.2, HSI.0 and P0.7. All external interrupts are sampled during PH1 or CLKOUT low and are latched internally. Holding levels on external interrupts for at least one state time will ensure recognition of the interrupts.

The external interrupts on the 80C196KC, although sampled during PH1, are edge triggered interrupts as opposed to level triggered.

Interrupts are not always acknowledged immediately. If the interrupt signal does not occur prior to 4 statetimes before the end of an instruction, the interrupt may not be acknowledged until after the next instruction has been executed. This is because an instruction is fetched and prepared for execution a few state times before it is actually executed.

There are 6 instructions which always inhibit interrupts from being acknowledged until after the next instruction has been executed. These instructions are:

EI, DI — Enable and disable all interrupts by toggling the global disable bit (PSW.9).

PUSHF — PUSH Flags pushes the PSW/INT_MASK pair then clears it, leaving both INT_MASK and PSW.9 clear.

POPF — POP Flags pops the PSW/INT_MASK pair off the stack

PUSHA — PUSH All does a PUSHF, then pushes the INT_MASK1/WSR pair and clears INT_MASK1

POPA — POP All pops the INT_MASK1/WSR pair and then does a POPF

Interrupts can also not occur immediately after execution of:

Unimplemented Opcodes

TRAP — The software trap instruction

SIGND — The signed prefix for multiply and divide instructions

When an interrupt is acknowledged the interrupt pending bit is cleared, and a call is forced to the location indicated by the specified interrupt vector. This call occurs after the completion of the instruction in process, except as noted above. The procedure of getting the vector and forcing the call requires 16 state times. If the stack is in external RAM an additional 2 state times are required. This assumes a 0 wait-state bus.

The maximum number of state times required from the time an interrupt is generated (not acknowledged) until the 80C196KC begins executing code at the desired location is the time of the longest instruction, NORML (Normalize — 39 state times), plus the 4 state times prior to the end of the previous instruction, plus the response time (16(internal stack) or 18(external stack) state times). Therefore, the maximum response time is 61 (39 + 4 + 18) state times. This does not include the 6 state times required for PUSHF if it is used as the first instruction in the interrupt routine or additional latency caused by having the interrupt masked or disabled. Refer to Figure 5-5, Interrupt Response Time, to visualize an example of worst case scenario.

Interrupt latency time can be reduced by careful selection of instructions in areas of code where interrupts are expected. Using 'EI' followed immediately by a long instruction (e.g. MUL, NORML, etc.) will increase the maximum latency by 4 state times, as an interrupt cannot occur between EI and the instruction following EI. The DI, PUSHF, POPF, PUSHA, POPA and TRAP instructions will also cause the same situation. Typically these instructions would only effect latency when one interrupt routine is already in process, as these instructions are seldom used at other times.

5.5 Interrupt Summary

Many of the interrupt vectors on the 8096BH were shared by multiple interrupts. The interrupts which were shared on the 8096BH are: Transmit Interrupt, Receive Interrupt, HSI FIFO Full, Timer2 Overflow and P2.2. On the 80C196KC, each of these interrupts have their own interrupt vectors. The source of the interrupt vectors are typically programmed through control registers. These registers can be read in HWindow 15 to determine the source of any interrupt. Interrupt sources with two possible interrupt vectors, serial receive interrupt sharing serial port and receive interrupt vectors for example, should be configured for only one interrupt vector.

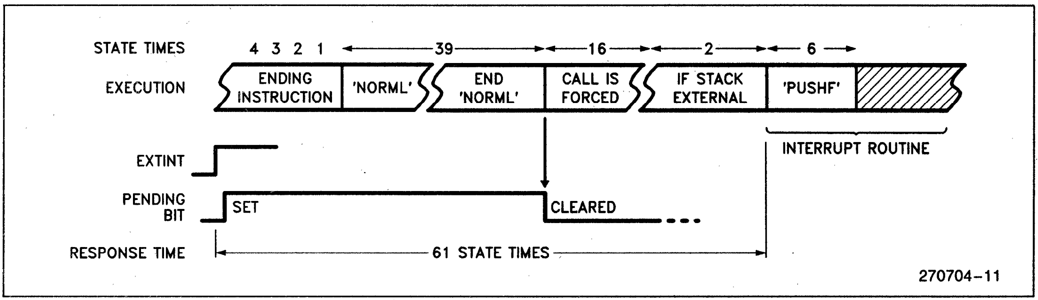


Figure 5-5. Interrupt Response Time

Interrupts with separate vectors include: NMI, TRAP, Unimplemented Opcode, Timer2 Capture, 4th Entry into HSI FIFO, Software timer, HSI.0 Pin, High Speed Outputs, and A/D conversion Complete. The NMI, TRAP and Unimplemented Opcode interrupts were covered in Section 5.1.

EXTINT and P0.7

The 80C196KC has two external interrupt vectors; EXTINT (200EH) and EXTINT1 (203AH). The EXTINT vector has two alternate sources selectable by IOC1.1, the external interrupt pin (Port 2.2) and Port 0.7. The external interrupt pin is the only source for the EXTINT1 interrupt vector. The external interrupt pin should not be programmed to interrupt through both vectors. Both external interrupt sources are rising edge triggered.

Serial Port Interrupts

The serial port generates one of three possible interrupts: Transmit interrupt TI(2030H), Receive Interrupt RI(2032H) and SERIAL(200CH). Refer to section 10 for information on the serial port interrupts. The 8096BH shared the TI and RI interrupts on the SERIAL interrupt vector. On the 80C196KC, these interrupts share both the serial interrupt vector and have their own interrupt vectors. Ideally, the transmit and receive interrupts should be programmed as separate interrupt vectors while disabling the SERIAL interrupt. For 8096BH compatibility, the interrupts can still use the SERIAL interrupt vector.

HSI FIFO FULL and HSI DATA AVAILABLE

HSI FIFO FULL and HSI DATA AVAILABLE interrupts shared the HSI DATA AVAILABLE interrupt vector on the 8096BH. The source of the HSI DATA AVAILABLE interrupt is controlled by the setting of I/O Control Register 1,(IOC1.7). Setting IOC1.7 to zero will generate an interrupt when a time value is loaded into the holding register. Setting the bit to one generates an interrupt when the FIFO, independent of the holding register, has six entries in it.

On the 80C196KC, separate interrupt vectors are available for the HSI FIFO FULL(203CH) and HSI DATA AVAILABLE(2004H) interrupts. The interrupts should be programmed for separate vector locations. Refer to Section 9 for more information on the High Speed Inputs.

HSI FIFO_4

The HSI FIFO can generate an interrupt when the HSI has four or more entries in the FIFO. The HSI FIFO_4 interrupt vectors through location 2034H. Refer to Section 9 for more information on the High Speed Inputs.

HSI.0 External Interrupt

A rising edge on HSI.0 pin can be used as an external interrupt. Sampling is guaranteed if the pin is held for at least one state time. The interrupt vectors through location 2008H. The pin does not have to be enabled to the FIFO to cause an interrupt.

Timer2 and Timer1 Overflow

Timer2 and Timer1 can interrupt on overflow. These interrupts shared the same interrupt vector TIMER OVERFLOW(2000H) on the 8096BH. The interrupts are individually enabled by setting bits 2 and 3 in IOC1. Which timer actually caused the interrupt can be determined by bits .4 and 5 of IOS1. On the 80C196KC Timer2 overflow (0H or 8000H) has a separate interrupt vector through location 2038H.

Timer2 Capture

The 80C196KC can interrupt in response to a Timer2 capture triggered by a rising edge on P2.7. Timer2 Capture vectors through location 2036H.

High Speed Output

The High Speed Output interrupt can be generated in response to a programmed HSO command which causes an external event. HSO commands which set or clear the High Speed Output pins are considered external events. Status Register IOS2 indicates which HSO events have occured and can be used to arbitrate which HSO command caused the interrupt. The High Speed Output interrupt vectors indirectly through location 2006H. For more information on High Speed Outputs, refer to Section 10.

Software Timers

HSO commands which create internal events can interrupt through the Software Timer interrupt vector. Internal events include triggering an A/D conversion, resetting Timer2 and software timers. Status registers IOS2 and IOS1 can be used to determine which internal HSO event has occured. Location 200AH is the interrupt vector for the Software Timer interrupt. Refer to Section 10 for more information on software timers and the HSO.

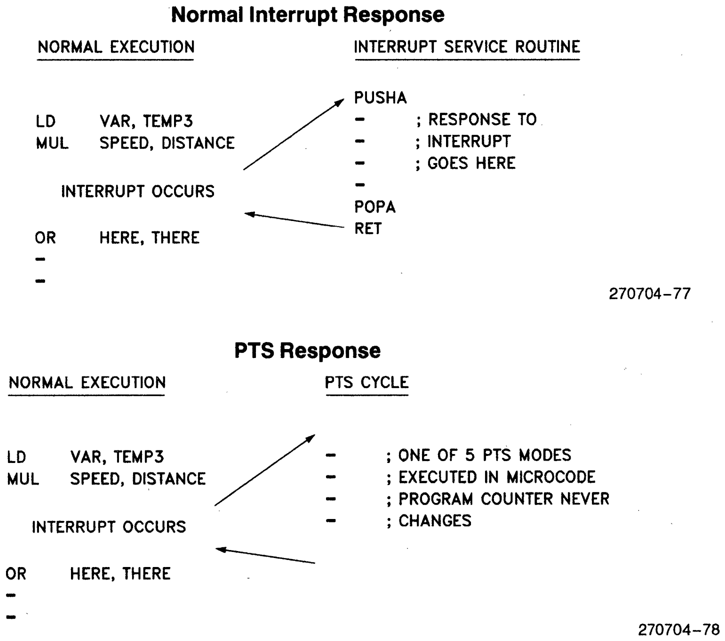
A/D Conversion Complete

The A/D Conversion Complete interrupt can generate an interrupt in response to a completed A/D conversion. The interrupt vectors indirectly through location 2002H. Refer to Section 12 for more information on the A/D Converter.

6.0 PERIPHERAL TRANSACTION SERVER

The Peripheral Transaction Server (PTS) is a new feature of the 80C196KC. The PTS provides DMA-like response to an interrupt with much less CPU overhead. Single and block transfer modes are supported, as well as special modes to service the A/D Converter and the HSI/O. Any of the 15 interrupt vectors can be alternatively mapped to its PTS channel.

Figure 6-1 shows the difference between a normal Interrupt Service Routine and the same interrupt mapped to its PTS channel. Instead of an Interrupt Service Routine, the PTS channel generates a PTS cycle. The software overhead of forcing the interrupt call, PUSHA, POPA, and executing the RET instruction is eliminated. Instead, the PTS cycle is interleaved with the normal instruction flow much like a DMA cycle.



Normal Interrupt Response

```
NORMAL EXECUTION                  INTERRUPT SERVICE ROUTINE

                                  PUSHA
LD    VAR, TEMP3                  -       ; RESPONSE TO
MUL   SPEED, DISTANCE             -       ; INTERRUPT
                                  -       ; GOES HERE
   INTERRUPT OCCURS               -
                                  POPA
OR    HERE, THERE                 RET
-
-
```

270704–77

PTS Response

```
NORMAL EXECUTION                  PTS CYCLE

LD    VAR, TEMP3                  -       ; ONE OF 5 PTS MODES
MUL   SPEED, DISTANCE             -       ; EXECUTED IN MICROCODE
                                  -       ; PROGRAM COUNTER NEVER
   INTERRUPT OCCURS               -       ; CHANGES

OR    HERE, THERE
-
-
```

270704–78

Figure 6-1. PTS vs Interrupt Response

6.1 PTS Control

The PTS vector table is composed of 15 words at locations 2040H–205CH as shown in Figure 6-2. The PTS vector table has the same format as the Interrupt vector table and the same priority scheme (14 highest, 0 lowest). All PTS channels have higher priority over any interrupts except NMI. Each PTS vector points to a PTS Control Block (PTSCB) which must reside in the internal RAM space (1AH–1FFH) at an address evenly divisible by 8. Figure 6-3 gives the format of the PTSCB for the 5 PTS modes. Unused bytes in the PTSCBs can be used as normal RAM locations. The PTSCB must be initialized by the user before the PTS channel is enabled. The function of each register is discussed in the next few sections.

The PTS is globally enabled by the PSE bit (Peripheral transaction Server Enable) in the PSW. PSE is set by the EPTS (Enable PTS) instruction and cleared by the DPTS (Disable PTS) instruction. When executing a PUSHA instruction the PSE bit is pushed onto the stack with the PSW and then cleared. The PTSSEL (PTS SELect) word register in HWindow 1 at 04H individually enables each PTS channel over the normal interrupt response. PTSSEL has the same format as the INT_PEND and INT_MASK registers and is shown in Figure 6-4. When a bit in PTSSEL is set, the associated interrupt request becomes a PTS request. Each PTS request will set the corresponding bit in the Interrupt Pending Register. If the corresponding bit in the Interrupt Mask Register is also set, and the PTS is globally enabled by the PSE bit, a PTS cycle will be initiated. See Figure 6-5.

PTS Vector	Location
HSI FIFO Full	205CH
EXTINT1	205AH
TIMER2 Overflow	2058H
TIMER2 Capture	2056H
4th HSI FIFO Entry	2054H
RI	2052H
TI	2050H
EXTINT	204EH
Serial Port	204CH
Software Timer	204AH
HSI.0 Pin	2048H
High Speed Outputs	2046H
HSI Data Available	2044H
A/D Conversion Complete	2042H
Timer Overflow	2040H

Figure 6-2. PTS Vector Table

As in the normal interrupt response, the current instruction is completed before the PTS cycle actually starts. The internal priority resolver handles the PTS requests based on their priority. Next, the PTS Vector is read from the PTS Vector table to get the address of

	Single Transfer	Block Transfer	A/D Mode	HSI & HSO Modes
	UNUSED	UNUSED	UNUSED	UNUSED
	UNUSED	PTSBLOCK	UNUSED	PTSBLOCK
	PTSDST (HI)	PTSDST (HI)	REG (HI)	UNUSED
	PTSDST (LO)	PTSDST (LO)	REG (LO)	UNUSED
	PTSSRC (HI)	PTSSRC (HI)	S/D (HI)	PTSSRC (HI)
	PTSSRC (LO)	PTSSRC (LO)	S/D (LO)	PTSSRC (LO)
	PTSCON	PTSCON	PTSCON	PTSCON
PTSVEC →	PTSCOUNT	PTSCOUNT	PTSCOUNT	PTSCOUNT

Figure 6-3. PTS Control Blocks

15															0
RSV	FIFO FULL	EXT INT1	T2 OVF	T2CAP	HSI4	RI	TI	EXT INT	SER PORT	SOFT TIMER	HSI.1 PIN	HSO PIN	HSI DATA	A/D DONE	TIMER OVF

PTSSEL @ 04H in HWINDOW 1

15															0
RSV	FIFO FULL	EXT INT1	T2 OVF	T2CAP	HSI4	RI	TI	EXT INT	SER PORT	SOFT TIMER	HSI.1 PIN	HSO PIN	HSI DATA	A/D DONE	TIMER OVF

PTSSRV @ 06H in HWINDOW 1

Figure 6-4. The PTSSRV and PTSSEL Registers

the PTS Control Block (PTSCB). The microcode then executes the proper PTS cycle, based on the contents of the PTSCB.

The PTSCOUNT in the PTSCB defines the number of PTS cycles to be run consecutively without software intervention. Since PTSCOUNT is an 8-bit value, the maximum number is 256. Loading PTSCOUNT with zero causes 256 transfers to occur. At the end of each PTS cycle, PTSCOUNT is decremented. When PTSCOUNT expires (i.e., equals 0), an actual interrupt called the end-of-PTS interrupt is requested which should invoke any processing needed and reinitialize the PTS channel.

When PTSCOUNT expires, a unique series of events happens. First, the associated bit in PTSSEL is cleared to inhibit any additional PTS cycles until after the end-of-PTS interrupt has executed. Secondly, the associated bit in PTSSRV (PTS Serve) is set to actually request the end-of-PTS interrupt. PTSSRV is located in HWindow 1 at 06H (see Figure 6-4). The PTSSRV register acts just like the Interrupt Pending registers in requesting interrupts. The PTSSRV register is used instead of the Pending Registers for the end-of-PTS interrupt so one actual PTS request from the interrupt source can be buffered in the Pending Register.

The end-of-PTS interrupt vectors through the associated location in the interrupt vector table. For example, if the TI interrupt is mapped to its PTS channel with its PTS vector at 2050H, its end-of-PTS interrupt is at 2030H. The end-of-PTS interrupt has the same priority as the normal interrupt vector. When the end-of-PTS interrupt is called, the bit in PTSSRV is automatically cleared, however the PTSSEL bit must be set manually to reenable the PTS channel. See Figure 6-5.

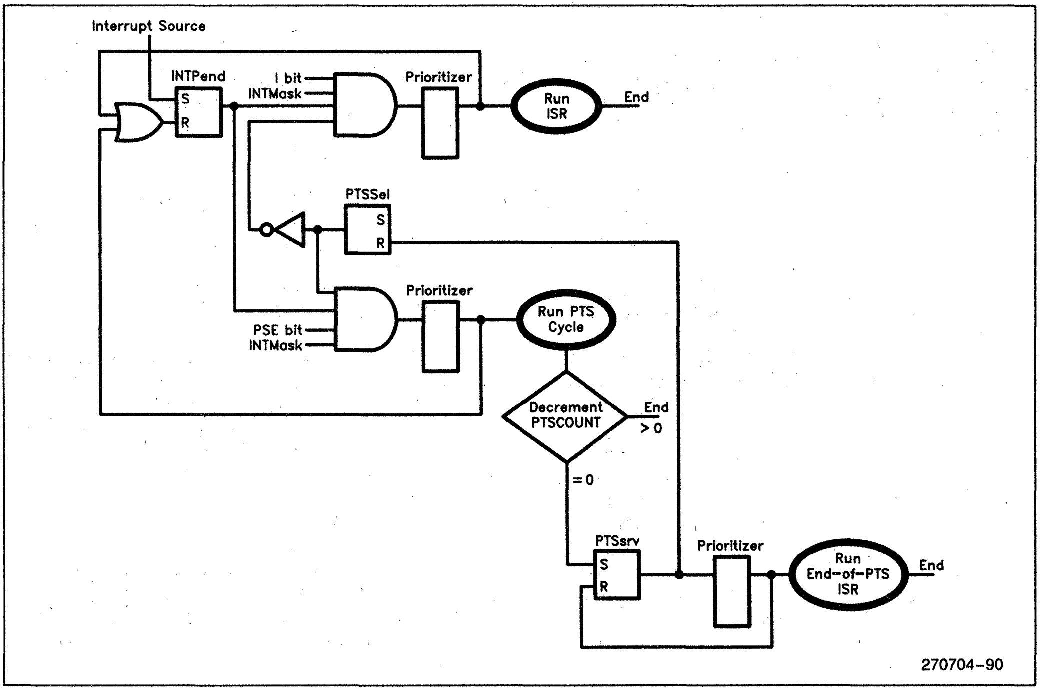


Figure 6-5

6.2 PTS Modes

PTS SINGLE TRANSFER MODE

In the Single Transfer Mode, the PTSCB (Figure 6-3) contains control (PTSCON), source (PTSSRC), destination (PTSDST), and count (PTSCOUNT) registers. A single transfer may be a byte or word and the PTSSRC and PTSDST may be optionally incremented at the end of the PTS cycle (see PTSCON in Figure 6-6). The registers increment by 1 for a byte transfer or by 2 for a word transfer. In the single transfer mode, both the auto-increment (SI, DI) and update bits (SU, DU) must be set if either PTSSRC or PTSDST are to be incremented.

7							0
M2	M1	M0	B/W	SU	DU	SI	DI

PTSCON

M2, M1, M0	(000) PTS Block Transfer Mode
	(100) PTS Single Transfer Mode
B/W	Byte (1)/Word (0) Transfer
SU	Update PTSSRC at End of PTS Cycle
DU	Update PTSDST at End of PTS Cycle
SI	PTSSRC Auto-Increment
DI	PTSDST Auto-Increment

7							0
M2	M1	M0	0	UPDT	0	1	0

PTSCON

M2, M1, M0	(110) PTS A/D Mode
	(011) PTS HSI Mode
	(001) PTS HSO Mode
UPDT	Optional Update of Incremented Value to: A/D— S/D REGISTER HSI— PTSDST HSO— PTSSRC

Figure 6-6. PTS Control Register Format

During a single transfer cycle, a byte or word is transferred from the memory location pointed to by the PTSSRC to the memory location pointed to by PTSDST. PTSSRC and PTSDST are optionally incremented with the SI, SU and DI, DU bits. PTSCOUNT is then decremented. If PTSCOUNT equals 0, the appropriate PTSSRV bit is set and the PTSSEL bit is cleared to disable any further PTS cycles until the End-of-PTS ISR is executed.

A single transfer takes 18 states + 3 for each memory controller reference.

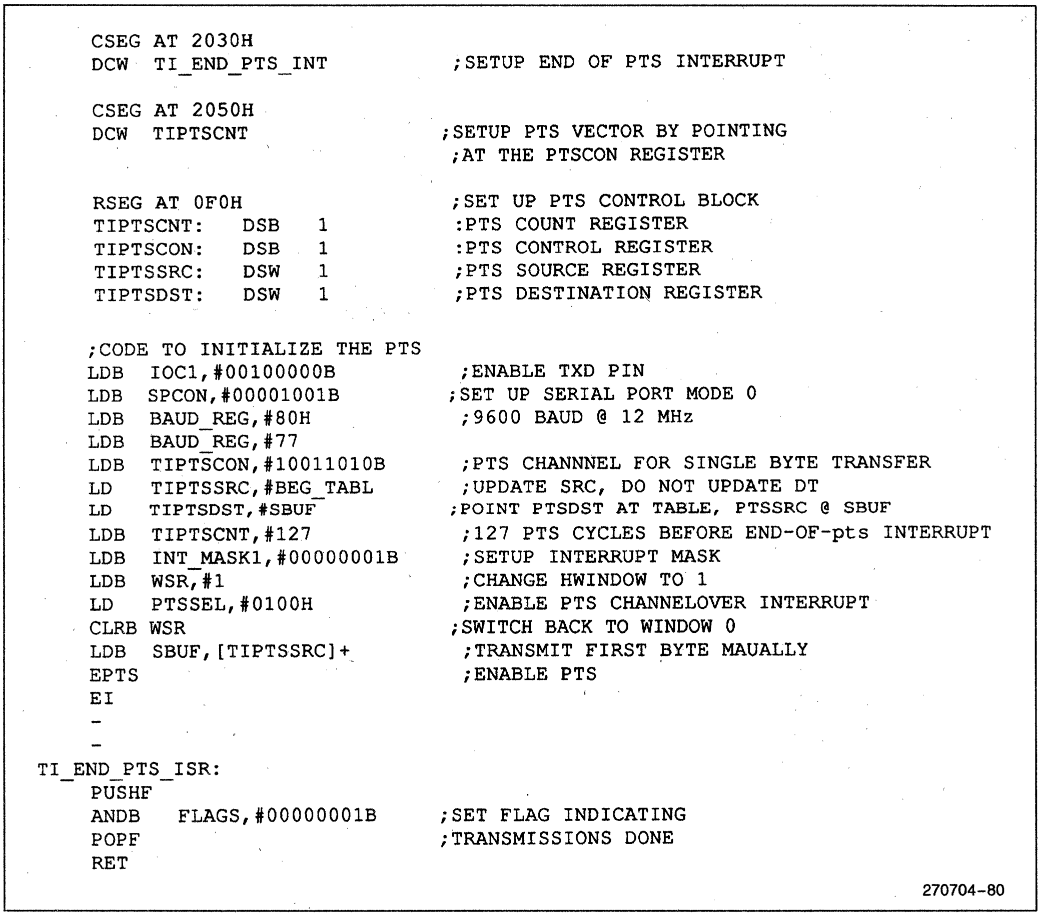
At this time, an example would probably be of great help. Let's say a 128-byte block of contiguous memory needs to be transmitted over the serial port to another host processor. Figure 6-7 shows the code necessary to accomplish this task using the TI interrupt and an Interrupt Service Routine. This code takes 35 state times to set up the ISR, 84 states to transfer each byte, and 59 states for the final ISR for a total of 10727 state times (35 + 127 (84) + 59).

This same task can easily be accomplished using the PTS channel associated with the TI interrupt with much less CPU overhead. The PTS channel is set up to do a single byte transfer on every TI PTS request. The PTSDST register always points to SBUF and does not increment. The PTSSRC register points to the beginning of the block of memory and is set up to increment every PTS cycle. The code to set up the PTS channel and the End-of-PTS interrupt is shown in Figure 6-6. It takes 64 state times to set up the PTS channel, 21 state times to transfer each byte, and 54 state times for the End-of-PTS interrupt for a grand total of 2721 state times (64 + 127 (21) + 54). This reduces the software overhead of this task by 75% compared to an Interrupt Service Routine.

```
        CSEG AT 2030H
        DCW   TI_ISR                  ;SET UP TI INTERRUPT VECTOR

        ;code to initialize TI interrupt

        LDB  IOC1,#00100000B          ;ENABLE TXD PIN
        LDB  SPCON,#00001001B         ;SET UP SERIAL PORT FOR MODE 0
        LDB  BAUD_REG,#77
        LDB  BAUD_REG,#80H            ;SET UP FOR 9600 BAUD @ 12 MHz
        LD   POINTER,#BEG_TABL       ;POINT AT BEGINNING OF TABLE
                                      ;TO BE TRANSMITTED
        LDB  INT_MASK1,#00000001B     ;INTIALIZE INTERRUPT MASK REG
        EI                            ;ENABLE INTERRUPTS
        LDB  SBUF,[POINTER]+         ;TRANSMIT FIRST BYTE IN TABLE

        -
        -

 ; TRANSMIT INTERRUPT SERVICE ROUTINE
 ; ROUTINE TAKES 84 STATE TIMES TO TRANSMIT BYTE, 59 IF IT SETS
 ; THE TRANSMISSION DONE FLAG
  TI_ISR:
        PUSHF
        LDB  SP_TMP,SP_STAT       ;MAKE BACKUP COPY OF SP_STAT
        JBC  SP_TMP,5,OUT         ;CHECK FOR BOGUS INTERRUPTS
        ANDB SP_TMP,#00100000B    ;CLEAR TI BIT IN SP_TEMP
        CMP  POINTER,#END_TABL    ;CHECK TO SEE IF AT END OF TABLE
        JNE  TRANS_AGAIN
        ORB  FLAGS,#000000001B    ;IF AT END OF TABLE, SET A FLAG
        SJMP OUT                  ;TO INDICATE DONE TRANSMITTING
TRANS_AGAIN:
         LDB  SBUF,[POINTER]+     ;SEND ANOTHER BYTE
        OUT:
         POPF
         RET                      ;RETURN TO MAIN PROGRAM
```

270704-79

Figure 6-7. Transmit Interrupt Service Routine



```
    CSEG AT 2030H
    DCW  TI_END_PTS_INT              ;SETUP END OF PTS INTERRUPT

    CSEG AT 2050H
    DCW  TIPTSCNT                    ;SETUP PTS VECTOR BY POINTING
                                     ;AT THE PTSCON REGISTER

    RSEG AT 0F0H                     ;SET UP PTS CONTROL BLOCK
    TIPTSCNT:   DSB   1              :PTS COUNT REGISTER
    TIPTSCON:   DSB   1              :PTS CONTROL REGISTER
    TIPTSSRC:   DSW   1              ;PTS SOURCE REGISTER
    TIPTSDST:   DSW   1              ;PTS DESTINATION REGISTER

    ;CODE TO INITIALIZE THE PTS
    LDB  IOC1,#00100000B              ;ENABLE TXD PIN
    LDB  SPCON,#00001001B            ;SET UP SERIAL PORT MODE 0
    LDB  BAUD_REG,#80H                ;9600 BAUD @ 12 MHz
    LDB  BAUD_REG,#77
    LDB  TIPTSCON,#10011010B          ;PTS CHANNNEL FOR SINGLE BYTE TRANSFER
    LD   TIPTSSRC,#BEG_TABL           ;UPDATE SRC, DO NOT UPDATE DT
    LD   TIPTSDST,#SBUF              ;POINT PTSDST AT TABLE, PTSSRC @ SBUF
    LDB  TIPTSCNT,#127                ;127 PTS CYCLES BEFORE END-OF-pts INTERRUPT
    LDB  INT_MASK1,#00000001B         ;SETUP INTERRUPT MASK
    LDB  WSR,#1                       ;CHANGE HWINDOW TO 1
    LD   PTSSEL,#0100H                ;ENABLE PTS CHANNELOVER INTERRUPT
    CLRB WSR                         ;SWITCH BACK TO WINDOW 0
    LDB  SBUF,[TIPTSSRC]+             ;TRANSMIT FIRST BYTE MAUALLY
    EPTS                              ;ENABLE PTS
    EI
    -
    -
TI_END_PTS_ISR:
    PUSHF
    ANDB    FLAGS,#00000001B         ;SET FLAG INDICATING
    POPF                             ;TRANSMISSIONS DONE
    RET
```

270704-80

Figure 6-8. Transmit Interrupt PTS Example

PTS BLOCK TRANSFER MODE

For a PTS Block Transfer, the PTSCB (Figure 6-3) contains all the same registers as the Single Transfer Mode with the addition of a block count (PTSBLOCK) register. The PTSCON also retains the same format as the Single Transfer Mode (see Figure 6-6).

When a Block Transfer is selected, PTSBLOCK determines how many byte or word transfers will take place (N = 1 to 32 transfers). Loading a zero in PTSBLOCK causes 32 transfers to take place. N transfers take place from the memory pointed to by PTSSRC to the memory pointed to by PTSDST. PTSSRC and PTSDST are optionally incremented after each transfer by the SI and DI bits. When N transfers have expired, the PTSSRC and PTSDST registers are optionally updated with the SU and DU bits. This allows the PTSSRC and PTSDST registers to be incremented during a Block Transfer with the SI and DI bits, and using the SU and DU bits, keep their final value or revert to their value at the beginning of the PTS cycle. Finally, the PTSCOUNT register is decremented and if it equals 0, the PTSSRV bit is set and the PTSSEL bit is cleared to request the end-of-PTS interrupt.

A Block Transfer PTS cycle takes 13 states + 7 for each transfer (1 minimum) + 3 for each memory controller reference.

Care must be taken when using the Block Transfer. A Block Transfer cannot be interrupted. It would be very easy to make a long uninterruptable instruction with a Block Transfer. Taking the worst case, a Block Transfer of 32 words from external source to external destination over an 8-bit bus would take about 500 states (assuming no wait states).

PTS A/D MODE

The A/D mode allows automatic restart of the A/D converter while storing the previous result in a table located in memory by mapping the A/D__DONE interrupt to its PTS channel. Figure 6-9 shows the A/D table format. The User sets up the A/D commands in the table and the PTS loads the A/D with the command and stores the result in the appropriate table location. For more information on the A/D Converter, see Section 12.

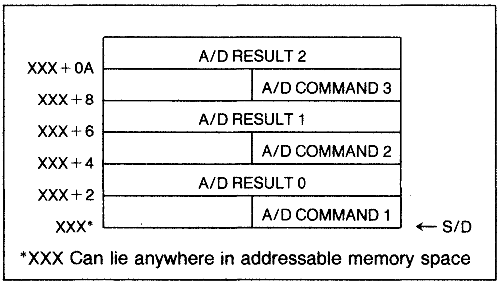


Figure 6-9. PTS A/D Table Format

The PTSCB contains a source/destination register (S/D), a register address (REG), and the PTS control register (PTSCON). The PTSCON format for the A/D Mode is shown in Figure 6-5. The S/D register points to the table in memory, and REG will point to the A/D__COMMAND register at location 02H in HWindow 0.

In a PTS A/D cycle, the word pointed to by S/D is loaded into a temporary internal register. S/D is then incremented by 2. Next, the A/D__RESULT register is stored at the location pointed to by S/D. The A/D command stored in the temporary register is now loaded into the A/D__COMMAND register to initiate another A/D conversion. Now the S/D is optionally updated to point to the next word in the table with the UPDT bit in PTSCON. If the S/D is not updated, the same A/D command is read and the result stored in the same location for every PTS cycle. Finally, PTSCOUNT is decremented and if it is zero, the PTSSEL bit is cleared and the PTSSRV bit is set to request the end-of-PTS interrupt.

The A/D mode takes 21 states when the table is in internal RAM/SFR space (0–1FFH) and 25 states with memory controller references (200H–0FFFFH).

PTS HSI MODE

The PTS HSI mode allows the FIFO to be dumped out to a table in either internal or external memory by mapping one of the 3 HSI interrupts to its PTS channel. This makes it easier to save several HSI events in memory for later processing. The format of the HSI table is shown in Figure 6-10. For more information on the HSI, see Section 9.

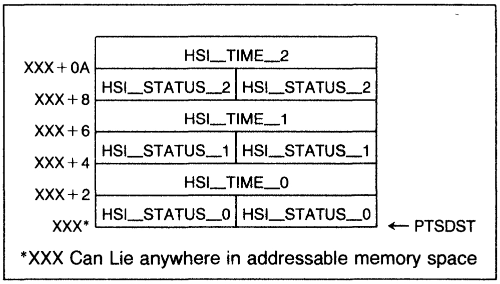


Figure 6-10. HSI PTS Table

The PTSCB is made up of a destination register (PTSDST), a block count register (PTSBLOCK), and the control register (PTSCON) as shown in Figures 6-3 and 6-5. PTSDST can be optionally updated at the end of the PTS cycle by setting the UPDT bit in PTSCON.

When this PTS cycle is initiated, PTSBLOCK is read to determine how many HSI transfers will take place (N = 1 to 7). For each transfer, the HSI__STATUS and HSI__TIME registers are written out to consecutive words in memory pointed to by the PTSDST. When N transfers have finished, PTSDST is optionally updated. Finally the value in PTSCOUNT is decremented and if it is 0, the PTSSRV bit is set and the PTSSEL bit is cleared to request the End-of-PTS interrupt.

The HSI can generate an interrupt or PTS request when the FIFO contains 1, 5 or 7 entries, so the PTSBLOCK register should contain one of these values.

The HSI mode takes 12 state times + 10 for each block transfer (1 minimum) for Internal RAM/SFR space and 16 states + 10 for each block transfer (1 minimum) with memory controller references.

PTS HSO (HIGH SPEED OUTPUT) Mode

The PTS HSO mode allows the HSO CAM to be loaded from a table located in internal or external memory as shown in Figure 6-11. For further information on the HSO, see Section 10. The HSO mode operates the same way as the HSI mode except the table is read rather than written.

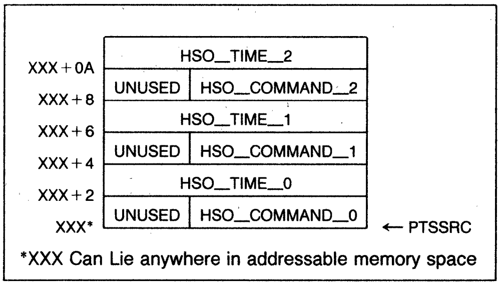


Figure 6-11. PTS HSO Table

The PTSCB contains a PTSSRC, PTSDST, PTSBLOCK, and PTSCON register. The PTSCON register for the HSO mode is shown in Figure 6-6. The source must always be incremented and the source can be optionally updated at the end of the PTS cycle. When this PTS cycle is initiated, PTSLOCK is read to determine how many HSO transfers will take place (N = 1 to 8). For each transfer, the two consecutive words pointed to by PTSSRC are read and loaded into the HSO_COMMAND and HSO_TIME registers, respectively. When the N transfers are done, PTSSRC is optionally incremented and PTSCOUNT is decremented. If PTSCOUNT equals 0, the PTSSEL bit is cleared and the PTSSRV is set to request an End-of-PTS interrupt.

The PTS HSO mode takes 11 state times + 10 for each block transfer (1 minimum) with internal RAM/SFR space or 15 states + 11 for each block transfer (1 minimum) with memory controller references.

PTS Latency

Because the prelude to a PTS request is so much like that of an Interrupt, the latency is calculated in the same manner. PTS latency is therefore defined to be the longest instruction (NORML − 39 states) + 4 states = 43 states. See Figure 5-5. This does not include any higher priority PTS requests that may be executing or any time that the PTS is disabled via the PSE bit in the PSW.

7.0 PULSE WIDTH MODULATION OUTPUT (D/A)

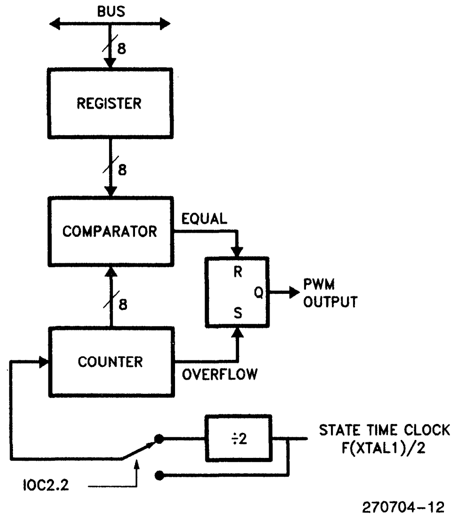
Digital to analog conversion can be done with any of three Pulse Width Modulation outputs; a block diagram of the circuit is shown in Figure 7-1. The 8-bit counter is incremented every state time. When it equals 0, the PWM output is set to a one. When the counter matches the value in the corresponding PWM register, the output is switched low. When the counter overflows, the outputs are once again switched high. A typical output waveform is shown in Figure 7-2. When the PWM register equals 00, the output is always low. Additionally, the PWM register will only be reloaded from the temporary latch when the counter overflows. This means the compare circuit will not recognize a new value until the counter has expired preventing missed edges.

The 80C196KC PWM unit has a prescaler bit (divide by 2) which is enabled by setting IOC2.2 = 1. The output waveform is a variable duty cycle pulse which repeats every 256 or 512 state times (32 μs or 64 μs at 16 MHz). Changes in the duty cycle are made by writing to the PWM register. PWM0 register is at location 17H in HWindow 0 and the value programmed into the PWM0 register can be read in HWindow 15 (WSR = 15). PWM0 is compatible with the PWM output on the 80C196KB. PWM1 and PWM2 registers are located at location 16H and 17H in HWindow 1 and are read/writable in HWindow 1. There are several types of motors which require a PWM waveform for more efficient operation. Additionally, if this waveform is integrated it will produce a DC level which can be changed in 256 steps by varying the duty cycle, as described in the next section.

XTAL1 =	8 MHz	10 MHz	16 MHz
IOC2.2 = 0	15.6 KHz	19.6 KHz	31.25 KHz
IOC2.2 = 1	7.8 KHz	9.8 KHz	15.63 KHz

Figure 7-3. PWM Frequencies

The PWM0 output shares a pin with Port 2, pin 5 so that these two features cannot be used at the same time. IOC1.0 equal to 1 selects the PWM function. PWM1 and PWM2 are multiplexed on Port 1, pins 3 and 4, respectively. T2CNTC register bit 2 and 3 in HWindow 1 enables the PWM1 and PWM2 outputs over the port function. All three PWM outputs use the same timer. Therefore, the outputs go high at the same time. When the pins are enabled as PWMs, the pin is no longer a quasi bidirectional port but has strong pullups and pulldowns. The ports cannot be returned to quasi bidirectionals unless the device is reset.



- Duty Cycle Programmable in 256 Steps

Figure 7-1. PWM Block Diagram

7.1 Analog Outputs

Analog outputs can be generated by two methods, either by using the PWM output or the HSO. Either device will generate a rectangular pulse train that varies in duty cycle and period. If a smooth analog signal is desired as an output, the rectangular waveform must be filtered.

In most cases this filtering is best done after the signal is buffered to make it swing from 0 to 5 volts since both of the outputs are guaranteed only to low current levels. A block diagram of the type of circuit needed is shown in Figure 7-4. By proper selection of components, accounting for temperature and power supply drift, a highly accurate 8-bit D to A converter can be made using either the HSO or the PWM output. Figure 7-5 shows two typical circuits. If the HSO is used the accuracy could be theoretically extended to 16-bits, however the temperature and noise related problems would be extremely hard to handle.

When driving some circuits it may be desirable to use unfiltered Pulse Width Modulation. This is particularly true for motor drive circuits. The PWM output can generate these waveforms if a fixed period on the order of 32 μs is acceptable. If this is not the case then the HSO unit can be used. The HSO can generate a variable waveform with a duty cycle variable in up to 65536 steps and a period of up to 66 milliseconds with Timer1.

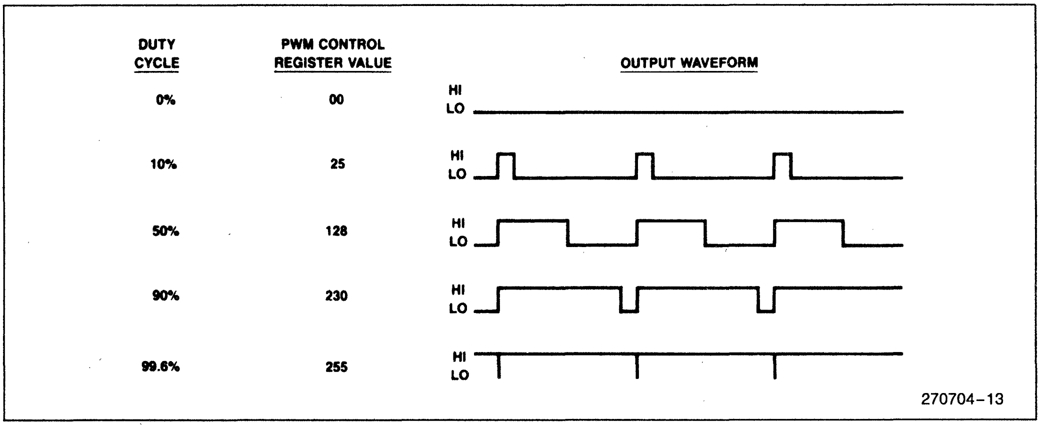


Figure 7-2. Typical PWM Outputs

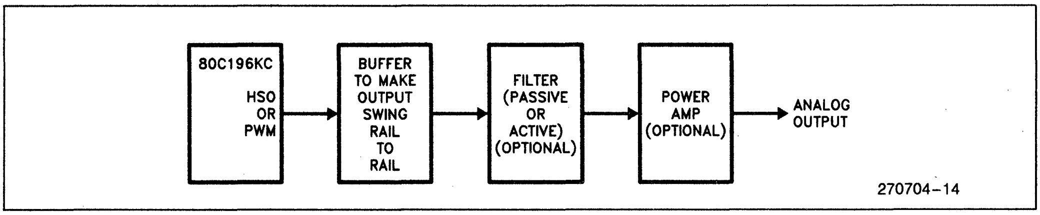

80C196KC
HSO OR PWM
BUFFER TO MAKE OUTPUT SWING RAIL TO RAIL
FILTER (PASSIVE OR ACTIVE) (OPTIONAL)
POWER AMP (OPTIONAL)
ANALOG OUTPUT
270704-14

Figure 7-4. D/A Buffer Block Diagram

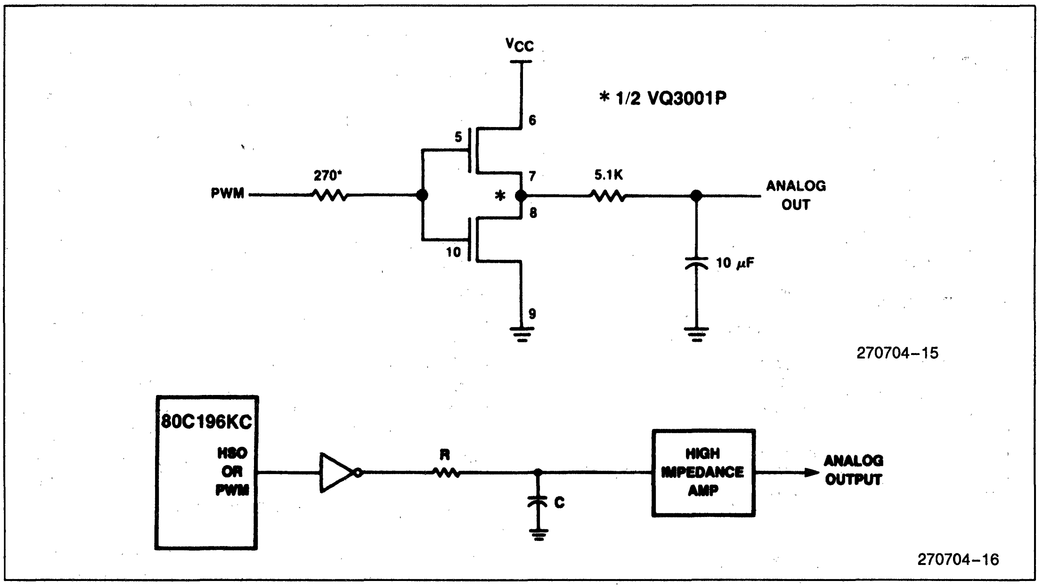


Figure 7-5. Buffer Circuits for D/A

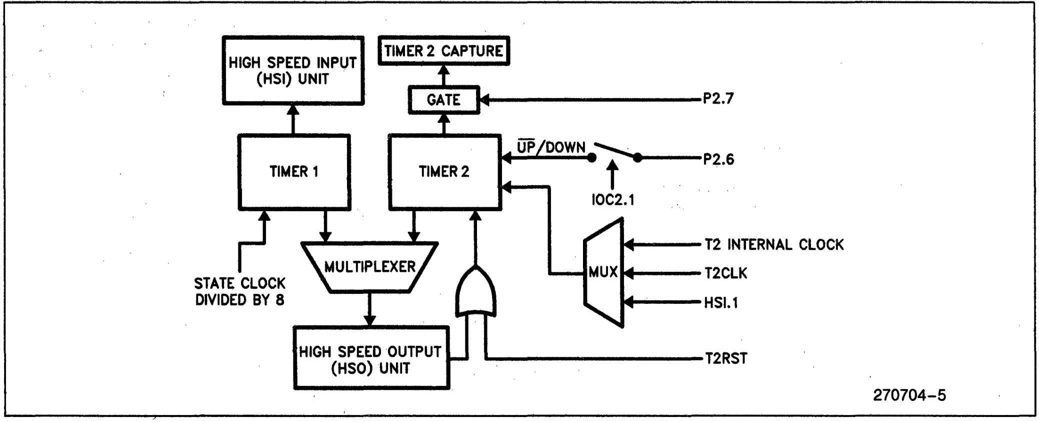


Figure 8-1. Timer Block Diagram

8.0 TIMERS

Figure 8-1 shows Timer1 and Timer2

8.1 Timer1

Timer1 is a 16-bit free-running timer which is incremented every eight state times. An interrupt can be generated in response to an overflow. It is read through location 0AH in HWindow 0 and written in HWindow 15. Care must be taken when writing to it if the High Speed I/O (HSIO) Subsystem is being used. HSO time entries in the CAM depend on exact matches with Timer1. Writes to Timer1 should be taken into account in software to ensure events in the HSO CAM are not missed or occur in an order which may be unexpected. Changing Timer1 with incoming events on the High Speed Input lines may corrupt relative references between captured inputs. Further information on the High Speed Outputs and High Speed Inputs can be found in Sections 9 and 10 respectively.

8.2 Timer2

Timer2 on the 80C196KC can be used as a reference for the HSO unit, an up/down counter, an external event capture or as an extra counter. Timer2 is clocked externally using either the T2CLK pin or the HSI.1 pin Timer2 counts on both positive and negative transitions. The maximum transition speed is once per state time in the Fast Increment mode, and once every 8 states otherwise. New on the 80C196KC, Timer2 can be clocked internally every 1 or 8 state times. Timer2 can be read and written through location 0CH in HWindow 0. Timer2 can be reset by hardware, software or the HSO unit. Either T2RST or HSI.0 can reset Timer2 externally depending on the setting of IOC0.5. Figure 8-2 shows the configuration and input pins of Timer2. Figure 8-3 shows the reset and clocking options for Timer2. The appropriate control registers can be read in HWindow 15 to determine the programmed modes. However, IOC0.1 (T2RST) is not latched and will read a 1.

Caution should be used when writing to the timers if they are used as a reference to the High Speed Output Unit. Programmed HSO commands could be missed if the timers do not count continuously in one direction. High Speed Output events based on Timer2 must be carefully programmed when using Timer2 as an up/down counter that can be reset externally. Programmed events could be missed or occur in the wrong order. Refer to Section 9 for more information on using the timers with the High Speed Output Unit.

Capture Register

The value in Timer2 can be captured into the T2CAPture register by a rising edge on Port 2 pin 7. The logic level must be held for at least one state time as discussed in the next section. T2CAP is located at 0CH in HWindow 15. The interrupt generated by a T2CAPture vectors through location 2036H.

Fast Increment Mode

Timer2 can be programmed to run in fast increment mode to count transitions every state time. Setting IOC2.0 programs Timer2 in the Fast Increment mode. In this mode, the events programmed on the HSO unit with Timer2 as a reference will not execute properly since the HSO requires eight state times to compare every location in the HSO CAM. With Timer2 as a reference for the HSO unit, Timer2 transitioning every state time may cause programmed HSO events to be missed. For this reason, Timer2 should not be used as a reference for the HSO if transitions occur faster than once every eight state times.

Timer2 should not be RESET in the fast increment mode. All Timer2 resets are synchronized to an eight state time clock. If Timer2 is reset when clocking faster than once every 8 states, it may reset on a different count.

Internal Clock Mode

A new feature on the 80C196KC is the ability for Timer2 to be clocked internally. Timer2 can be clocked every 1 or 8 states. In the 8 state mode, Timer2 increments at the same time as Timer1. Internal clocking is enabled by setting T2CNTC.0. Clocking Timer2 every state time is controlled by setting IOC2.0 while clearing IOC2.0 causes Timer2 to count every 8 states.

Up/Down Counter Mode

Timer2 can be made to count up or down based on the Port 2.6 pin if IOC2.1 = 1. However, caution must be

	Bit = 1	Bit = 0
IOC0.1	Reset Timer2 each write	No action
IOC0.3	Enable external reset	Disable
IOC0.5	HSI.0 is ext. reset source	T2RST is reset source
IOC0.7	HSI.1 is T2 clock source	T2CLK is clock source
IOC1.3	Enable Timer2 overflow int.	Disable overflow interrupt
IOC2.0	Enable fast increment	Disable fast increment
IOC2.1	Enable downcount feature	Disable downcount
P2.6	Count down if IOC2.1 = 1	Count up
IOC2.5	Interrupt on 7FFFH/8000H	Interrupt on 0FFFFH/0000H
P2.7	Capture Timer2 into T2CAPture on rising edge	
T2CNTC.0	Selects Timer2 internal clock source	Selects Timer2 internal clock source

Figure 8-2. Timer2 Configuration and Control Pins

used when this feature is working in conjunction with the HSO. If Timer2 does not complete a full cycle it is possible to have events in the CAM which never match the timer. These events would stay in the CAM until the CAM is cleared or the chip is reset.

8.3 Sampling on External Timer Pins

The T2UP/DN, T2CLK, T2RST, and T2CAP pins are sampled during PH1. PH1 roughly corresponds to CLKOUT low externally. For valid sampling, the inputs should be present 45 ns prior to the rising edge of CLKOUT or it may not be sampled until the next CLKOUT. To synchronize the inputs, the rising edge of CLKOUT should latch the inputs and hold them until the next rising edge of CLKOUT. T2UP/DN and T2CLK need to be synchronized unless they never transition within one state time of each other. Otherwise, Timer2 may count in the wrong direction.

8.4 Timer Interrupts

Both Timer1 and Timer2 can trigger a timer overflow interrupt and set a flag in the I/O Status Register 1 (IOS1). Timer1 overflow is controlled by setting IOC1.2 and the interrupt status is indicated in IOS1.5. The TIMER OVERFLOW interrupt is enabled by setting INT__MASK.0.

A Timer2 overflow condition interrupts through location 2000H by setting IOC1.3 and setting INT__MASK.0. Alternatively, Timer2 overflow can interrupt through location 2038H by setting INT__MASK1.3. The status of the Timer2 overflow interrupt is indicated in IOS1.4.

Interrupts can be generated if Timer2 crosses the 0FFFFH/0000H boundary or the 7FFFH/8000H boundary in either direction. By having two interrupt points it is possible to have interrupts enabled even if Timer2 is counting up and down centered around one of the interrupt points. The boundaries used to control the Timer2 interrupt is determined by the setting of IOC2.5. When set, Timer2 will interrupt on the 7FFFH/8000H boundary, otherwise, the 0FFFFH/0000H boundary interrupts.

A T2CAPTURE interrupt is enabled by setting INT__MASK1.3. The interrupt will vector through location 2036H.

Caution must be used when examining the flags, as any access (including Compare and Jump on Bit) of IOS1 clears bits 0 through 5 including the software timer flags. It is, therefore, recommended to copy the byte to a temporary register before testing bits. Writing to IOS1 in HWindow 15 will set the status bits but not cause interrupts. The general enabling and disabling of the timer interrupts are controlled by the Interrupt Mask Register bit 0. In all cases, setting a bit enables a function, while clearing a bit disables it.

9.0 HIGH SPEED INPUTS

The High Speed Input Unit (HSI) can record the time an event occurs with respect to Timer1. There are 4 lines (HSI.0 through HSI.3) and up to a total of 8 events can be recorded. HSI.2 and HSI.3 are bidirectional pins which can also be used as HSO.4 and HSO.5. The I/O Control Registers (IOC0 and IOC1) determine the functions of these pins. The values programmed into IOC0 and IOC1 can be read in HWindow 15. A block diagram of the HSI unit is shown in Figure 9-1.

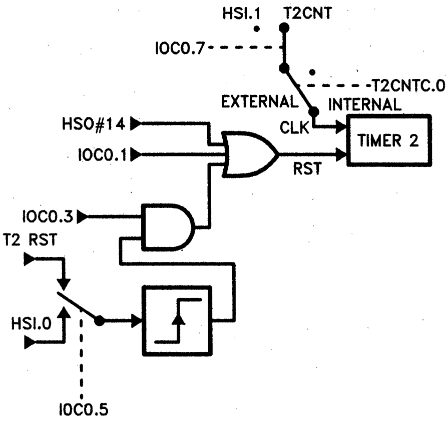


Figure 8-3. Timer2 Clock and Reset Options

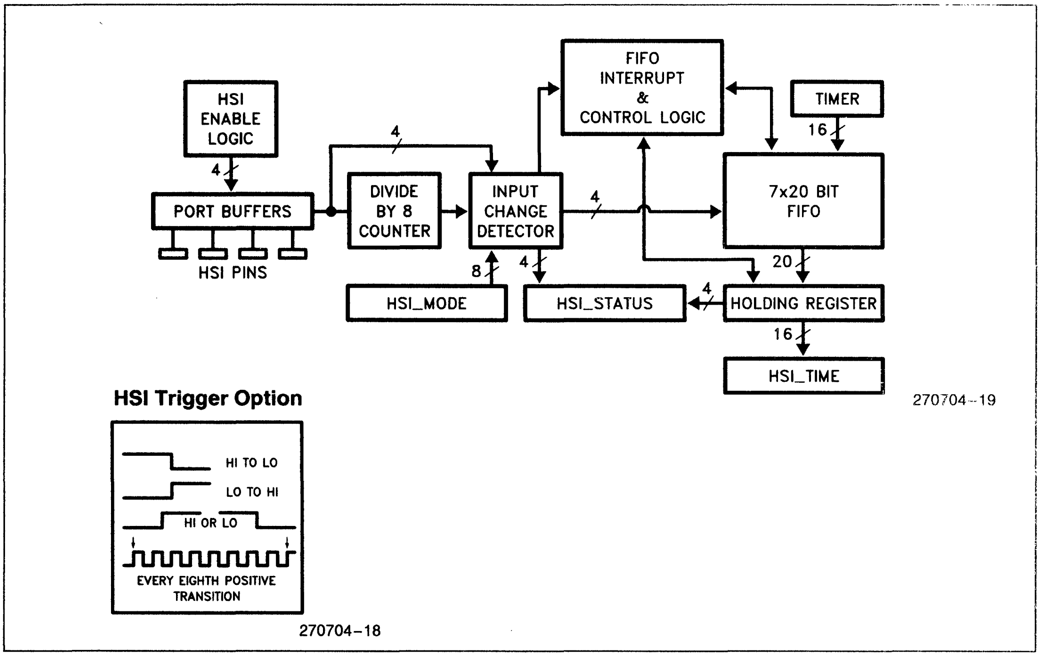
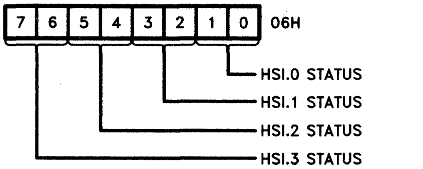


Figure 9-1. High Speed Input Unit

HSI Status Register (HSI_Status)



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT (STATUS BIT) INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT (INPUT BIT) INDICATES THE CURRENT INPUT LEVEL OF THE PIN.

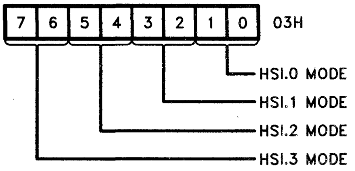
270704-20

Figure 9-2. HSI Status Register Diagram

When an HSI event occurs, a 7×20 FIFO stores the 16 bits of Timer1, and the 4 bits indicating which pins recorded events associated with that time tag. Multiple pins can recognize events with the same time tag. Therefore, if multiple pins are being used as HSI inputs, software must check each status bit when processing an HSI event. It can take up to 8 state times for this information to reach the holding register. For this reason, 8 state times must elapse between consecutive reads of HSI_TIME. When the FIFO is full, one additional event, for a total of 8 events, can be stored by considering the holding register part of the FIFO. If the FIFO and holding register are full, any additional events will not be recorded.

9.1 HSI Modes

There are 4 possible modes of operation for each of the HSI pins. The HSI_MODE register at location 03H controls which pins will look for what type of events. In HWindow 15, reading the register will read back the programmed HSI mode. The 8-bit register is set up as shown in Figure 9-3.



WHERE EACH 2-BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

270704-21

Figure 9-3. HSI Mode Register 1

The maximum input speed is 1 event every 8 state times except when the 8 transition mode is used, in which case it is 1 transition per state time.

The HSI pins can be individually enabled and disabled using bits in IOC0, as shown in Figure 9-4. If the pin is disabled, events are not entered in the FIFO. However, the input bits of the HSI_STATUS (Figure 9-2) are always valid regardless of whether the pin is enabled to the FIFO. This allows the HSI pins to be used as general purpose input pins.

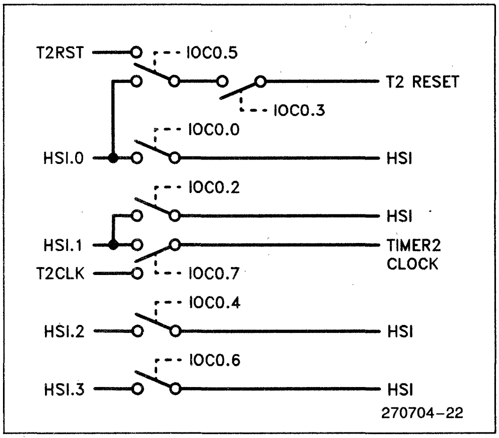


Figure 9-4. IOC0 Control of HSI Pin Functions

9.2 HSI Status

Bits 6 and 7 of I/O Status Register 1 (IOC1—see Figure 9-5) indicate the status of the HSI FIFO. If bit 7 is set, the HSI holding register is loaded. The FIFO may or may not contain 1–5 events. If bit 6 of IOC1 is set, the FIFO contains 6 entries. If the FIFO fills, future events will not be recorded. Reading IOC1 clears bits 0–5, so keep an image of the register and test the image to retain all 6 bits.

The HSI holding register must be read in a certain order. The HSI_STATUS Register (Figure 9-2) is read first to obtain the status and input bits. Second, the HSI_TIME Register (04H) is read to obtain the time tag. Reading HSI_TIME unloads one level of the FIFO. If the HSI_TIME is read before HSI_STATUS, the contents of HSI_STATUS associated with that time HSI_TIME tag are lost.

If the HSI_TIME register is read without the holding register being loaded, the returned value will be indeterminate. Under the same conditions, the four bits in HSI_STATUS indicating which events have occurred will also be indeterminate. The four HSI_STATUS bits which indicate the current state of the pins will always return the correct value.

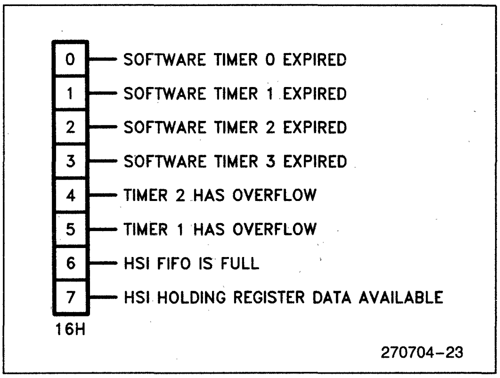

0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE
16H
270704-23

Figure 9-5. I/O Status Register 1

It should be noted that many of the status register conditions are changed by a reset, see Section 13. Writing to HSI_TIME in HWindow 15 will write to the HSI FIFO holding register. Writing to HSI_STATUS in HWindow 15 will set the status bits but will not affect the input bits.

9.3 HSI Interrupts

Interrupts can be generated by the HSI unit in three ways: each time a value moves from the FIFO into the holding register, when the FIFO has 4 or more events stored; when the FIFO has 6 or more events.

The HSI_DATA_AVAILABLE and HSI_FIFO_FULL interrupts are shared on the 8096BH. The source for the HSI_DATA_AVAILABLE interrupt is controlled by IOC1.7. When IOC1.7 is cleared, the HSI will generate an interrupt when the holding register is loaded. The interrupt indicates at least one HSI event has occurred and is ready to be processed. The interrupt vectors through location 2004H. The interrupt is enabled by setting INT_MASK.2. The generation of a HSI_DATA_AVAILABLE interrupt will set IOS1.7. The HSI_FIFO_ FULL interrupt will vector through HSI_DATA_AVAILABLE if IOC1.7 is set. On the 80C196KC, the HSI_FIFO_FULL has a separate interrupt vector at location 203CH.

A HSI_FIFO_FULL interrupt occurs when the HSI_FIFO has six or more entries loaded independent of the holding register. Since all interrupts are rising edge triggered, the processor will not be reinterrupted until the FIFO first contains 5 or less records, then contains six or more. The HSI FIFO FULL interrupt mask bit is INT_MASK1.6. The occurrence of a HSI FIFO FULL interrupt is indicated by IOS1.6. Earlier warning of a impending FIFO full condition can be achieved by the HSI FIFO 4th Entry interrupt.

The HSI_FIFO_4 interrupt generates an interrupt when four or more events are stored in the HSI FIFO independent of the holding register. The interrupt is enabled by setting INT_MASK1.2. The HSI_FIFO_4 vectors indirectly through location 2034H. There is no status flag associated with the HSI_FIFO_4 interrupt since it has its own independent interrupt vector.

The HSI.0 pin can generate an interrupt on the rising edge even if its not enabled to the HSI FIFO. An interrupt generated by this pin vectors through location 2008H.

9.4 HSI Input Sampling

The HSI pins are sampled internally once each state time. Any value on these pins must remain stable for at least 1 full state time to guarantee that it is recognized. The actual sampling occurs during PH1 or during CLKOUT low. The HSI inputs should be valid at least 45 ns before the rising of CLKOUT. Otherwise, the HSI input may be sampled in the next CLKOUT. Therefore, if information is to be synchronized to the HSI it should be latched on the rising edge of CLKOUT.

9.5 Initializing the HSI

When starting the HSI, two things need to be done. The FIFO should first be flushed and the HSI initialized. The FIFO should be flushed to clear out any pending events. The following section of code can be used to flush the FIFO:

```
reflush: ld 0, HSI_TIME ;clear an event
         skip           ;wait 8 state times
         skip
         jbs IOS1, 7, reflush
```

When initializing the HSI, interrupt(s) need to be enabled and the HSI pins need to be individually enabled to the FIFO through IOC0. It is very important to initialize the interrupts before the HSI pins or a FIFO lockout condition could occur. For example, if the HSI pins were enabled first, an event could get loaded into the holding register before the HSI_DATA_AVAILABLE interrupt is enabled. If this happens, no HSI_DATA_AVAILABLE interrupts will ever occur.

10.0 HIGH SPEED OUTPUTS

The High Speed Output unit (HSO) trigger events at specific times with minimal CPU overhead. Events are generated by writing commands to the HSO_COMMAND register and the relative time at which the events are to occur into the HSO_TIME register. In HWindow 15, these registers will read the last value programmed in the holding register. The programmable events include: starting an A/D conversion, resetting Timer2, setting 4 software flags, and switching 6 output lines (HSO.0 through HSO.5). The format of

	7	6	5	4	3	2	1	0	
HSO_ COMMAND	CAM LOCK	TMR2/ $\overline{\text{TMR1}}$	SET/ CLEAR	INT/ $\overline{\text{INT}}$	CHANNEL				06H

CAM Lock — Locks event in CAM if this is enabled by IOC2.6 (ENA_LOCK)
TMR2/$\overline{\text{TMR1}}$ — Events Based on Timer2/Based on Timer1 if 0
SET/$\overline{\text{CLEAR}}$ — Set HSO pin/Clear HSO pin if 0
INT/$\overline{\text{INT}}$ — Cause interrupt/No interrupt if 0

CHANNEL: (in Hex)

0–5:	HSO pins 0–5 separately
6:	HSO pins 0 and 1 together
7:	HSO pins 2 and 3 together
8–B:	Software Timers 0–3
C:	HSO pins 0–5 together
D:	Unflagged Event (Do not use for future compatibility)
E:	Reset Timer2
F:	Start A/D Conversion

Figure 10-1. HSO Command Register

the HSO_COMMAND register is shown in Figure 10-1. Command OCH, sets or clears all of the HSO pins, is new on the 80C196KC ODH is reserved for use on future products. Up to eight events can be pending at one time and interrupts can be generated whenever any of these events are triggered. HSO.4 and HSO.5 are bidirectional pins which are multiplexed with HSI.2 and HSI.3 respectively. Bits 4 and 6 of I/O Control Register 1 (IOC1.4, IOC1.6) enable HSO.4 and HSO.5 as outputs. The Control Registers can be read in HWindow 15 to determine the programmed modes for the HSO. However, the IOC2.7(CAM CLEAR) bit is not latched and will read as a one. Entries can be locked in the CAM to generate periodic events or waveforms.

10.1 HSO Interrupts and Software Timers

The HSO unit can generate two types of interrupts. The High Speed Output execution interrupt can be generated (if enabled) for HSO commands which change one or more of the six output pins. The other HSO interrupt can be generated by any other HSO command, (e.g. triggering the A/D, resetting Timer2 or a software Timer Interrupt).

HSO Interrupt Status

Register IOS2 at location 17H displays the HSO events which have occurred. IOS2 is shown in Figure 10-2. The events displayed are HSO.0 through HSO.5, Timer2 Reset and start of an A/D conversion. IOS2 is cleared when accessed. Therefore, the register should be saved in an image register if more than one bit is being tested. Writing to this register in HWindow 15 will set the status bits but not cause interrupts. In HWindow 15, writing to IOS2 can set the High Speed Output lines to an initial value.

SOFTWARE TIMERS

The HSO can be programmed to generate interrupts at preset times. Up to four such "Software Timers" can be in operation at a time. As each preprogrammed time is reached, the HSO unit sets a Software Timer Flag. If the interrupt bit in the HSO command register was set then a Software Timer Interrupt will also be generated. The interrupt service routine can then examine I/O Status register 1 (IOS1) to determine which software timer expired and caused the interrupt. When the HSO resets Timer2 or starts an A/D conversion, it can also generate a software timer interrupt.

If more than one software timer interrupt occurs in the same time, multiple status bits will be set. Each read of IOS1 (see Figure 10-5) will clear bits 0 through 5. Be certain to save the byte before testing it. See also Section 12.5.

10.2 HSO CAM

A block diagram of the HSO unit is shown in Figure 10-3. The Content Addressable Memory (CAM) file is the center of control. One CAM register is compared with the timer values every state time, taking 8 state times to compare all CAM registers with the timers. This defines the resolution of the HSO to be 8 state times (1 microsecond at an oscillator frequency of 16 MHz).

Each CAM register is 24 bits wide. Sixteen bits specify the time the action is to be carried out, and 8 bits define the action to take place. The format of the command to the HSO unit is shown in Figure 10-1. Note that bit 5 is ignored for command channels 8 through 0FH, except for command 0CH.

To enter a command into the CAM file, write the 8-bit "Command Tag" into location 0006H followed by the

IOS2:	7	6	5	4	3	2	1	0
	START A/D	T2 RESET	HSO.5	HSO.4	HSO.3	HSO.2	HSO.1	HSO.0

17H
read

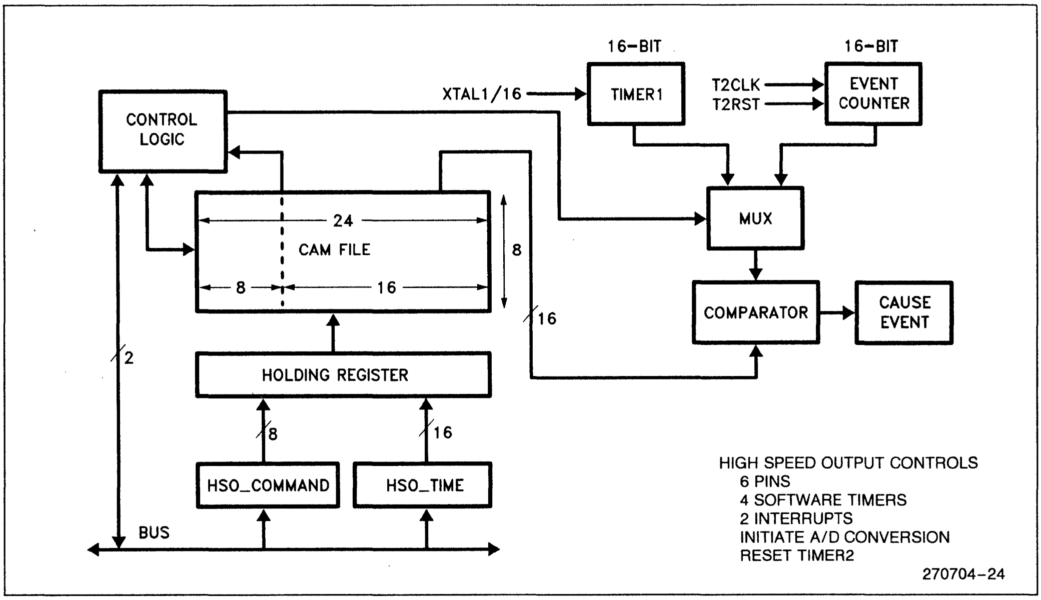
Indicates which HSO event occcured

START A/D: HSO_CMD 15, start A/D

T2RESET: HSO_CMD 14, Timer2 Reset

HSO.0–5: Output pins HSO.0 through HSO.5

Figure 10-2. I/O Status Register 2



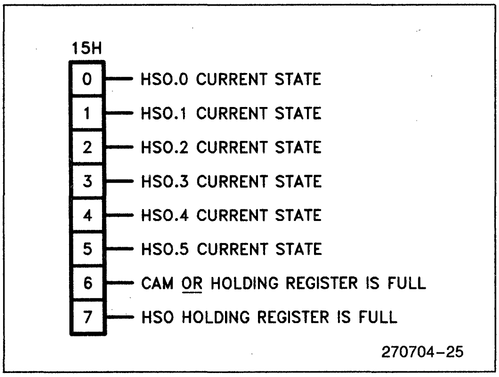
HIGH SPEED OUTPUT CONTROLS
6 PINS
4 SOFTWARE TIMERS
2 INTERRUPTS
INITIATE A/D CONVERSION
RESET TIMER2

270704-24

Figure 10-3. High Speed Output Unit

time the action is to be carried out into word address 0004H. The typical code would be:

```
LDB HSO_COMMAND,#what_to_do
ADD HSO_TIME,Timer1,#when_to_do_it
```



270704-25

Figure 10-4. I/O Status Register 0

Writing the time value loads the HSO Holding Register with both the time and the last written command tag. The command does not actually enter the CAM file until an empty CAM register becomes available.

Commands in the holding register will not execute even if their time tag is reached. Commands must be in the CAM to execute. Commands in the holding register can also be overwritten. Since it can take up to 8 state times for a command to move from the holding register to the CAM, 8 states must be allowed between successive writes to the CAM.

To provide proper synchronization, the minimum time that should be loaded to Timer1 is Timer1 + 2. Smaller values may cause the Timer match to occur 65,636 counts later than expected. A similar restriction applies if Timer2 is used.

Care must be taken when writing the command tag for the HSO, because an interrupt can occur between writing the command tag and loading the time value. If the interrupt service routine writes to the HSO, the command tag used in the interrupt routine will overwrite the command tag from the main routine. One way of avoiding this problem would be to disable interrupts when writing to the HSO unit.

10.3 HSO Status

Before writing to the HSO, ensure that the Holding Register is empty. If it is not, writing to the HSO will overwrite the value in the Holding Register. I/O Status Register 0 (IOS0) bits 6 and 7 indicate the status of the HSO unit. If IOS0.6 equals 0, the holding register is empty and at least one CAM register is empty. If IOS0.7 equals 0, the holding register is empty. The programmer should carefully decide which of these two flags is the best to use for each application. This register also shows the current status of the HSO.0 through HSO.5. The HSO pins can be set by writing to this register in HWindow 15. The format for I/O Status Register 0 is shown in Figure 10-4.

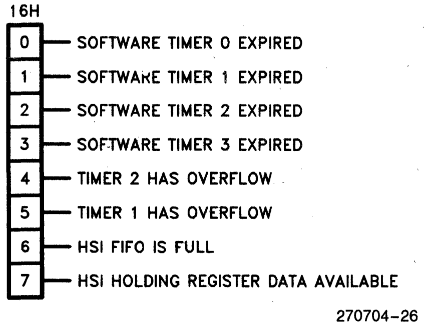

16H
0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE
270704-26

Figure 10-5. I/O Status Register 1 (IOS1)

The expiration of software timer 0 through 4, and the overflow of Timer1 and Timer2 are indicated in IOS1. The status bits can be set in HWindow 15 but not cause interrupts. The register is shown in Figure 10-5.

Whenever the processor reads this register all of the time-related flags (bits 5 through 0) are cleared. This applies not only to explicit reads such as:

```
LDB    AL,IOS1
```

but also to implicit reads such as:

```
JB     IOS1.3,somewhere_else
```

which jumps to somewhere_else if bit 3 of IOS1 is set. In most cases this situation can best be handled by having a byte in the register file which maintains an image of the register. Any time a hardware timer interrupt or a HSO software timer interrupt occurs the byte can be updated:

```
ORB    IOS1_image,IOS1
```

leaving IOS1_image containing all the flags that were set before plus all the new flags that were read and cleared from IOS1. Any other routine which needs to sample the flags can safely check IOS1_image. Note that if these routines need to clear the flags that they have acted on, then the modification of IOS1_image must be done from inside a critical region.

10.4 Clearing the HSO and Locked Entries

All 8 CAM locations of the HSO are compared before any action is taken. This allows a pending external event to be cancelled by simply writing the opposite event to the CAM. However, once an entry is placed in the CAM, it cannot be removed until either the specified timer matches the written value , a chip reset occurs or IOC2.7 is set. IOC2.7 clears all entries in the CAM.

Internal events cannot be cleared by writing an opposite event. This includes events on HSO channels 8-B and E-F. The only method for clearing these events is by a reset or setting IOC2.7.

HSO LOCKED ENTRIES

The CAM Lock bit (HSO_Command.7) can be set to keep commands in the CAM, otherwise the commands will clear from the CAM as soon as they cause an event. This feature allows for generation periodic events based on Timer2 and must be enabled by setting IOC2.6. To clear locked events from the CAM, the entire CAM can be cleared by writing a one to the CAM clear bit IOC2.7. A chip reset will also clear the CAM.

Locked entries are useful in applications requiring periodic or repetitive events. Timer2 used as an HSO reference can generate periodic events with the use of the HSO T2RST command. HSO events programmed with a HSO time less then the Timer2 reset time will occur repeatedly as Timer2 resets. Recurrent software tasks can be scheduled by locking software timers commands into the High Speed Output Unit. Continuous sampling of the A/D converter can be accompished by programming a locked HSO A/D conversion command. One of the most useful features is the generation of multiple PWM's on the High Speed Output lines. Locked entries provide the ability to program periodic events while minimizing software overhead.

Individual external events setting or clearing an HSO pin can by cancelled by writing the opposite event to the CAM. The HSO events do not occur until the timer reference has changed state. An event programmed to set and clear an HSO event at the same time will cancel each other out. Locked entries can correspondingly be cancelled using this method. However, the entries remain in the HSO CAM and can quickly fill up the available eight locations. As an alternative, all entries in the HSO CAM can be cleared by setting IOC2.7.

10.5 HSO Precautions

Timer1 is incremented every 8 state-times. When Timer1 is being used as the reference timer for an HSO command, the comparator has a chance to look at all 8 CAM registers before Timer1 changes its value. Writing to Timer1, which is allowed in HWindow 15, should be carefully done. The user should ensure writing to Timer1 will not cause programmed HSO events to be missed or occur in the wrong order. The same precaution applies to Timer2.

The HSO requires at least eight state times to compare each entry in the CAM. Therefore, the fast increment mode for Timer2 cannot be used as a reference for the HSO if transitions occur faster then once every eight state times.

Referencing events when Timer2 is being used as an up/down counter could cause events to occur in opposite order or be missed entirely. Additionally, locked entries could occur several times if Timer2 is oscillating around the time tag for an entry.

When using Timer2 as the HSO reference, caution must be taken that Timer2 is not reset prior to the highest value for a Timer2 match in the CAM. If that match is never reached, the event will remain in the CAM until the device is reset or CAM is cleared.

10.6 HSO Output Timing

Changes in the HSO lines are synchronized to Timer1 or Timer2. All of the external HSO lines due to change at a certain value of a timer will change just after the incrementing of the timer. Internally, the timer changes every eight state times during Phase1. From an external perspective the HSO pin should change just prior to the falling edge of CLKOUT and be stable by its rising edge. Information from the HSO can be latched on the CLKOUT rising edge. Internal events also occur when the reference timer increments.

11.0 SERIAL PORT

The serial port has one synchronous and 3 asynchronous modes. The asynchronous modes are full duplex, meaning they can transmit and receive at the same time. The receiver is double buffered so that the reception of a second byte can begin before the first byte has been read. The transmitter on the 80C196KC is also double buffered allowing continuous transmissions. The port is functionally compatible with the serial port on the MCS-51 family of microcontrollers, although the software controlling the ports is different.

Data to and from the serial port is transferred through SBUF(RX) and SBUF(TX), both located at 07H. SBUF(TX) holds data ready for transmission and SBUF(RX) contains data received by the serial port. SBUF(TX) and SBUF(RX) can be read and written in HWindow 15.

Mode 0, the synchronous shift register mode, is designed to expand I/O over a serial line. Mode 1 is the standard 8 bit data asynchronous mode used for normal serial communications. Modes 2 and 3 are 9 bit data asynchronous modes typically used for interprocessor communications.

11.1 Serial Port Status and Control

Control of the serial port is done through the Serial Port Control (SP_CON) register shown in Figure 11-1. Writing to location 11H accesses SP_CON while reading it accesses SP_STAT. The upper 3 bits of SP_CON must be written as 0s for future compatibility. On the 80C196KC the SP_STAT register contains bits to indicate receive Overrun Error (OE), Framing Error (FE), and Transmitter Empty (TXE). The bits which were also present on the 8096BH are the Transmit Interrupt (TI) bit, the Receive Interrupt (RI) bit, and the Received Bit 8 (RB8) or Receive Parity Error (RPE) bit. SP_STAT is read-only in HWindow 0 and is shown in Figure 11-1.

In all modes, the RI flag is set after the last data bit is sampled, approximately in the middle of a bit time. Data is held in the receive shift register until the last data bit is received, then the data byte is loaded into SBUF (RX). The receiver on the 80C196KB also checks for a valid stop bit. If a stop bit is not found within the appropriate time, the Framing Error (FE) bit is set.

Since the receiver is double-buffered, reception on a second data byte can begin before the first byte is read. However, if data in the shift register is loaded into SBUF (RX) before the previous byte is read, the Overflow Error (OE) bit is set. Regardless, the data in SBUF (RX) will always be the latest byte received; it will never be a combination of the two bytes. The RI, FE, and OE flags are cleared when "SP_STAT" is read. However, RI does not have to be cleared for the serial port to receive data.

SP_CON:	7	6	5	4	3	2	1	0	
	X	X	X	TB8	REN	PEN	M2	M1	11H

TB8 — Sets the ninth data bit for transmission. Cleared after each transmission. Not valid if parity is enabled.

REN — Enables the receiver

PEN — Enables the Parity function (even parity)

M2, M1 — Sets the mode. Mode0 = 00, Model = 01, Mode2 = 10, Mode3 = 11

SP_STAT	7	6	5	4	3	2	1	0	
	RB8/ RPE	RI	TI	FE	TXE	OE	X	X	11H

RB8 — Set if the 9th data bit is high on reception (parity disabled)

RPE — Set if parity is enabled and a parity error occurred

RI — Set when the last data bit is sampled

TI — Set at the beginning of the STOP bit transmission

FE — Set if no STOP bit is found at the end of a reception

TXE — Set if two bytes can be transmitted

OE — Set if receiver is overwritten

Figure 11-1. Serial Port Control and Status Registers

The Transmitter Empty (TXE) bit is set if the transmit buffer is empty and ready to take up to two characters. TXE gets cleared as soon as a byte is written to SBUF. Two bytes may be written consecutively to SBUF if TXE is set. One byte may be written if TI alone is set. By definition, if TXE has just been set, a transmission has completed and TI will be set. The TI bit is reset when the CPU reads the SP_STAT registers.

The TB8 bit is cleared after each transmission and both TI and RI are cleared when SP_STAT is read. The RI and TI status bits can be set by writing to SP_STAT in HWindow 15 but they will not cause an interrupt. Reading of SP_CON in HWindow 15 will read the last value written. Whenever the TXD pin is used for the serial port it must be enabled by setting IOC1.5 to a 1. IOC1 register 1 can be read in HWindow 15 to determine the setting.

STARTING TRANSMISSIONS AND RECEPTIONS

In Mode 0, if REN = 0, writing to SBUF (TX) will start a transmission. A rising edge on REN, or clearing RI with REN = 1, will start a reception. Setting REN = 0 will stop a reception in progress and inhibit further receptions. To avoid a partial reception, REN must be set to zero before RI is cleared. This can be handled in an interrupt environment by using software flags or in straight-line code by using the Interrupt Pending register to signal the completion of a reception.

In the asynchronous modes, writing to SBUF (TX) starts a transmission. A falling edge on RXD will begin a reception if REN is set to 1. New data placed in SBUF (TX) is held and will not be transmitted until the end of the stop bit has been sent.

In all modes, the RI flag is set after the last data bit is sampled approximately in the middle of the bit time. For all modes, the TI flag is set after the last data bit (either 8th or 9th) is sent, also in the middle of the bit time. The flags clear when SP_STAT is read, but do not have to be clear for the port to receive or transmit. The serial port interrupt bit is set as a logical OR of the RI and TI bits. Note that changing modes will reset the Serial Port and abort any transmission or reception in progress on the channel.

DETERMINING BAUD RATES

Baud rates in all modes are determined by the contents of a 8-bit register at location 000EH. Reading or writing this register in HWindow 15 is reserved by Intel for future use. This register must be loaded sequentially with 2 bytes (least significant byte first). The MSB of this register selects one of two sources for the input frequency to the baud rate generator. If it is a 1, the XTAL1 pin is selected, if not, the T2CLK pin is used. The maximum input frequency is 4 MHz on T2CLK.

This provides the needed synchronization to the internal serial port clocks.

The unsigned integer represented by the lower 15 bits of the baud rate register defines a number B, where B has a maximum value of 32767. The baud rate equations are shown below.

Asynchronous Modes 1, 2 and 3:

$$\text{BAUD\_REG} = \frac{\text{XTAL1}}{\text{Baud Rate * 16}} - 1 \text{ OR } \frac{\text{T2CLK}}{\text{Baud Rate * 8}}$$

Synchronous Mode 0:

$$\text{BAUD\_REG} = \frac{\text{XTAL1}}{\text{Baud Rate * 2}} - 1 \text{ OR } \frac{\text{T2CLK}}{\text{Baud Rate}}$$

Note that B cannot equal 0, except when using XTAL1 and not in mode 0.

Common baud rate values, using XTAL1 at 16 MHz, are shown below.

Baud Rate	Baud Register Value	
	Mode 0	Others
9600	8340H	8067H
4800	8682H	80CFH
2400	8D04H	81A0H
1200	9A0AH	8340H
300	E82BH	8D04H

The maximum baud rates are 4.0 Mbaud synchronous and 1.0 Mbaud asynchronous with 16 MHz on XTAL1.

11.2 Serial Port Interrupts

The serial port generates one of three possible interrupts: Transmit Interrupt TI(2030H), Receive Interrupt RI(2032H) and SERIAL(200CH). When the RI bit gets set an interrupt is generated through either 200CH or 2032H depending on which interrupt is enabled. INT_MASK1.1 controls the serial port receive interrupt through location 2032H and INT_MASK.6 controls the RI interrupt through location 200CH. The 8096BH shared the TI and RI interrupts on the SERIAL interrupt vector. On the 80C196KC, these interrupts share both the serial interrupt vector and have their own interrupt vectors. Because the interrupts now have separate vectors, you do not have to sort the interrupts out in the same interrupt service routine.

When the TI bit is set it can cause an interrupt through the vectors at locations 200CH or 2030H. Interrupt through location 2030H is determined by INT_MASK1.0. Interrupts through the Serial interrupt are controlled by the same bit as the RI interrupt (INT_MASK.6).

11.3 Serial Port Modes

MODE 0

Mode 0 is a synchronous mode and is commonly used for shift register based I/O expansion. In this mode the TXD pin outputs a set of 8 pulses while the RXD pin either transmits or receives data. Data is transferred 8 bits at a time with the LSB first. A diagram of the relative timing of these signals is shown in Figure 11-2. This is the only mode which uses RXD as an output.

Mode 0 Timings

In Mode 0, the TXD pin sends out a clock train, while the RXD pin transmits or receives the data. Figure 11-2 shows the waveforms and timing.

In this mode the serial port expands the I/O capability of the 80C196KC by simply adding shift registers. A schematic of a typical circuit is shown in Figure 11-3. This circuit inverts the data coming in, so it must be reinverted in software.

MODE 1

Mode 1 is the standard asynchronous communications mode. The data frame used in this mode is shown in Figure 11-4. It consists of 10 bits; a start bit (0), 8 data bits (LSB first), and a stop bit (1). If parity is enabled by setting SPCON.2, an even parity bit is sent instead of the 8th data bit and parity is checked on reception.

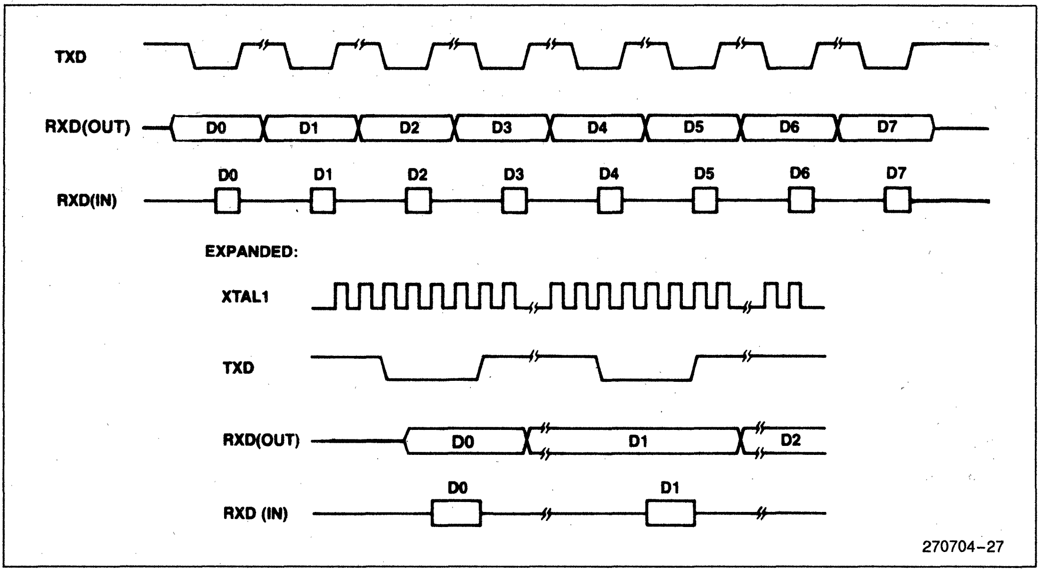


Figure 11-2. Serial Port Mode 0 Timing

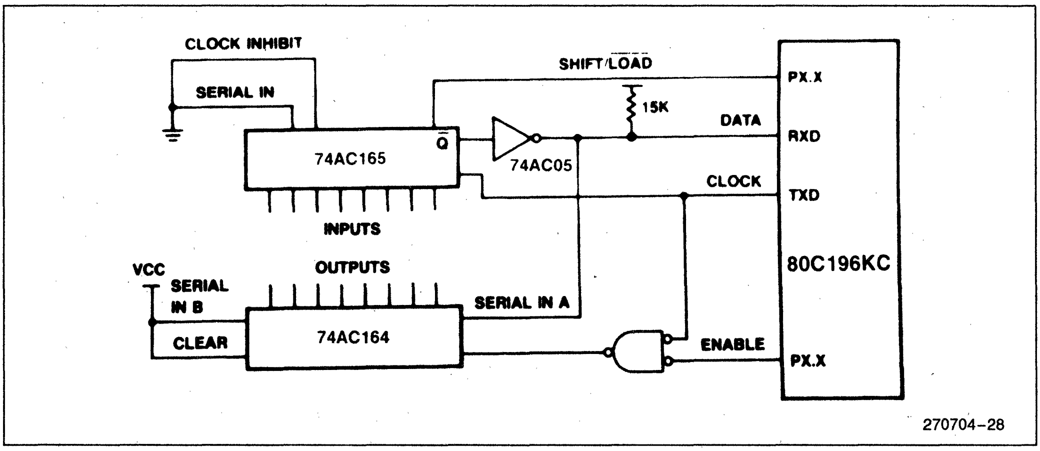


Figure 11-3. I/O Expansion in Mode 0

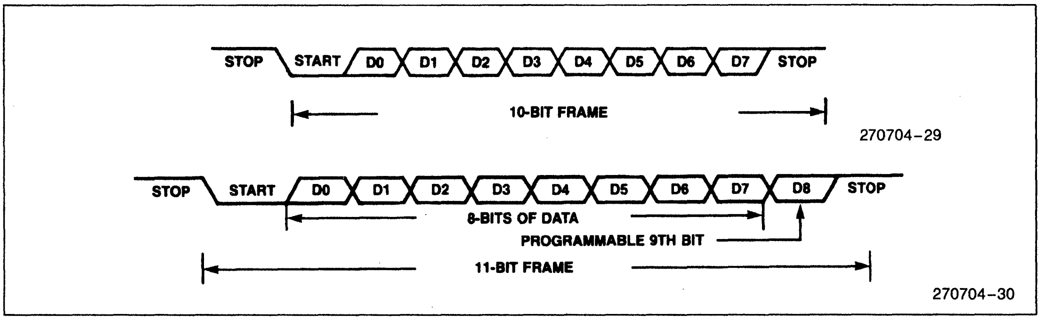


Figure 11-4. Mode 1, 2, and 3 Timing

The transmit and receive functions are controlled by separate shift clocks. The transmit shift clock starts when the baud rate generator is initialized, the receive shift clock is reset when a '1 to 0' transition (start bit) is received. The transmit clock may therefore not be in sync with the receive clock, although they will both be at the same frequency.

The TI (Transmit Interrupt) and RI (Receive Interrupt) flags are set to indicate when operations are complete. TI is set when the last data bit of the message has been sent, not when the stop bit is sent. If an attempt to send another byte is made before the stop bit is sent the port will hold off transmission until the stop bit is complete. RI is set when 8 data bits are received, not when the stop bit is received. Note that when the serial port status register is read both TI and RI are cleared.

Caution should be used when using the serial port to connect more than two devices in half-duplex, (i.e. one wire for transmit *and* receive). If the receiving processor does not wait for one bit time after RI is set before starting to transmit, the stop bit on the link could be corrupted. This could cause a problem for other devices listening on the link.

MODE 2

Mode 2 is the asynchronous 9th bit recognition mode. This mode is commonly used with Mode 3 for multiprocessor communications. Figure 11-4 shows the data frame used in this mode. It consists of a start bit (0), 9 data bits (LSB first), and a stop bit (1). When transmitting, the 9th bit can be set to a one by setting the TB8 bit in the control register before writing to SBUF (TX). The TB8 bit is cleared on every transmission. During reception, the serial port interrupt and the Receive Interrupt will not happen unless the 9th bit being received is set. This provides an easy way to have selective reception on a data link. Parity cannot be enabled in this mode.

MODE 3

Mode 3 is the asynchronous 9th bit mode. The data frame for this mode is identical to that of Mode 2. The transmission differences between Mode 3 and Mode 2 are that parity can be enabled (PEN = 1) and cause the 9th data bit to take the even parity value. The TB8 bit can still be used if parity is not enabled (PEN = 0). When in Mode 3, a reception always causes an interrupt, regardless of the state of the 9th bit. The 9th bit is stored if PEN = 0 and can be read in bit RB8. If PEN = 1 then RB8 becomes the Receive Parity Error (RPE) flag.

11.4 Multiprocessor Communications

Mode 2 and 3 are provided for multiprocessor communications. In Mode 2 if the received 9th data bit is zero, the RI bit is not set, and will not cause an interrupt. In Mode 3, the RI bit is set and always causes an interrupt regardless of the value in the 9th bit. The way to use this feature in multiprocessor systems is described below.

The master processor is set to Mode 3 so it always gets interrupts from serial receptions. The slaves are set in Mode 2 so they only have receive interrupts if the 9th bit is set. Two types of frames are used: address frames which have the 9th bit set and data frames which have the 9th bit cleared. When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address frame which identifies the target slave. Each slave can examine the received byte and see if it is being addressed. The addressed slave switches to Mode 3 to receive the coming data frames, while the slaves that were not addressed stay in Mode 2 continue executing.

12.0 A/D CONVERTER

Analog Inputs to the 80C196KC System are handled by the A/D converter System. As shown in Figure 12-1, the converter system has an 8 channel multiplexer, a sample-and-hold, and a 10-bit successive approximation A/D converter. Conversions can be performed on one of eight channels, the inputs of which share pins with port 0.

The A/D converter on the 80C196KC has many improvements over the 80C196KB converter. The converter can perform either 8- or 10-bit conversions. By performing an 8-bit conversion, resolution is traded off for a shorter conversion time. A significant improvement to the A/D Converter is the Sample Window and the Conversion time are programmable in state times.

Conversions are started by loading the AD_COMMAND with the channel number, and whether an 8- or 10-bit conversion is performed, as shown in Figure 12-2. The conversion can be started immediately by setting the GO bit to a 1. If the GO bit is set to 0, the conversion will start when triggered by the HSO. The result and status of the conversion is read in the AD_RESULT (High) and AD_RESULT (Low) registers, as shown in Figure 12-3. The AD_RESULT register can be accessed as a byte or word.

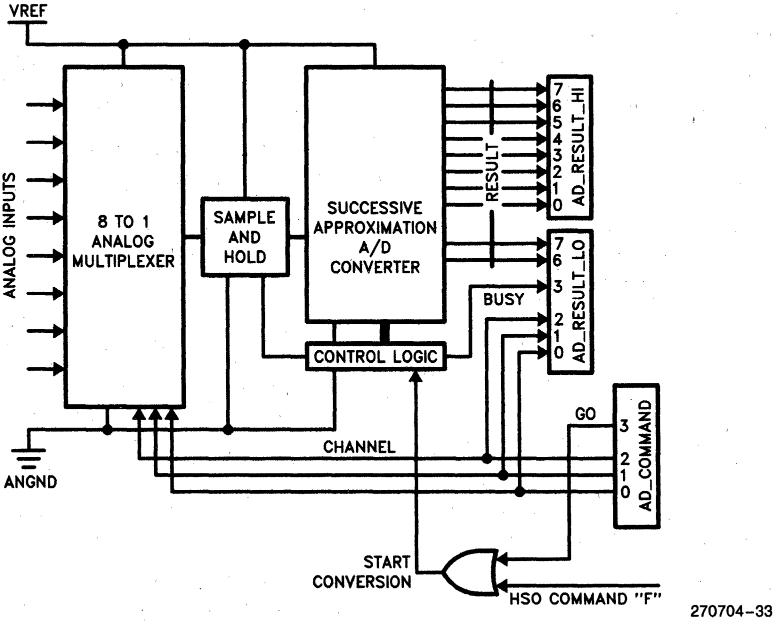


Figure 12-1. A/D Converter Block Diagram

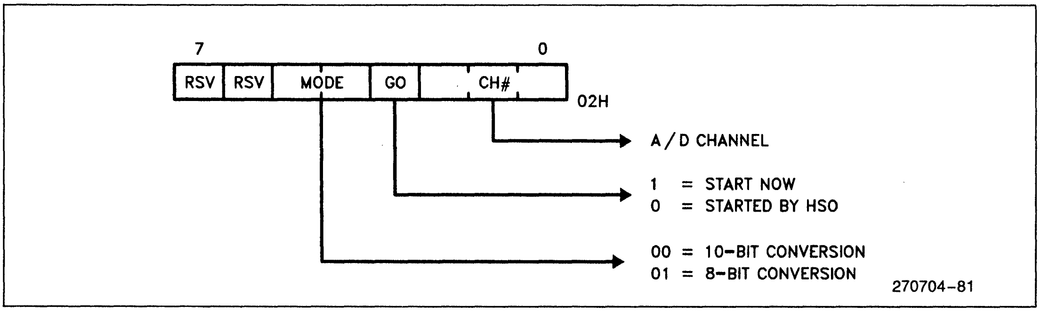


Figure 12-2. A/D_COMMAND Register

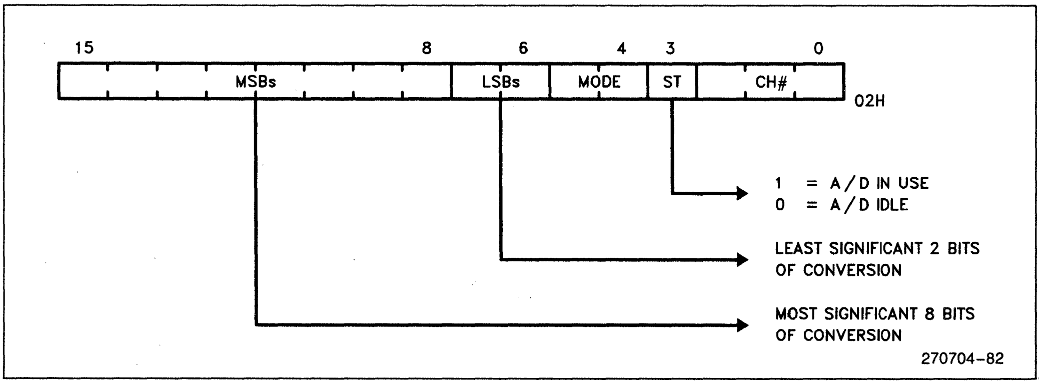
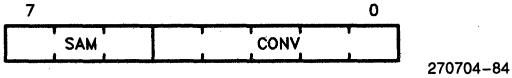

15
8
6
4
3
0
MSBs
LSBs
MODE
ST
CH#
02H
1 = A / D IN USE
0 = A / D IDLE
LEAST SIGNIFICANT 2 BITS OF CONVERSION
MOST SIGNIFICANT 8 BITS OF CONVERSION
270704-82

Figure 12-3. A/D_RESULT Register

Programmable Sample and Convert Times

There are two parameters that define the time an A/D conversion will take; the Sample Time plus the Convert time. The Sample time is the time the analog input channel is actually connected to the Sample Capacitor. If this time is too short, the Sample capacitor will not charge properly. If the Sample time is too long, the input may change and errors occur. The Convert time is defined to be the length of time to convert one bit of the analog voltage on the Sample Capacitor to a digital value. The Convert time has to be long enough for the comparator to settle and resolve the voltage, but short enough so the Sample Capacitor will not discharge and lose resolution.

Because the 80C196KC can run from 3.5 MHz to 16 MHz, it is difficult to optimize both the Sample and Convert times using only the fast and normal conversion modes on the 80C196KB. Therefore, an A/D_TIME register in HWINDOW 1 was added so both the Sample and Convert times could be programmed in number of state times. The fast and normal conversions are still present to remain compatible with the 80C196KB. Figure 12-5 shows the A/D_TIME register and the equations for calculating the number of state times for an A/D conversion. IOC2 is shown in Figure 12-6 which enables the different conversion times.



For 8 and 10 Bit Conversions $T = 4 * SAM + B * (CONV + 1) + 2.5$

T = Number of States Times

B = Number of Bits (8 or 10)

80C196KB Compatible Mode Times

Prescalar On	156.5 States
Prescalar Off	89.5 States

Figure 12-5. AD_TIME Register

The A/D__TIME register only programs the speed the A/D can run, NOT the speed it can convert correctly. The 80C196KC data sheet will contain the correct values for the Sample and Convert times in microseconds.

Restrictions on the A/D Converter

1. For an A/D conversion, initialize the A/D registers in the following order; AD__TIME, IOC2, and AD__COMMAND.
2. Do not start a conversion using the AD__TIME register when a conversion using a 80C196KB compatible mode is in progress (and VICE-VERSA).
3. Never write zero to the AD__TIME register.

12.1 A/D Conversion Process

The conversion process is initiated by the execution of HSO command 0FH, or by writing a one to the GO Bit in the A/D Control Register. Either activity causes a start conversion signal to be sent to the A/D converter control logic. If an HSO command was used, the conversion will begin when Timer1 increments. This aids applications attempting to approach spectrally pure sampling, since successive samples spaced by equal Timer1 delays will occur with a variance of about ± 50 ns (assuming a stable clock on XTAL1). However, conversions initiated by writing a one to the ADCON register GO Bit will start within three state times after the instruction has completed execution resulting in a variance of about 0.38 μs (XTAL1 = 16 MHz).

To perform the actual analog-to-digital conversion the 80C196KC implements a successive approximation algorithm. The converter hardware consists of a 256-resistor ladder, a comparator, coupling capacitors and a 10-bit successive approximation register (SAR) with logic that guides the process. The resistor ladder provides 20 mV steps (V_{REF} = 5.12V), while capacitive coupling creates 5 mV steps within the 20 mV ladder voltages. Therefore, 1024 internal reference voltages are available for comparison against the analog input to generate a 10-bit conversion result. For an 8-bit conversion, there are 256 levels.

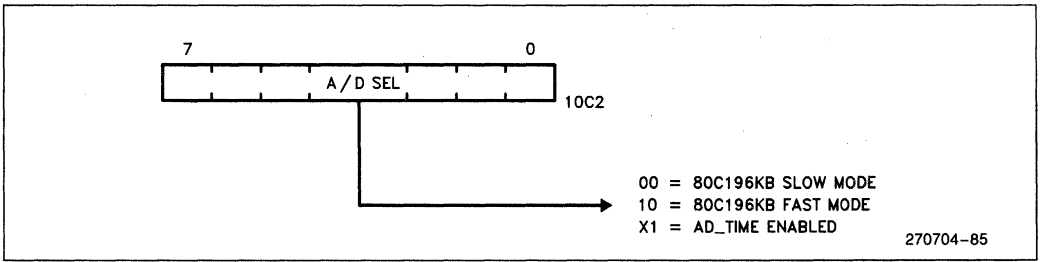


Figure 12-6. IOC2

A successive approximation conversion is performed by comparing a sequence of reference voltages, to the analog input, in a binary search for the reference voltage that most closely matches the input. The ½ full scale reference voltage is the first tested. This corresponds to a 10-bit result where the most significant bit is zero, and all other bits are ones (0111.1111.11b). If the analog input was less than the test voltage, bit 10 of the SAR is left a zero, and a new test voltage of ¼ full scale (0011.1111.11b) is tried. If this test voltage was lower than the analog input, bit 9 of the SAR is set and bit 8 is cleared for the next test (0101.1111.11b). This binary search continues until 10 or 8 tests have occurred, at which time the valid 10-bit or 8-bit conversion result resides in the SAR where it can be read by software.

12.2 A/D Interface Suggestions

The external interface circuitry to an analog input is highly dependent upon the application, and can impact converter characteristics. In the external circuit's design, important factors such as input pin leakage, sample capacitor size and multiplexer series resistance from the input pin to the sample capacitor must be considered. The following calculation assumes a 1 μs Sample Window.

For the 80C196KC, these factors are idealized in Figure 12-7. The external input circuit must be able to charge a sample capacitor (C_S) through a series resistance (R_I) to an accurate voltage given a D.C. leakage (I_L). On the 80C196KC, C_S is around 2 pF, R_I is around ±5 KΩ and I_L is specified as 3 μA maximum. In determining the necessary source impedance R_S, the value of V_{BIAS} is not important.

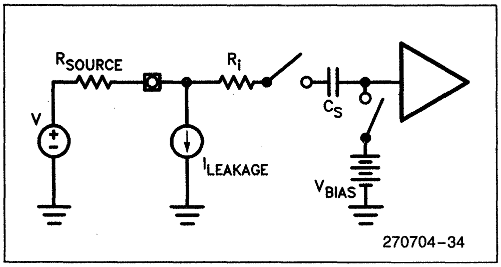


Figure 12-7. Idealized A/D Sampling Circuitry

External circuits with source impedances of 1 KΩ or less will be able to maintain an input voltage within a tolerance of about ±0.61 LSB (1.0 KΩ × 3.0 μA= 3.0 mV) given the D.C. leakage. Source impedances above 2 KΩ can result in an external error of at least one LSB due to the voltage drop caused by the 3 μA leakage. In addition, source impedances above 25 KΩ may degrade converter accuracy as a result of the internal sample capacitor not being fully charged during the sample window.

If large source impedances degrade converter accuracy because the sample capacitor is not charged during the sample time, an external capacitor connected to the pin compensates for this degradation. Since the sample capacitor is 2 pF, a 0.005 μF capacitor will charge the sample capacitor to an accurate input voltage of ±0.5LSB (2048 * 2 pF). An external capacitor does not compensate for the voltage drop across the source resistance, but charges the sample capacitor fully during the sample time.

Placing an external capacitor on each analog input will also reduce the sensitivity to noise, as the capacitor combines with series resistance in the external circuit to form a low-pass filter. In practice, one should include a small series resistance prior to the external capacitor on the analog input pin and choose the largest capacitor value practical, given the frequency of the signal being converted. This provides a low-pass filter on the input, while the resistor will also limit input current during over-voltage conditions.

Figure 12-8 shows a simple analog interface circuit based upon the discussion above. The circuit in the figure also provides limited protection against over-voltage conditions on the analog input. Should the input voltage inappropriately drop significantly below ground, diode D2 will forward bias at about 0.8 DCV. This will limit the current sourced by the input pin to an acceptable amount. *However, before any circuit is used in an actual application, it should be thoroughly analyzed for applicability to the specific problem at hand.*

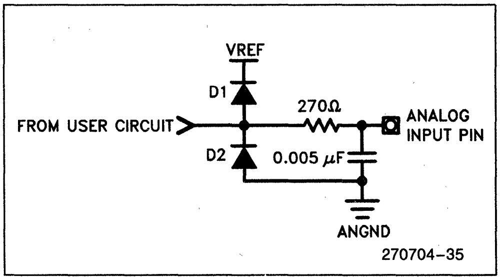


Figure 12-8. Suggested A/D Input Circuit

ANALOG REFERENCES

Reference supply levels strongly influence the absolute accuracy of the conversion. Bypass capacitors should be used between V_{REF} and ANGND. ANGND should be within about a tenth of a volt of V_{SS}. V_{REF} should be well regulated and used only for the A/D converter. The V_{REF} supply can be between 4.5V and 5.5V and needs to be able to source around 5 mA. See Section 13 for the minimum hardware connections.

Note that if only ratiometric information is desired, V_{REF} can be connected to V_{CC}. In addition, V_{REF} and ANGND must be connected even if the A/D converter is not being used. Remember that Port 0 receives its power from the V_{REF} and ANGND pins even when it is used as digital I/O.

12.3 The A/D Transfer Function

The conversion result is a 10-bit ratiometric representation of the input voltage, so the numerical value obtained from the conversion will be:

$$\text{INT } [1023 \times (V_{IN} - \text{ANGND})/(V_{REF} - \text{ANGND})].$$

This produces a stair-stepped transfer function when the output code is plotted versus input voltage (see Figure 12-9). The resulting digital codes can be taken as simple ratiometric information, or they provide information about absolute voltages or relative voltage changes on the inputs. The more demanding the application is on the A/D converter, the more important it is to fully understand the converter's operation. For simple applications, knowing the absolute error of the converter is sufficient. However, closing a servo-loop with analog inputs necessitates a detailed understanding of an A/D converter's operation and errors.

The errors inherent in an analog-to-digital conversion process are many: quantizing error, zero offset, full-scale error, differential non-linearity, and non-linearity. These are "transfer function" errors related to the A/D converter. In addition, converter temperature drift, V_{CC} rejection, sample-hold feedthrough, multiplexer off-isolation, channel-to-channel matching and random noise should be considered. Fortunately, one "Absolute Error" specification is available which describes the sum total of all deviations between the actual conversion process and an ideal converter. However, the various sub-components of error are important in many applications. These error components are described in Section 12.5 and in the text below where ideal and actual converters are compared.

An unavoidable error simply results from the conversion of a continuous voltage to an integer digital representation. This error is called quantizing error, and is always ±0.5 LSB. Quantizing error is the only error seen in a perfect A/D converter, and is obviously present in actual converters. Figure 12-9 shows the transfer function for an ideal 3-bit A/D converter (i.e. the Ideal Characteristic).

Note that in Figure 12-9 the Ideal Characteristic possesses unique qualities: it's first code transition occurs when the input voltage is 0.5 LSB; it's full-scale code transition occurs when the input voltage equals the full-scale reference minus 1.5 LSB; and it's code widths are all exactly one LSB. These qualities result in a digitization without offset, full-scale or linearity errors. In other words, a perfect conversion.

Figure 12-10 shows an Actual Characteristic of a hypothetical 3-bit converter, which is not perfect. When the Ideal Characteristic is overlaid with the imperfect characteristic, the actual converter is seen to exhibit errors in the location of the first and final code transitions and code widths. The deviation of the first code transition from ideal is called "zero offset", and the deviation of the final code transition from ideal is "full-scale error". The deviation of the code widths from ideal causes two types of errors. Differential Non-Linearity and Non-Linearity. Differential Non-Linearity is a local linearity error measurement, whereas Non-Linearity is an overall linearity error measure.

Differential Non-Linearity is the degree to which actual code widths differ from the ideal one LSB width. It gives the user a measure of how much the input voltage may have changed in order to produce a one count change in the conversion result. Non-Linearity is the worst case deviation of code transitions from the corresponding code transitions of the Ideal Characteristic. Non-Linearity describes how much Differential Non-Linearities could add up to produce an overall maximum departure from a linear characteristic. If the Differential Non-Linearity errors are too large, it is possible for an A/D converter to miss codes or exhibit non-monotonicity. Neither behavior is desirable in a closed-loop system. A converter has no missed codes if there exists for each output code a unique input voltage range that produces that code only. A converter is monotonic if every subsequent code change represents an input voltage change in the same direction.

Differential Non-Linearity and Non-Linearity are quantified by measuring the Terminal Based Linearity Errors. A Terminal Based Characteristic results when an Actual Characteristic is shifted and rotated to eliminate zero offset and full-scale error (see Figure 12-11). The Terminal Based Characteristic is similar to the Ac-

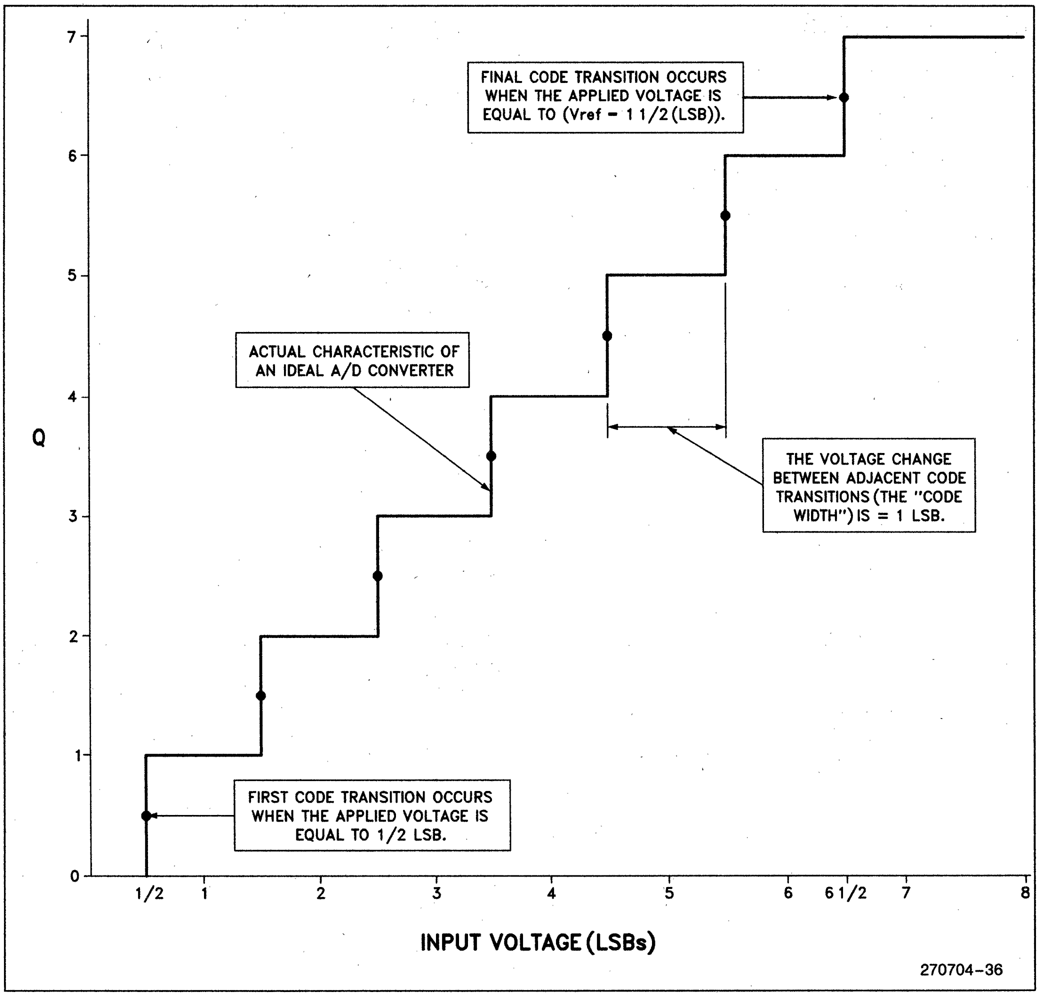


Figure 12-9. Ideal A/D Characteristic

tual Characteristic that would be seen if zero offset and full-scale error were externally trimmed away. In practice, this is done by using input circuits which include gain and offset trimming. In addition, V_{REF} on the 80C196KC could also be closely regulated and trimmed within the specified range to affect full-scale error.

Other factors that affect a real A/D Converter system include sensitivity to temperature, failure to completely reject all unwanted signals, multiplexer channel dissimilarities and random noise. Fortunately these effects are small.

Temperature sensitivities are described by the rate at which typical specifications change with a change in temperature.

Undesired signals come from three main sources. First, noise on V_{CC}—V_{CC} Rejection. Second, input signal changes on the channel being converted after the sample window has closed—Feedthrough. Third, signals applied to channels not selected by the multiplexer—Off-Isolation.

Finally, multiplexer on-channel resistances differ slightly from one channel to the next causing Channel-to-Channel Matching errors, and random noise in general results in Repeatability errors.

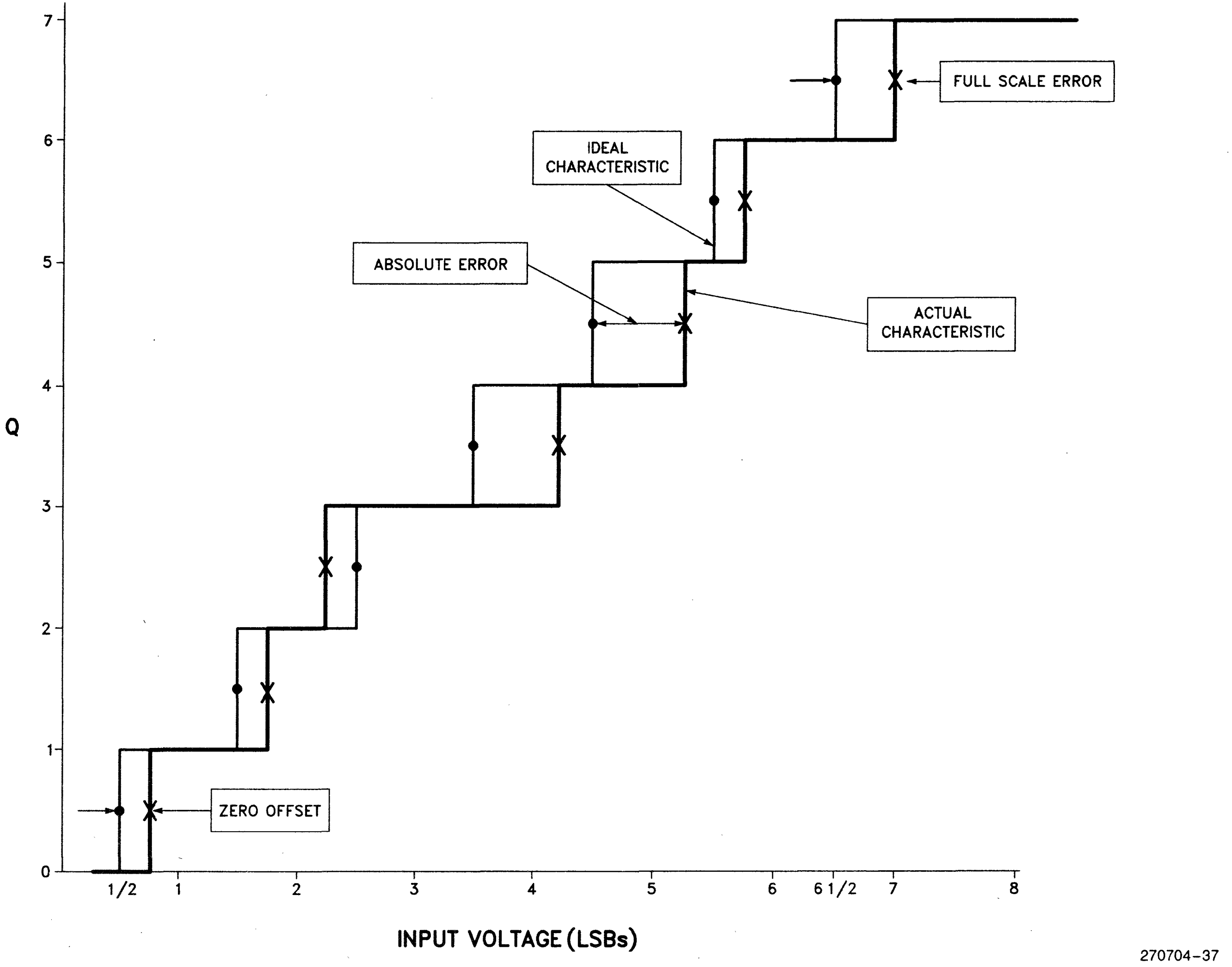
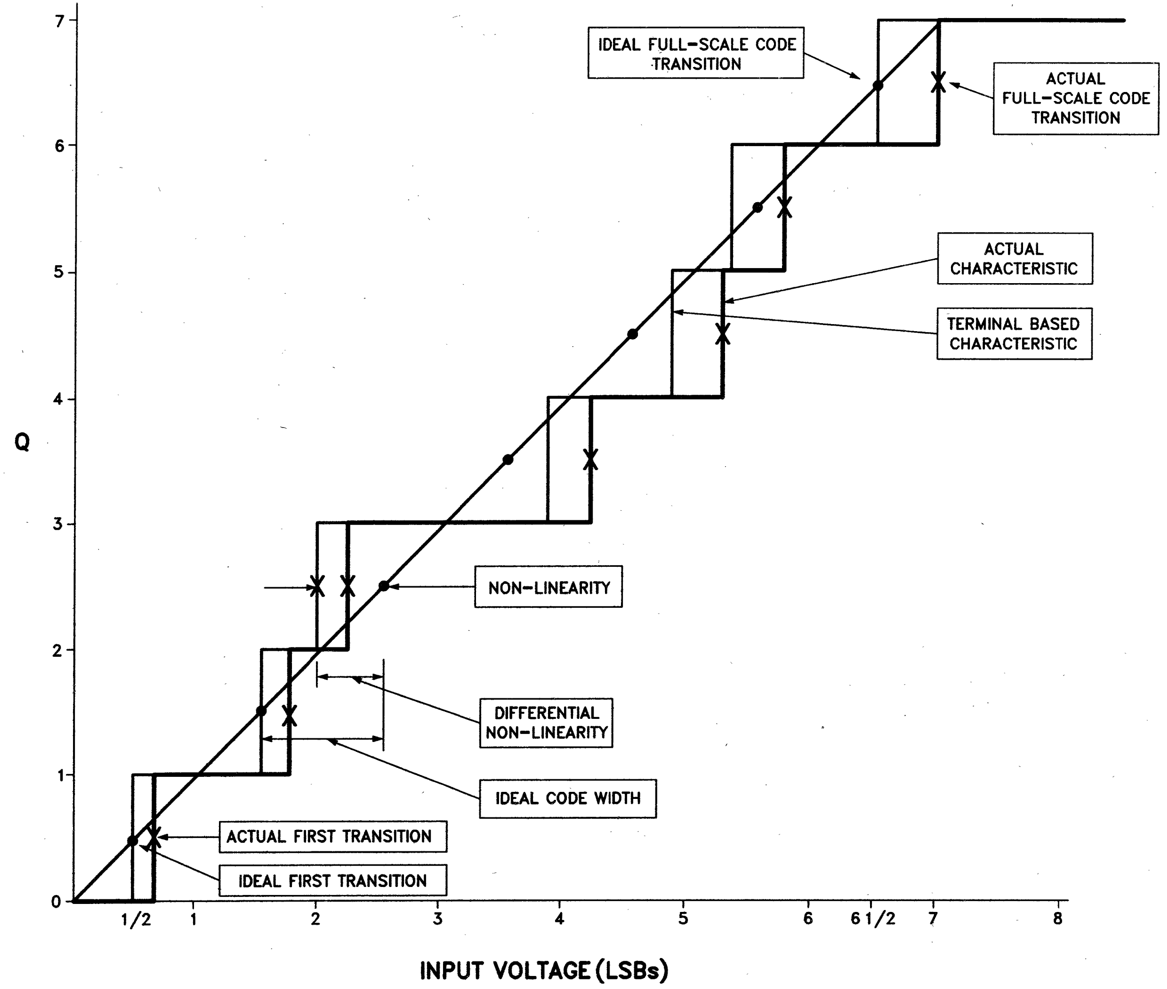


Figure 12-10. Actual and Ideal Characteristics



270704-38

Figure 12-11. Terminal Based Characteristic

12.4 A/D Glossary of Terms

Figures 12-9, 12-10, and 12-11 display many of these terms. Refer to AP-406 'MCS-96 Analog Acquisition Primer' for additional information on the A/D terms.

ABSOLUTE ERROR—The maximum difference between corresponding actual and ideal code transitions. Absolute Error accounts for all deviations of an actual converter from an ideal converter.

ACTUAL CHARACTERISTIC—The characteristic of an actual converter. The characteristic of a given converter may vary over temperature, supply voltage, and frequency conditions. An Actual Characteristic rarely has ideal first and last transition locations or ideal code widths. It may even vary over multiple conversion under the same conditions.

BREAK-BEFORE-MAKE—The property of a multiplexer which guarantees that a previously selected channel will be deselected before a new channel is selected. (e.g. the converter will not short inputs together.)

CHANNEL-TO-CHANNEL MATCHING—The difference between corresponding code transitions of actual characteristics taken from different channels under the same temperature, voltage and frequency conditions.

CHARACTERISTIC—A graph of input voltage versus the resultant output code for an A/D converter. It describes the transfer function of the A/D converter.

CODE—The digital value output by the converter.

CODE CENTER—The voltage corresponding to the midpoint between two adjacent code transitions.

CODE TRANSITION—The point at which the converter changes from an output code of Q, to a code of Q+1. The input voltage corresponding to a code transition is defined to be that voltage which is equally likely to produce either of two adjacent codes.

CODE WIDTH—The voltage corresponding to the difference between two adjacent code transitions.

CROSSTALK—See "Off-Isolation".

D.C. INPUT LEAKAGE—Leakage current to ground from an analog input pin.

DIFFERENTIAL NON-LINEARITY—The difference between the ideal and actual code widths of the terminal based characteristic of a converter.

FEEDTHROUGH—Attenuation of a voltage applied on the selected channel of the A/D converter after the sample window closes.

FULL SCALE ERROR—The difference between the expected and actual input voltage corresponding to the full scale code transition.

IDEAL CHARACTERISTIC—A characteristic with its first code transition at V_{IN} = 0.5 LSB, its last code transition at V_{IN} = (V_{REF} − 1.5 LSB) and all code widths equal to one LSB.

INPUT RESISTANCE—The effective series resistance from the analog input pin to the sample capacitor.

LSB—LEAST SIGNIFICANT BIT: The voltage value corresponding to the full scale voltage divided by 2^n, where n is the number of bits of resolution of the converter. For a 10-bit converter with a reference voltage of 5.12 volts, one LSB is 5.0 mV. For an 8-bit conversion, one LSB equals 20 mV. Note that this is different than digital LSBs.

MONOTONIC—The property of successive approximation converters which guarantees that increasing input voltages produce adjacent codes of increasing value, and that decreasing input voltages produce adjacent codes of decreasing value.

NO MISSED CODES—For each and every output code, there exists a unique input voltage range which produces that code only.

NON-LINEARITY—The maximum deviation of code transitions of the terminal based characteristic from the corresponding code transitions of the ideal characteristics.

OFF-ISOLATION—Attenuation of a voltage applied on a deselected channel of the A/D converter. (Also referred to as Crosstalk.)

REPEATABILITY—The difference between corresponding code transitions from different actual characteristics taken from the same converter on the same channel at the same temperature, voltage and frequency conditions.

RESOLUTION—The number of input voltage levels that the converter can unambiguously distinguish between. Also defines the number of useful bits of information which the converter can return.

SAMPLE DELAY—The delay from receiving the start conversion signal to when the sample window opens.

SAMPLE DELAY UNCERTAINTY—The variation in the Sample Delay.

SAMPLE TIME—The time that the sample window is open.

SAMPLE TIME UNCERTAINTY—The variation in the sample time.

SAMPLE WINDOW—Begins when the sample capacitor is attached to a selected channel and ends when the sample capacitor is disconnected from the selected channel.

SUCCESSIVE APPROXIMATION—An A/D conversion method which uses a binary search to arrive at the best digital representation of an analog input.

TEMPERATURE COEFFICIENTS—Change in the stated variable per degree centigrade temperature change. Temperature coefficients are added to the typical values of a specification to see the effect of temperature drift.

TERMINAL BASED CHARACTERISTIC—An Actual Characteristic which as been rotated and translated to remove zero offset and full-scale error.

V_{CC} REJECTION—Attenuation of noise on the V_{CC} line to the A/D converter.

ZERO OFFSET—The difference between the expected and actual input voltage corresponding to the first code transition.

13.0 I/O PORTS

There are five 8-bit I/O ports on the 80C196KC. Some of these ports are input only, some are output only, some are bidirectional and some have alternate functions. In addition to these ports, the HSI/O unit provides extra I/O lines if the timer related features of these lines are not needed.

Port 0 is an input port which is also used as the analog input for the A/D converter. Port 0 is read at location 0EH. Port 1 is a quasi-bidirectional port and is read or written to through location 0FH. The 3 Most Significant bits of Port 1 are multiplexed with the control signals for the HOLD/HLDA bus. Port pins 1.3 and 1.4 are multiplexed with the 2 extra PWM outputs. Port 2 contains three types of port lines: quasi-bidirectional, input and output. Port2 is read or written from location 10H. The ports cannot be read or written in HWindow 15. The input and output lines are shared with other functions in the 80C196KC as shown in Figure 13-1. Ports 3 and 4 are open-drain bidirectional ports which share their pins with the address/data bus. On EPROM and ROM parts, Port 3 and 4 are read and written through location 1FFEH.

While discussing the characteristics of the I/O pins some approximate current or voltage specifications will be given. The exact specifications are available in the latest version of the data sheet that corresponds to the device being used.

PIN	FUNC.	ALTERNATE FUNCTION	CONTROL REG.
2.0	Output	TXD (Serial Port Transmit)	IOC1.5
2.1	Input	RXD (Serial Port Receive)	SPCON.3
2.2	Input	Extint	IOC1.1
2.3	Input	T2CLK (Timer2 Clock & Baud)	IOC0.7
2.4	Input	T2RST (Timer2 Reset)	IOC0.5
2.5	Output	PWM Output	IOC1.0
2.6	QBD*	Timer2 up/down select	IOC2.1
2.7	QBD*	Timer2 Capture	N/A

*QBD = Quasi-bidirectional

Figure 13-1. Port 2 Multiple Functions

13.1 Input Ports

Input ports and pins can only be read. There are no output drivers on these pins. The input leakage of these pins is in the microamp range. The specific values can be found in the data sheet for the device being considered. Figure 13-2 shows the input port structures.

In addition to acting as a digital input, each line of Port 0 can be selected to be the input of the A/D converter as discussed in Section 12. The capacitance on these pins is approximately 1 pF and will instantaneously increase by around 2 pF when the pin is being sampled by the A/D converter.

Port 0 pins are special in that they may individually be used as digital inputs and analog inputs at the same time. A Port 0 pin being used as a digital input acts as the high impedance input ports just described. However, Port 0 pins being used as analog inputs are required to provide current to the internal sample capacitor when a conversion begins. This means the input characteristics of a pin will change if a conversion is being done on that pin. V_{REF} and ANGND must always be connected for Port 0 to function.

Port 0 is only sampled when the SFR is read to reduce the noise in the A/D converter. The data must be stable one state time before the SFR is read.

13.2 Quasi-Bidirectional Ports

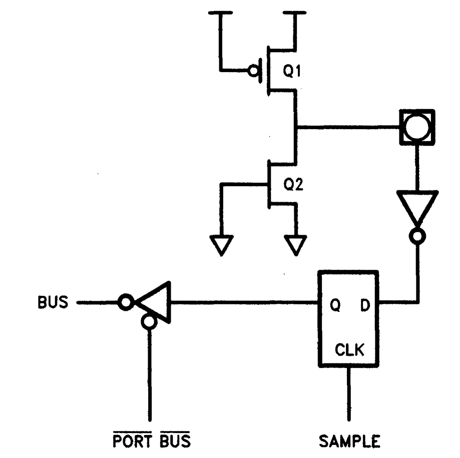
Port 1 and Port 2 have quasi-bidirectional I/O pins. When used as inputs the data on these pins must be stable one state time prior to reading the SFR. This timing is also valid for the input-only pins of Port 2 and is similar to the HSI in that the sample occurs during PH1 or during CLKOUT low. When used as outputs, the quasi-bidirectional pins will change state shortly after CLKOUT falls. If the change was from '0' to a '1' the low impedance pullup will remain on for one state time after the change.

Port 1, Port 2.6 and Port 2.7 are quasi-bidirectional ports. When the processor writes to the pins of a quasi-bidirectional port it actually writes into a register which in turn drives the port pin. When the processor reads these ports, it reads the pin directly. If a port pin is to be used as an input then the software should write a one to its associated SFR bit, which will turn the low-impedance pull-down device off and leave the pin pulled up with a high impedance pullup device. This device can be easily driven down by the device driving the input.

If some pins of a port are to be used as inputs and some are to be used as outputs the programmer should be careful when writing to the port.

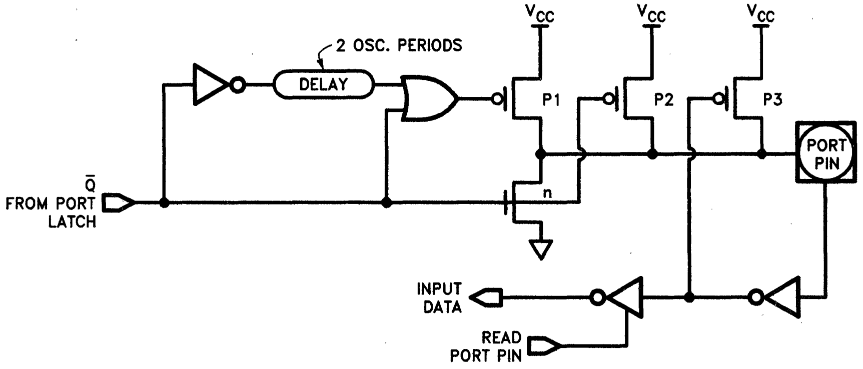
Particular care should be exercised when using read-modify-write instruction. It is possible for a Quasi-Bidirectional Pin to be written as a one, but read back as a zero if an external device (i.e., a transistor base) is pulling the pin below V_{IH}.

Quasi-bidirectional pins can be used as input and output pins without the need for a data direction register. They output a strong low value and a weak high value. The weak high value can be externally pulled low providing an input function. Figure 13-3 shows the configuration of a CHMOS quasi-bidirectional port.



NOTE:
*Q1 and Q2 are ESD Protection Devices

Figure 13-2. Input Port Structure



CHMOS Configuration. pFET 1 is turned on for 2 osc. periods after Q̄ makes a 0-to-1 transition. During this time, pFET 1 also turns on pFET 3 through the inverter to form a latch which holds the 1. pFET 2 is also on.

Figure 13-3. CHMOS Quasi-Bidirectional Port Circuit

Outputting a 0 on a quasi-bidirectional pin turns on the strong pull-down and turns off all of the pull-ups. When a 1 is output the pull-down is turned off and 3 pull-ups (strong-P1, weak-P3, very weak-P2) are turned on. Each time a pin switches from 0 to 1 transistor P1 turns on for two oscillator periods. P2 remains on until a zero is written to the pin. P3 is used as a latch, so it is turned on whenever the pin is above the threshold value (around 2 volts).

To reduce the amount of current which flows when the pin is externally pulled low, P3 is turned off when the pin voltage drops below the threshold. The current required to pull the pin from a high to a low is at its maximum just prior to the pull-up turning off. An external driver can switch these pins easily. The maximum current required occurs at the threshold voltage and is approximately 700 microamps.

When the Port I pins are used as their alternate function, (HOLD, HLDA, BREQ, PWMs), the pins act like a standard output port.

HARDWARE CONNECTION HINTS

When using the quasi-bidirectional ports as inputs tied to switches, series resistors may be needed if the ports will be written to internally after the part is initialized. The amount of current sourced to ground from each pin is typically 7 mA or more. Therefore, if all 8 pins are tied to ground, 56 mA will be sourced. This is equivalent to instantaneously doubling the power used by the chip and may cause noise in some applications.

This potential problem can be solved in hardware or software. In software, never write a zero to a pin being used as an input.

In hardware, a 1K resistor in series with each pin will limit current to a reasonable value without impeding the ability to override the high impedance pullup. If all 8 pins are tied together a 120Ω resistor would be reasonable. The problem is not quite as severe when the inputs are tied to electronic devices instead of switches, as most external pulldowns will not hold 20 mA to 0.0 volts.

Writing to a Quasi-Bidirectional Port with electronic devices attached to the pins requires special attention. Consider using P1.0 as an input and trying to toggle P1.1 as an output:

```
ORB  IOPORT1, #00000001B ; Set P1.0
                         ; for input
XORB IOPORT1, #00000010B ; Complement
                         ; P1.1
```

The first instruction will work as expected but two problems can occur when the second instruction executes. The first is that even though P1.1 is being driven high by the 80C196KC it is possible that it is being held low externally. This typically happens when the port pin drives the base of an NPN transistor which in turn drives whatever there is in the outside world which needs to be toggled. The base of the transistor will clamp the port pin to the transistor's Vbe above ground, typically 0.7V. The 80C196KC will input this value as a zero even if a one has been written to the port pin. When this happens the XORB instruction will always write a one to the port pin's SFR and the pin will not toggle.

The second problem, which is related to the first, is that if P1.0 happens to be driven to a zero when Port 1 is read by the XORB instruction, then the XORB will write a zero to P1.0 and it will no longer be useable as an input.

The first situation can best be solved by the external driver design. A series resistor between the port pin and the base of the transistor often works by bringing up the voltage present on the port pin. The second case can be taken care of in the software fairly easily:

```
LDB  AL, IOPORT1
XORB AL, #010B
ORB  AL, #001B
STB  AL, IOPORT1
```

A software solution to both cases is to keep a byte in RAM as an image of the data to be output to the port; any time the software wants to modify the data on the port it can then modify the image byte and copy it to the port.

If a switch is used on a long line connected to a quasi-bidirectional pin, a pullup resistor is recommended to reduce the possibility of noise glitches and to decrease the rise time of the line. On extremely long lines that are handling slow signals, a capacitor may be helpful in addition to the resistor to reduce noise.

13.3 Output Ports

Output pins include the bus control lines, the HSO lines, and some of Port 2. These pins can only be used as outputs as there are no input buffers connected to them. The output pins are changed before the rising edge of PH1 and is valid some time during PH1. Externally, PH1 corresponds to CLKOUT low. It is not possible to use immediate logical instructions such as XOR to toggle these pins.

The control outputs and HSO pins have output buffers with the same output characteristics as those of the bus pins. Included in the category of control outputs are: TXD, RXD (in Mode 0), PWM, CLKOUT, ALE, BHE, RD, and WR. The bus pins have 3 states: output high, output low, and high impedance. Figure 13-4 shows the internal configuration of an output pin.

13.4 Ports 3 and 4/AD0-15

These pins have two functions. They are either bidirectional ports with open-drain outputs or System Bus pins which the memory controller uses when it is accessing off-chip memory. If the $\overline{EA}$ line is low, the pins always act as the System Bus. Otherwise they act as bus pins only during a memory access.

Accessing Port 3 and 4 as I/O is easily done from internal registers. Since the LD and ST instructions require the use of internal registers, it may be necessary to first move the port information into an internal location before utilizing the data. If the data is already internal, the LD is unnecessary. For instance, to write a word value to Port 3 and 4 . . .

```
LD intreg, portdata   ; register ←
                      ; data
                      ; not needed if
                      ; already
                      ; internal

ST intreg, 1FFEH      ; register →
                      ; Port 3 and 4
```

To read Port 3 and 4 requires that "ones" be written to the port registers to first setup the input port configuration circuit. Note that the ports are reset to this input condition, but if zeroes have been written to the port, then ones must be re-written to any pins which are to be used as inputs. Reading Port 3 and 4 from a previously written zero condition is as follows . . .

```
LD intregA, #0FFFFH   ; setup port
                      ; change mode
                      ; pattern

ST intregA, 1FFEH     ; register →
                      ; Port 3 and 4
                      ; LD & ST not
                      ; needed if
                      ; previously
                      ; written as ones

LD intregB, 1FFEH     ; register ←
                      ; Port 3 and 4
```

When acting as the system bus the pins have strong drivers to both V_{CC} and V_{SS}. These drivers are used whenever data is being output on the system bus and are not used when data is being output by Ports 3 and 4. The pins, external input buffers and pulldowns are shared between the bus and the ports. The ports use different output buffers which are configured as open-drain, and require external pullup resistors. (open-drain is the MOS version of open-collector.) The port pins and their system bus functions are shown in Figure 13-5.

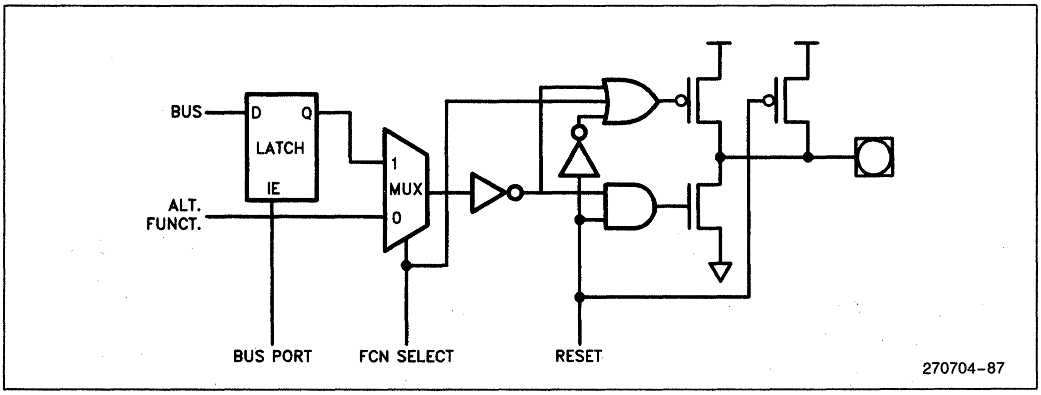


Figure 13-4. Output Port

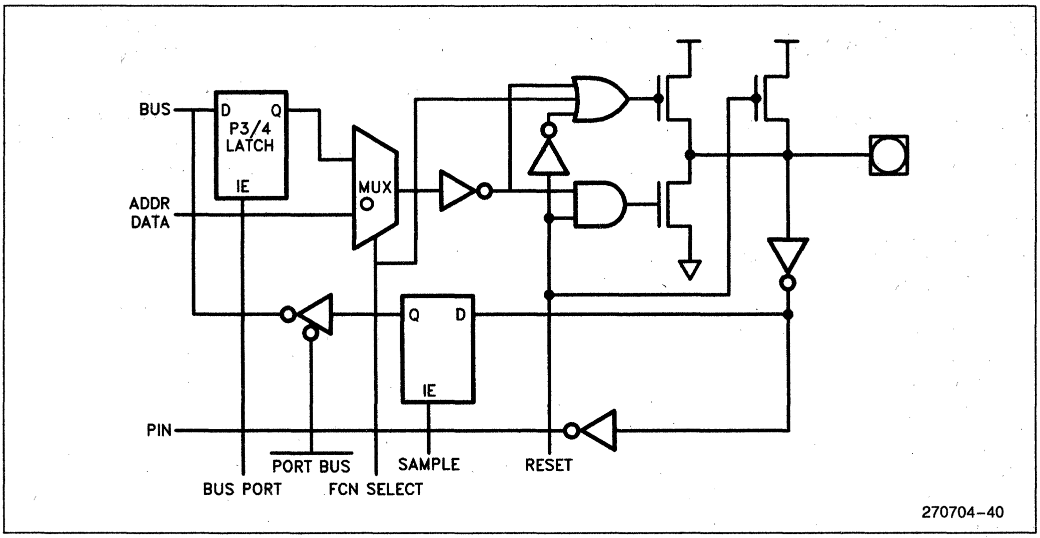


Figure 13-5. Port 3 and 4

Ports 3 and 4 on the 80C196KC are open drain ports. A diagram of the output buffers connected to Ports 3 and 4 and the bus pins is shown in Figure 13-5.

When Ports 3 and 4 are to be used as inputs, they must first be written with a '1'. This will put the ports in a high impedance mode. When they are used as outputs, a pullup resistor must be used externally. A 15K pullup resistor will source a maximum of 0.33 milliamps, so it would be a reasonable value to choose if no other circuits with pullups were connected to the pin.

Ports 3 and 4 are addressed as off-chip memory-mapped I/O. The port pins will change state shortly after the falling edge of CLKOUT. When these pins are used as Ports 3 and 4 they are open drains, their structure is different when they are used as part of the bus.

Port 3 and 4 can be reconstructed as I/O ports from the Address/Data bus.

14.0 MINIMUM HARDWARE CONSIDERATIONS

The 80C196KC requires several external connections to operate correctly. Power and ground must be connected, a clock source must be generated, and a reset circuit must be present. We will look at each of these areas in detail.

14.1 Power Supply

Power to the 80C196KC flows through 6 pins. V_{CC} supplies the positive voltage to the digital portion of the chip while V_{REF} supplies the A/D converter and Port 0 with a positive voltage. These two pins need to be connected to a 5 volt power supply. When using the A/D converter, it is desirable to connect V_{REF} to a separate power supply, or at least a separate trace to minimize the noise in the A/D converter.

The four common return pins, $V_{SS}1$, $V_{SS}2$, $V_{SS}3$, and Angd, must all be nominally at 0 volts. Even if the A/D converter is not being used, V_{REF} and Angd must still be connected for Port0 to function.

14.2 Noise Protection Tips

Due to the fast rise and fall times of high speed CMOS logic, noise glitches on the power supply lines and outputs at the chip are not uncommon. The 80C196KC is no exception. So it is extremely important to follow good design and board layout techniques to keep noise to a minimum. Liberal use of decoupling capacitors, V_{CC} and ground planes, and transient absorbers can all be of great help. It is much easier to design a board with these features then to search for random noise on a poorly designed PC board. For more information on noise, refer to Applications Note AP-125, 'Designing Microcontroller Systems for Noisy Environments' in the *Embedded Control Application Handbook.*

14.3 Oscillator and Internal Timings

ON-CHIP OSCILLATOR

The on-chip oscillator circuitry for the 80C196KC, as shown in Figure 14.1, consists of a crystal-controlled, positive reactance oscillator. In this application, the crystal is operated in its fundamental response mode as an inductive reactance in parallel resonance with capacitance external to the crystal.

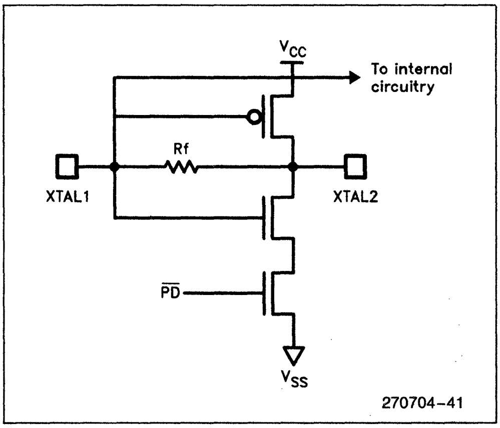


Figure 14-1. On-chip Oscillator Circuitry

The feedback resistor, Rf, consists of paralleled n-channel and p-channel FETs controlled by the PD (powerdown) bit. Rf acts as an open when in Powerdown Mode. Both XTAL1 and XTAL2 also have ESD protection on the pins which is not shown in the figure.

The crystal specifications and capacitance values in Figure 14-2 are not critical. 20 pF is adequate for any frequency above 1 MHz with good quality crystals. Ceramic resonators can be used instead of a crystal in cost sensitive applications. For ceramic resonators, the manufacturer should be contacted for values of the capacitors.

An external oscillator may encounter as much as a 100 pF load at XTAL1 when it starts-up. This is due to interaction between the amplifier and its feedback capacitance. Once the external signal meets the V_{IL} and V_{IH} specifications the capacitance will not exceed 20 pF.

INTERNAL TIMINGS

Internal operation of the chip is based on the oscillator frequency divided by two, giving the basic time unit, known as a 'state time'. With a 16 MHz crystal, a state time is 125 ns. Since the 80C196KC can operate at many frequencies, the times given throughout this overview will be in state times.

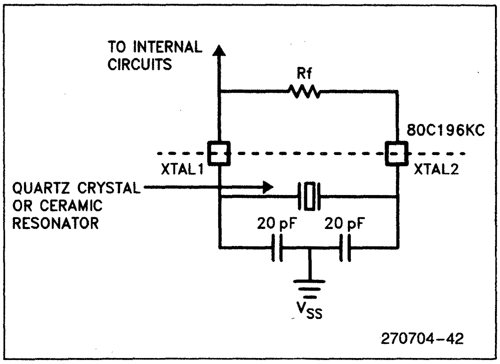


Figure 14-2. External Crystal Connections

To drive the 80C196KC with an external clock source, apply the external clock signal to XTAL1 and let XTAL2 float. An example of this circuit is shown in Figure 14-3. The required voltage levels on XTAL1 are specified in the data sheet. The signal on XTAL1 must be clean with good solid levels.

It is important that the minimum high and low times are met to avoid having the XTAL1 pin in the transition range for long periods of time. The longer the signal is in the transition region, the higher the probability that an external noise glitch could be seen by the clock generator circuitry. Noise glitches on the 80C196KC internal clock lines will cause unreliable operation.

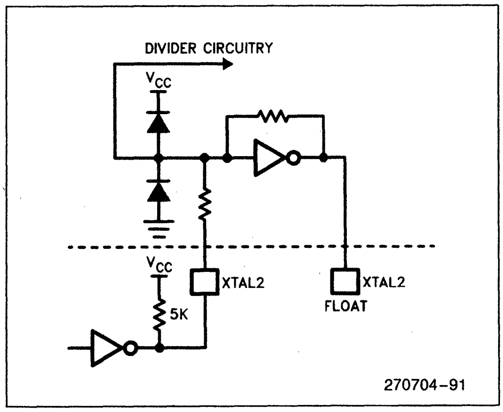


Figure 14-3. External Clock Drive

Two non-overlapping internal phases are created by the clock generator: phase 1 and phase 2 as shown in Figure 14-4. CLKOUT is generated by the rising edge of phase 1 and phase 2. This is not the same as the 8096BH, which uses a three phase clock. Changing from a three phase clock to a two phase speeds up operation for a set oscillator frequency. Consult the latest data sheet for AC timing specifications.

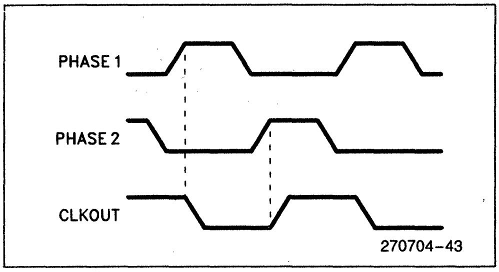


Figure 14-4. Internal Clock Phases

14.4 Reset and Reset Status

Reset starts the 80C196KC off in a known state. To reset the chip, the $\overline{\text{RESET}}$ pin must be held low for at least 16 state times after the power supply is within tolerance and the oscillator has stabilized. As soon as the $\overline{\text{RESET}}$ pin is held low, the I/O and control pins are asynchronously driven to their reset condition.

After the $\overline{\text{RESET}}$ pin is brought high, state reset sequence occurs as shown in Figure 14-5. During this time the CCB (Chip Configuration Byte) is read from location 2018H and stored in the CCR (Chip Configuration Register). The $\overline{\text{EA}}$ (External Access) pin qualifies whether the CCB is read from external or internal memory. Figure 14-6 gives the reset status of all the pins and Special Function Registers.

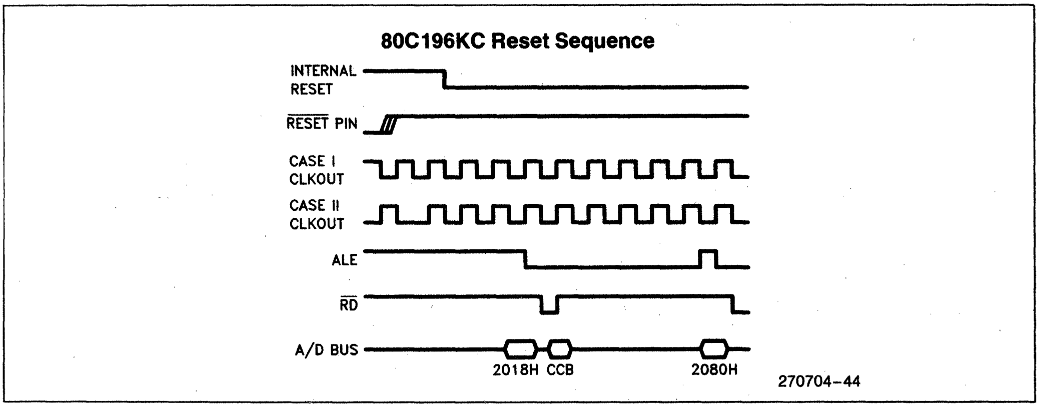


Figure 14-5. Reset Sequence

Pin Name	Multiplexed Port Pins	Value of the Pin on Reset
$\overline{RESET}$		Mid-sized Pullup
ALE		Weak Pullup
$\overline{RD}$		Weak Pullup
$\overline{BHE}$		Weak Pullup
$\overline{WR}$		Weak Pullup
INST		Weak Pulldown
$\overline{EA}$		Undefined Input *
READY		Undefined Input *
NMI		Undefined Input *
BUSWIDTH		Undefined Input *
CLKOUT		CLKOUT
System Bus	P3.0–P4.7	Weak Pullups
ACH0–7	P0.0–P0.7	Undefined Input *
PORT1	P1.0–P1.7	Weak Pullups
TXD	P2.0	Semi-Weak Pullup
RXD	P2.1	Undefined Input *
EXTINT	P2.2	Undefined Input *
T2CLK	P2.3	Undefined Input *
T2RST	P2.4	Undefined Input *
PWM	P2.5	Semi-Weak Pulldown
—	P2.6–P2.7	Weak Pullups
HSI0–HSI1		Undefined Input *
HSI2/HSO4		Undefined Input *
HSI3/HSO5		Undefined Input *
HSO0–HSO3		Weak Pulldown

Register Name	Value
AD_RESULT	7FF0H
AD_TIME	0FFH
HSI_STATUS	x0x0x0x0B
SBUF(RX)	00H
INT_MASK	00000000B
INT_PENDING	00000000B
TIMER1	0000H
TIMER2	0000H
IOPORT1	11111111B
IOPORT2	11000001B
SP_STAT/SP_CON	00001011B
IMASK1	00000000B
IPEND1	00000000B
WSR	XXXX0000B
HSI_MODE	11111111B
IOC2	X0000000B
IOC0	000000X0B
IOC1	00100001B
PWM_CONTROLS	00H
IOPORT3	11111111B
IOPORT4	11111111B
IOS0	00000000B
IOS1	00000000B
IOS2	00000000B
T2CNTC	11111000B

*These pins must be driven and not left floating.

Figure 14-6. Chip Reset Status

WATCHDOG TIMER

There are three ways in which the 80C196KC can reset itself. The watchdog timer will reset the 80C196KC if it is not cleared in 64K state times. The watchdog timer is enabled the first time it is cleared. To clear the watchdog, write a '1E' followed immediately by an 'E1' to location 0AH. Once enabled, the watchdog can only be disabled by a reset and on the 80C196KC.

RST INSTRUCTION

Executing a RST instruction will also reset the 80C196KC. The opcode for the RST instruction is 0FFH. By putting pullups on the Addr/data bus, unimplemented areas of memory will read 0FFH and cause the 80C196KC to be reset.

RESET CIRCUITS

The simplest way to reset an 80C196KC is to insert a capacitor between the $\overline{\text{RESET}}$ pin and V_{SS}. The 80C196KC has an internal pullup. A 5 uF or greater capacitor should provide sufficient reset time as long as V_{CC} rises quickly.

Figure 14-7 shows what the $\overline{\text{RESET}}$ pin looks like internally. The $\overline{\text{RESET}}$ pin functions as an input and as an output to reset an entire system with a watchdog timer overflow, or by executing a RST instruction. For a system reset application, the reset circuit should be a one-shot with an open collector output. The reset pulse may have to be lengthened and buffered since $\overline{\text{RESET}}$ is only asserted for 16 state times. A capacitor cannot be connected directly to $\overline{\text{RESET}}$ if it is to drive the reset pins of other chips in the circuit. The capacitor may keep the voltage on the pin from going below guaranteed V_{IL} for circuits connected to the $\overline{\text{RESET}}$ pin. Figure 14-8 shows an example of a system reset circuit.

14.5 Minimum Hardware Connections

Figure 14-9 shows the minimum connections needed to get the 80C196KC up and running. It is important to tie all unused inputs to V_{CC} or V_{SS}. If these pins are left floating, they can float to a mid voltage level and draw excessive current. Some pins such as NMI or EXTINT may generate spurious interrupts if left unconnected.

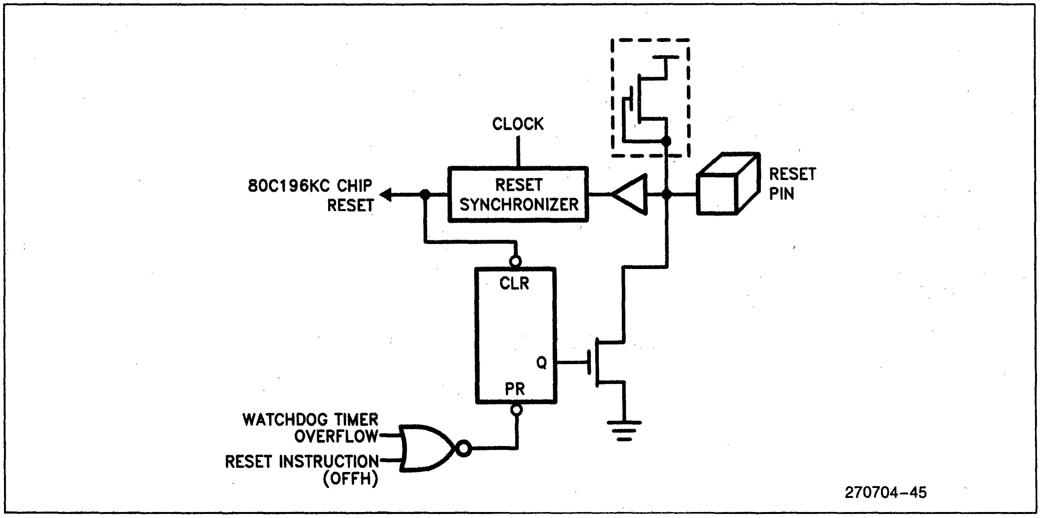
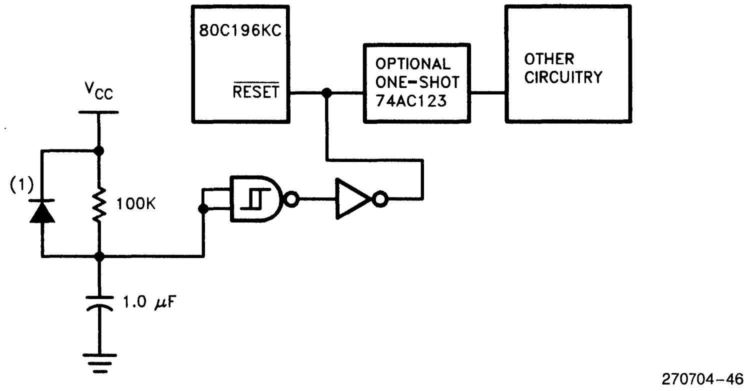


Figure 14-7. Reset Pin



NOTE:
1. The diode will provide a faster cycle time repetitive power-on-resets.

Figure 14-8. System Reset Circuit

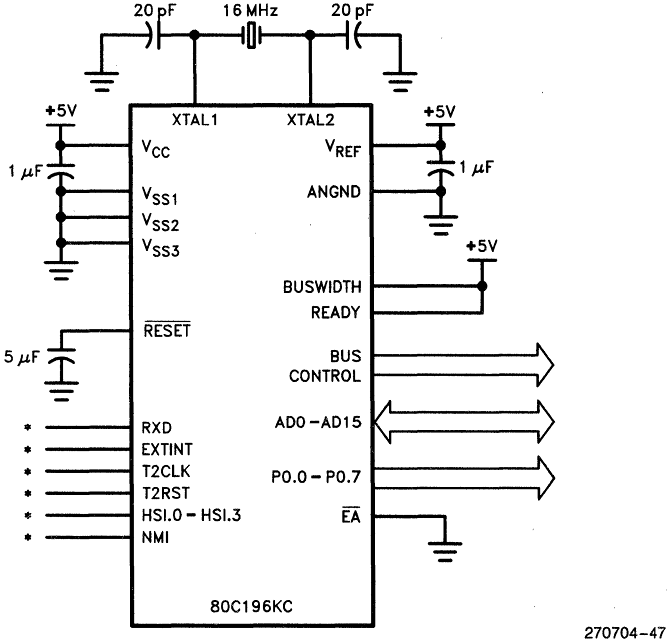


Figure 14-9. 80C196KC Minimum Hardware Connections

15.0 SPECIAL MODES OF OPERATION

The 80C196KC has Idle and Powerdown Modes to reduce the amount of current consumed by the chip. The 80C196KC also has an ONCE (ON-Circuit-Emulation) Mode to isolate itself from the rest of the components in the system.

15.1 Idle Mode

The Idle Mode is entered by executing the instruction 'IDLPD #1'. In the Idle Mode, the CPU stops executing. The CPU clocks are frozen at logic state zero, but the peripheral clocks and CLKOUT continue to be active. Power consumption in the Idle Mode is reduced to about 40% of the active Mode.

The CPU exits the Idle Mode by any enabled interrupt source or a hardware reset. Since all of the peripherals are running, the interrupt can be generated by the HSI, HSO, A/D, serial port, etc. When an interrupt brings the CPU out of the Idle Mode, the CPU vectors to the corresponding interrupt service routine and begins executing. The CPU returns from the interrupt service routine to the next instruction following the 'IDLPD #1' instruction that put the CPU in the Idle Mode.

A PTS cycle also causes the CPU to exit the IDLE mode. The CPU begins executing the instruction following the "IDLE #1" instruction that put the device into IDLE mode.

In the Idle Mode, the system bus control pins (ALE, RD, WR, INST, and BHE), go to their inactive states. Ports 3 and 4 will retain the value present in their data latches if being used as I/O ports. If these ports are the ADDR/DATA bus, the pins will float.

It is important to note the Watchdog Timer continues to run in the Idle Mode if it is enabled. So the chip must be awakened every 64K state times to clear the Watchdog or the chip will reset.

15.2 Powerdown Mode

The Powerdown Mode is entered by executing the instruction, 'IDLPD #2'. In the Powerdown Mode, all internal clocks are frozen at logic state zero and the oscillator is shut off. All 232 bytes of registers plus the 256 bytes of extra RAM and most peripherals hold their values if V_{CC} is maintained. Power is reduced to the device leakage and is in the uA range.

In Powerdown, the bus control pins go to their inactive states. All of the output pins will assume the value in their data latches. Ports 3 and 4 will continue to act as ports in the single chip mode or will float if acting as the ADDR/DATA bus.

To prevent accidental entry into the Powerdown Mode, this feature may be disabled at reset by clearing bit 0 of the CCR (Chip Configuration Register). Since the default value of the CCR bit 0 is 1, the Powerdown Mode is normally enabled.

The Powerdown Mode can be exited by a chip reset or a high level on the external interrupt pin. If the RESET pin is used, it must be asserted long enough for the oscillator to stabilize.

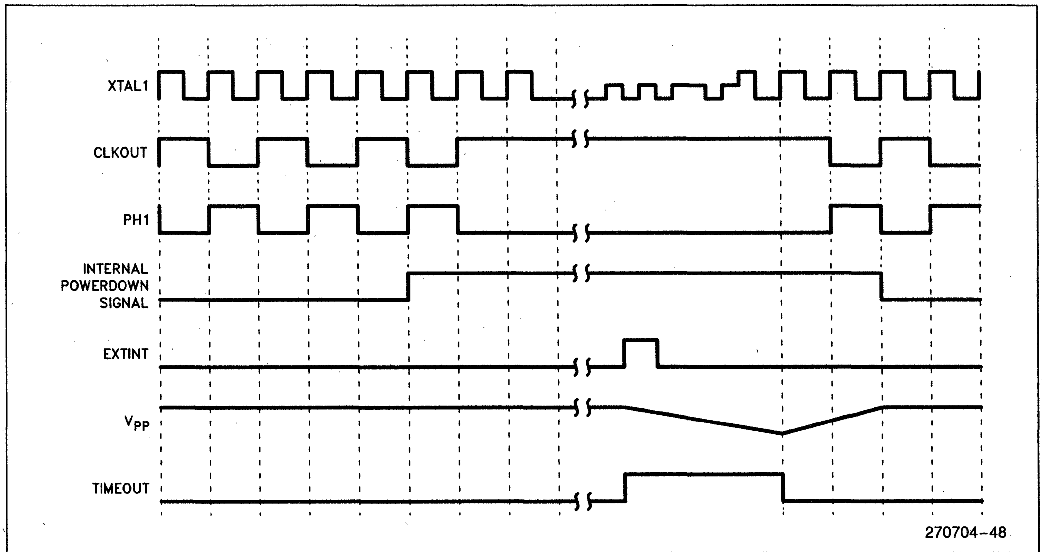


Figure 15-1. Power Up and Power Down Sequence

When exiting Powerdown with an external interrupt, a positive level on the pin mapped to INT7 (either EXTINT or port0.7) will bring the chip out of Powerdown Mode. The interrupt does not have to be unmasked to exit Powerdown. An internal timing circuit ensures that the oscillator has time to stabilize before turning on the internal clocks. Figure 15-1 shows the power down and power up sequence using an external interrupt.

During normal operation, before entering Powerdown Mode, the V_{PP} pin will rise to V_{CC} through an internal pullup. The user must connect a capacitor between V_{PP} and V_{SS}. A positive level on the external interrupt pin starts to discharge this capacitor. The internal current source that discharges the capacitor can sink approximately 100 uA. When the voltage goes below about 1 volt on the V_{PP} pin, the chip begins executing code. A 1uF capacitor would take about 4 ms to discharge to 1 volt.

If the external interrupt brings the chip out of Powerdown, the corresponding bit will be set in the interrupt pending register. If the interrupt is unmasked, the device will immediately execute the interrupt service routine, and return to the instruction following the IDLPD instruction that put the chip into Powerdown. If the interrupt is masked, the chip will start at the instruction following the IDLPD instruction. The bit in the pending register will remain set, however.

All peripherals should be in an inactive state before entering Powerdown. If the A/D converter is in the middle of a conversion, it is aborted. If the chip comes out of Powerdown by an external interrupt, the serial port will continue where it left off. Make sure that the serial port is done transmitting or receiving before entering Powerdown. The SFRs associated with the A/D and the serial port may also contain incorrect information when returning from Powerdown.

When the chip is in Powerdown, it is impossible for the watchdog timer to time out because its clock has stopped. Systems which must use the Watchdog and Powerdown, should clear the Watchdog right before entering Powerdown. This will keep the Watchdog from timing out when the oscillator is stabilizing after leaving Powerdown.

15.3 ONCE™ and Test Modes

Test Modes can be entered on the 80C196KC by holding ALE,$\overline{\text{WR}}$, $\overline{\text{HLDA}}$ or $\overline{\text{RD}}$ in their active state on the rising edge of $\overline{\text{RESET}}$. For this reason the I_{OL} and I_{OH} in Reset Specifications must be carefully verified. The only Test Mode not reserved for use by Intel is the ONCE, or ON-Circuit-Emulation Mode.

ONCE is entered by driving TXD pin low on the rising edge of $\overline{\text{RESET}}$. The TXD pin will source about 1 mA at a logical 1 during reset. External circuitry must not pull the pin low or the ONCE mode will be entered. All pins except XTAL1 and XTAL2 are floated. Some of the pins are not truly high impedance as they have weak pullups or pulldowns. The ONCE Mode is useful in electrically removing the 80C196KC from the rest of the system. A typical application of the ONCE Mode would be to program discrete EPROMs onboard without removing the 80C196KC from its socket.

16.0 EXTERNAL MEMORY INTERFACING

16.1 Bus Operation

There are several different external operating modes on the 80C196KC. The standard bus mode uses a 16 bit multiplexed address/data bus. Other bus modes include an 8 bit external bus mode and a mode in which the bus size can be dynamically switched between 8-bits and 16-bits. In addition, there are several options available on the type of bus control signals which make an external bus simple to design.

In the standard mode, external memory is addressed through lines AD0-AD15 which form a 16 bit multiplexed bus. The address/data bus shares pins with ports 3 and 4. Figure 16-1 shows an idealized timing diagram for the external bus signals.

Address Latch Enable (ALE) provides a strobe to transparent latches (74AC373s) to demultiplex the bus. To avoid confusion, the latched address signals will be called MA0-MA15 and the data signals will be named MD0-MD15.

The data returned from external memory must be on the bus and stable for a specified setup time before the rising edge of $\overline{\text{RD}}$ (read). The rising edge of $\overline{\text{RD}}$ signals the end of the sampling window. Writing to external memory is controlled with the $\overline{\text{WR}}$ (write) pin. Data is valid on MD0–MD15 on the rising edge of $\overline{\text{WR}}$. At this time data must be latched by the external system. The 80C196KB has ample setup and hold times for writes.

When $\overline{\text{BHE}}$ is asserted, the memory connected to the high byte of the data bus is selected. When MA0 is a 0, the memory connected to the low byte of the data bus is selected. In this way accesses to a 16-bit wide memory can be to the low (even) byte only (MA0 = 0, $\overline{\text{BHE}}$ = 1), to the high (odd) byte only (MA0 = 1, $\overline{\text{BHE}}$ = 0), or the both bytes (MA0 = 0, $\overline{\text{BHE}}$ = A0).

When a block of memory is decoded for reads only, the system does not have to decode $\overline{\text{BHE}}$ and MA0. The 80C196KB will discard the byte it does not need. For systems that write to external memory, a system must generate separate write strobes to both the high and low byte of memory. This is discussed in more detail later.

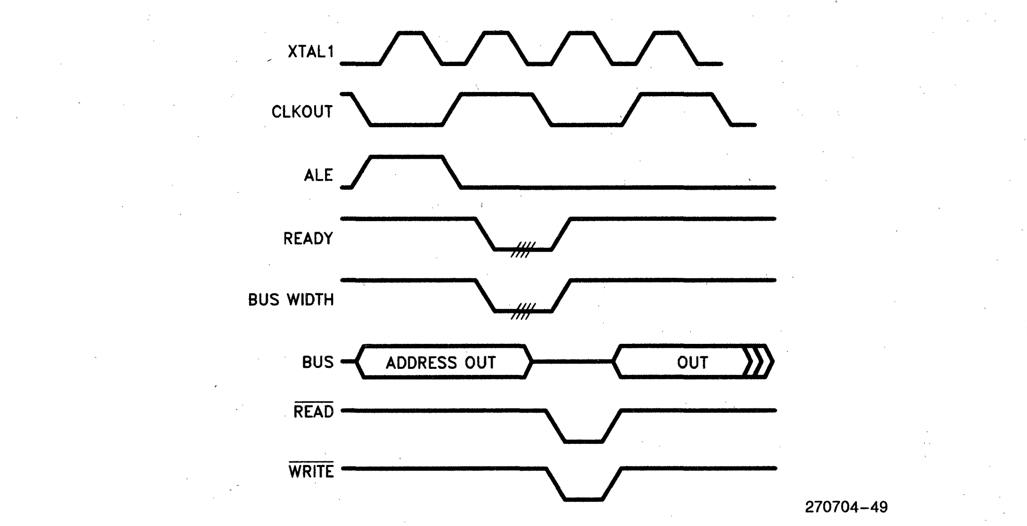


Figure 16-1. Idealized Bus Timings

All of the external bus signals are gated by the rising and falling edges of CLKOUT. A zero waitstate bus cycle consists of two CLKOUT periods. Therefore, there are 4 clock edges that generate a complete bus cycle. The first falling edge of CLKOUT asserts ALE and drives an address on the bus. The rising edge of CLKOUT drives ALE inactive. The next falling edge of CLKOUT asserts $\overline{RD}$ (read) and floats the bus for a read cycle. During a $\overline{WR}$ (write) cycle, this edge asserts $\overline{WR}$ and drives valid data on the bus. On the last rising edge of CLKOUT, data is latched into the 80C196KB for a read cycle, or data is valid for a write cycle.

16.2 Chip Configuration Register

The CCR (Chip Configuration Register) is the first byte fetched from memory following a chip reset. The CCR is fetched from the CCB (Chip Configuration Byte) at location 2018H in either internal or external memory depending on the $\overline{EA}$ pin. The CCR is only loaded once during the reset sequence. Once loaded, the CCR cannot be changed until the next reset.

The CCR is shown in Figure 16-2.

Bits 7 and 6 of the CCR control ROM/EPROM protection. ROM/EPROM protection is covered in Section 17. The next two bits control the internal READY mode. The three next bits determine the bus control signals. The next bit enables or disables the Powerdown Mode.

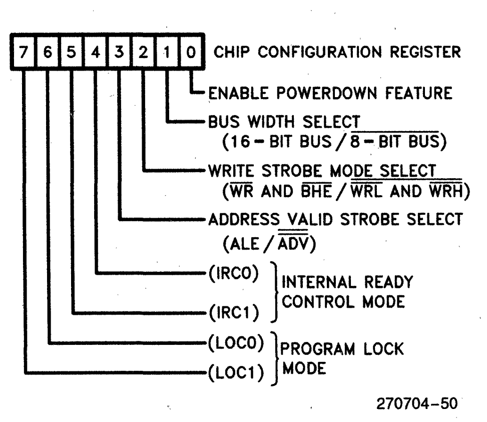


Figure 16-2. Chip Configuration Register

READY control

The user has two options for ready control. He can use the READY pin and/or the internal ready control bits. Ready control is only valid for external memory. On-chip RAM/SFR space and on-chip ROM/EPROM is always accessed with 0 waitstates. The modes are chosen by bits 4 and 5 of the CCR and are shown in Figure 16-3.

IRC1	IRC0	Description
0	0	Limit to one waitstate
0	1	Limit to two waitstates
1	0	Limit to three waitstates
1	1	Wait states not limited internally

Figure 16-3. Ready Control Modes

The internal ready control logic limits the number of waitstates that slow devices can insert into the bus cycle. When the READY pin is pulled low, waitstates are inserted into the bus cycle until the READY pin goes high, or the number of waitstate equal the number programmed into the CCR. So the ready control is a logical OR between the READY pin and the internal ready control.

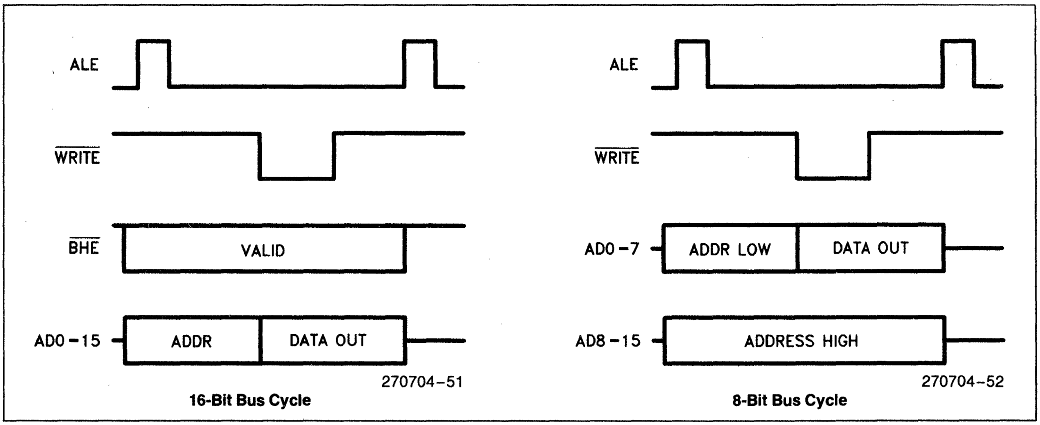
This feature gives very simple and flexible ready control. For example, every slow memory chip select line could be ORed together and connected to the READY pin with Internal Ready Control programmed to insert the desired number of waitstates into the bus cycle.

Bus control

Using the CCR, the 80C196KC can generate several types of control signals designed to reduce external hardware. The ALE, WR, and BHE pins serve dual functions. Bits 2 and 3 of the CCR specify the function performed by these control lines.

Standard bus control

If CCR bits 2 and 3 are 1s, the standard bus control signals ALE, WR, and BHE are generated as shown in Figure 16-4. ALE rises as the address starts to be driven, and falls to externally latch the address. WR is driven for every write. BHE and MA0 can be combined to form WRL and WRH for even and odd byte writes.



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 16-4. Standard Bus Control

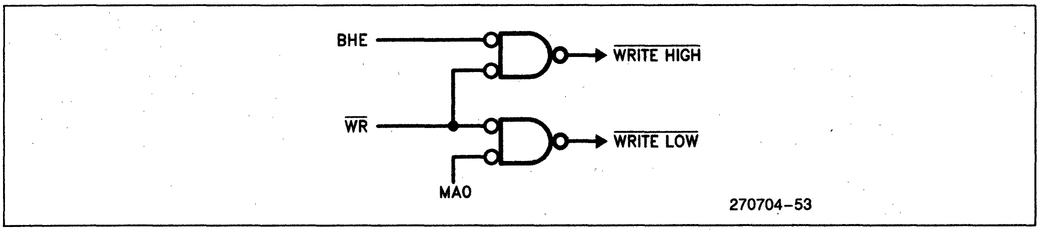


Figure 16-5. Decoding $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$

Figure 16-5 is an example of external circuitry to decode $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$.

Write Strobe Mode

The Write Strobe Mode eliminates the need to externally decode odd and even byte writes. If the CCR bit 2 is 0, and the bus is a 16-bit cycle, $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$ are generated in place of $\overline{\text{WR}}$ and $\overline{\text{BHE}}$. $\overline{\text{WRL}}$ is asserted for all byte writes to an even address and all word writes. $\overline{\text{WRH}}$ is asserted for all byte writes to odd addresses and all word writes. The Write Strobe mode is shown in Figure 16-6.

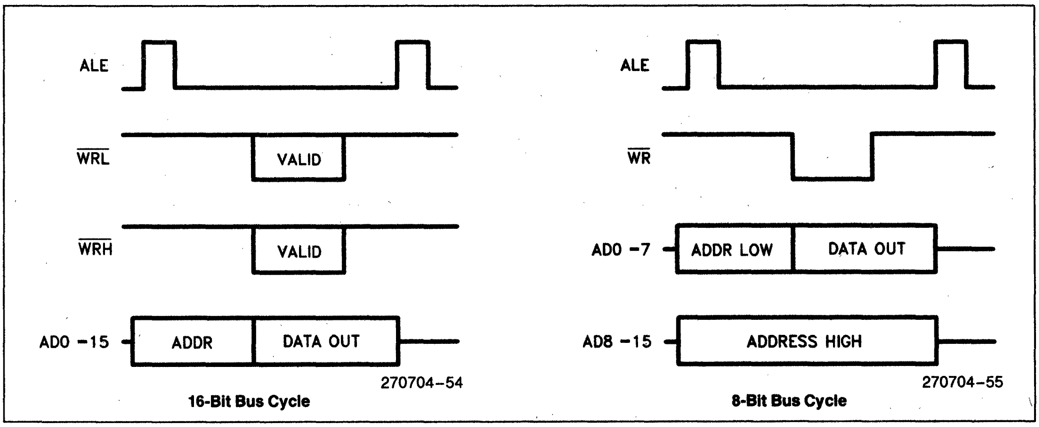
In the eight bit mode, $\overline{\text{WRL}}$ and $\overline{\text{WRH}}$ are asserted for both even and odd addresses.

Address Valid Strobe Mode

Address Valid strobe replaces ALE if the CCR bit 3 is 0. When Address valid Strobe mode is selected, $\overline{\text{ADV}}$ will be asserted after an external address is setup. It will stay asserted until the end of the bus cycle as shown in Figure 16-7. $\overline{\text{ADV}}$ can be used as a simple chip select for external memory. $\overline{\text{ADV}}$ looks exactly like ALE for back to back bus cycles. The only difference is $\overline{\text{ADV}}$ will be inactive when the external bus is idle.

Address Valid with Write Strobe

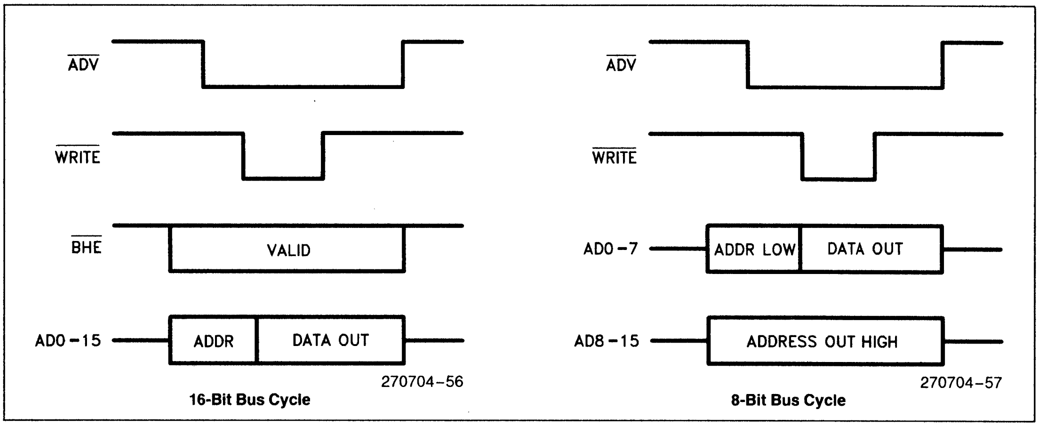
If the CCR bits 2 and 3 are 0, the Address Valid with Write Strobe mode is enabled. Figure 16-8 shows the signals.



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 16-6. Write Strobe Mode



16-Bit Bus Cycle

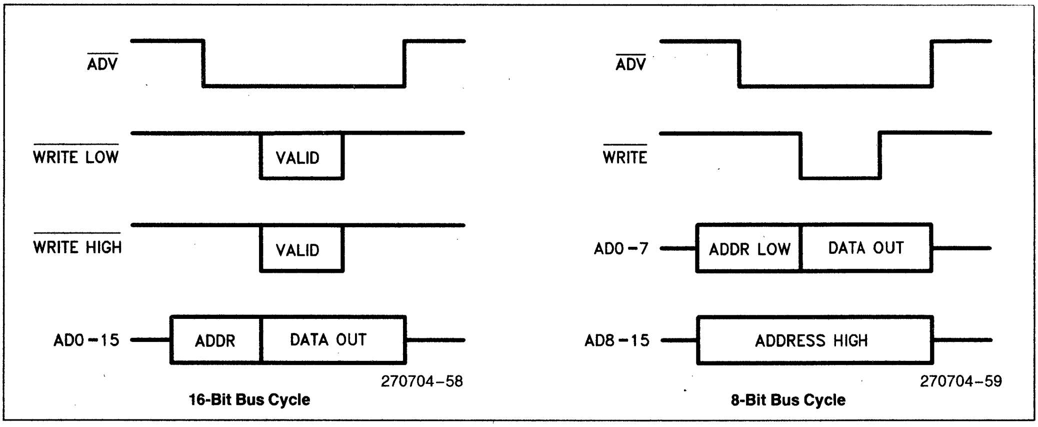
8-Bit Bus Cycle

Figure 16-7. Address Valid Strobe Mode

16.3 Bus Width

The 80C196KC external bus width can be run-time configured to operate as a 16-bit multiplexed address/data bus, or a multiplexed 16-bit address/8-bit data bus.

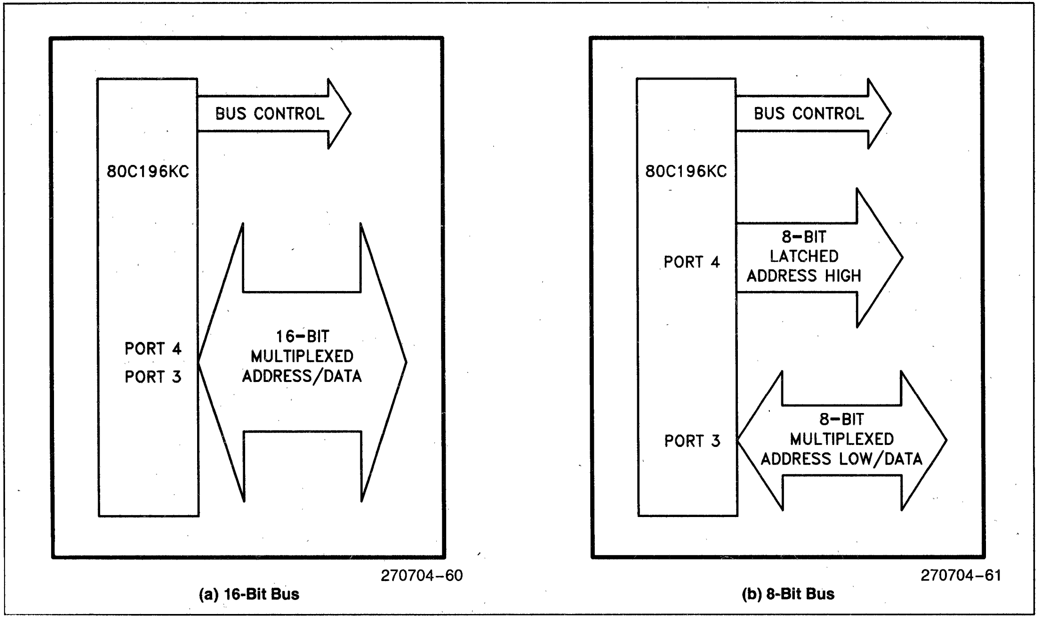
During 16-bit bus cycles, Ports 3 and 4 contain the address multiplexed with data using ALE to latch the address. In 8-bit bus cycles, Port 3 is multiplexed with address/data but Port 4 only outputs the upper 8 address bits. The Addresses on Port 4 are valid throughout the entire bus cycle. Figure 16-9 shows the two bus width options.



16-Bit Bus Cycle

8-Bit Bus Cycle

Figure 16-8. Address Valid with Write Strobe Mode



(a) 16-Bit Bus

(b) 8-Bit Bus

Figure 16-9. Bus Width Options

The external bus width can be changed every bus cycle if a 1 was loaded into bit CCR1 at reset. The bus width is changed on the fly by using the BUSWIDTH pin. If the BUSWIDTH pin is a 1, the bus cycle is 16-bits. For an 8-bit bus cycle, the BUSWIDTH pin is a 0. The BUSWIDTH pin is sampled by the 80C196KC after the address is on the bus. The BUSWIDTH pin has about the same timing as the READY pin.

Applications for the BUSWIDTH pin are numerous. For example, a system could have code fetched from 16 bit memory, while data would come from 8 bit memory. This saves the cost of using two 8 bit static RAMS if the capacity of only one is needed. This system could be easily implemented by tying the chip select input of the 8-bit memory to the BUSWIDTH pin.

If the CCR bit 1 is a 0, the 80C196KC is locked into the 8 bit mode and the BUSWIDTH pin is ignored.

When executing code from a 8-bit bus, some performance degradation is to be expected. The prefetch queue cannot be kept full under all conditions from an 8-bit bus. Also, word reads and writes to external memory take an extra bus cycle.

INST PIN

The INST pin is useful for decoding more than 64K of addressing space. The INST pin allows both 64K of code space and 64K of data space. For instruction fetches from external memory, the INST pin is asserted, or high for the entire bus cycle. For data reads and writes, the INST pin is low. The INST pin is low for the Chip Configuration Byte fetch and for interrupt vector fetches.

16.4 $\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ Protocol

The 80C196KC supports a bus exchange protocol, allowing other devices to gain control of the bus. The protocol consists of three signals, $\overline{\text{HOLD}}$, $\overline{\text{HLDA}}$, and $\overline{\text{BREQ}}$. $\overline{\text{HOLD}}$ is an input asserted by a device which requests the 80C196KC bus. Figure 16-10 shows the timing for $\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$. The 80C196KC responds by releasing the bus and asserting $\overline{\text{HLDA}}$. When the device is done accessing the 80C196KC bus, it relinquishes the bus by deasserting the $\overline{\text{HOLD}}$ pin. The 80C196KC will remove its $\overline{\text{HDLA}}$ and assume control of the bus. The third signal, $\overline{\text{BREQ}}$, is asserted by the 80C196KC during the hold sequence when it has a pending external bus cycle. The 80C196KC deasserts $\overline{\text{BREQ}}$ at the same time it deasserts $\overline{\text{HDLA}}$.

The $\overline{\text{HOLD}}$, $\overline{\text{HLDA}}$, and $\overline{\text{BREQ}}$ pins are multiplexed with P1.7, P1.6, and P1.5, respectively. To enable $\overline{\text{HOLD}}$, $\overline{\text{HLDA}}$ and $\overline{\text{BREQ}}$, the HLDEN bit (WSR.7) must be set. HLDEN is cleared during reset. Once this bit is set, the port1 pins cannot be returned to being quasi-bidirectional pins until the device is $\overline{\text{RESET}}$, but can still be read as inputs. The $\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ feature, however, can be disabled by clearing the HLDEN bit for locked bus cycles.

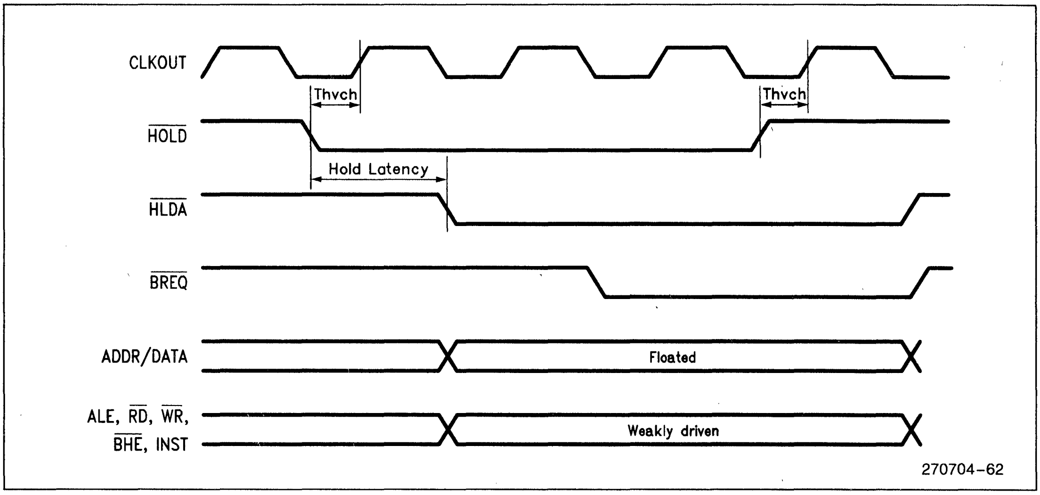


Figure 16-10. $\overline{\text{HOLD}}/\overline{\text{HLDA}}$ Timings

$\overline{\text{HOLD}}$ is sampled on phase 1, or when CLKOUT is low.

When the 80C196KC acknowledges the hold request, the output buffers for the addr/data bus are floated. ALE and INST are weakly held LOW and $\overline{\text{ADV}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$ are weakly held high during HOLD. The request to hold latency is dependent on the state of the bus controller.

MAXIMUM HOLD LATENCY

The time between $\overline{\text{HOLD}}$ being asserted and $\overline{\text{HLDA}}$ being driven is known as Hold Latency. After recognizing $\overline{\text{HOLD}}$, the 80C196KC waits for any current bus cycle to finish, and then asserts $\overline{\text{HLDA}}$. There are 3 types bus cycles; 8-bit external cycle, 16-bit external cycle, and an idle bus. Accessing on-chip ROM/EPROM is an idle bus.

$\overline{\text{HOLD}}$ is an asynchronous input. There are two different system configurations for asserting $\overline{\text{HOLD}}$. The 80C196KC will recognize $\overline{\text{HOLD}}$ internally on the next clock edge if the system meets T_{HVCH} ($\overline{\text{HOLD}}$ valid to CLKOUT high). If T_{HVCH} is not met ($\overline{\text{HOLD}}$ applied asynchronously), $\overline{\text{HOLD}}$ may be recognized one clock later (see Figure 16-12). Consult the latest 80C196KC data sheet for the T_{HVCH} specification.

Figure 16-12 shows the 80C196KC entering $\overline{\text{HOLD}}$ when the bus is idle. This is the minimum hold latency for both the synchronous and asynchronous cases. If T_{HVCH} is met, $\overline{\text{HLDA}}$ is asserted about on the next falling edge of CLKOUT. See the data sheet for T_{CLHAL} (CLKOUT low to $\overline{\text{HLDA}}$ low) specification. For this case, the minimum hold latency = T_{HVCL} + 0.5 states + T_{CLHAL}.

If $\overline{\text{HOLD}}$ is asserted asynchronously, the minimum hold latency increases by one state time and = T_{HVCL} + 1.5 states + T_{CLHAL}.

Figure 16-11 summarizes the additional hold latency added to the minimum latency for the 3 types of bus cycles. When accessing external memory, add one state for each waitstate inserted into the bus cycle. For an 8-bit bus, worst case hold latency is for word reads or writes. For this case, the bus controller must access the bus twice, which increases latency by two states.

For exiting Hold, the minimum hold latency times apply for when the 80C196KC will deassert $\overline{\text{HLDA}}$ in response to $\overline{\text{HOLD}}$ being removed.

	Max Hold Latency
Idle Bus	Min
16-bit External Access	Min + 1 State
8-bit External Access	Min + 3 States

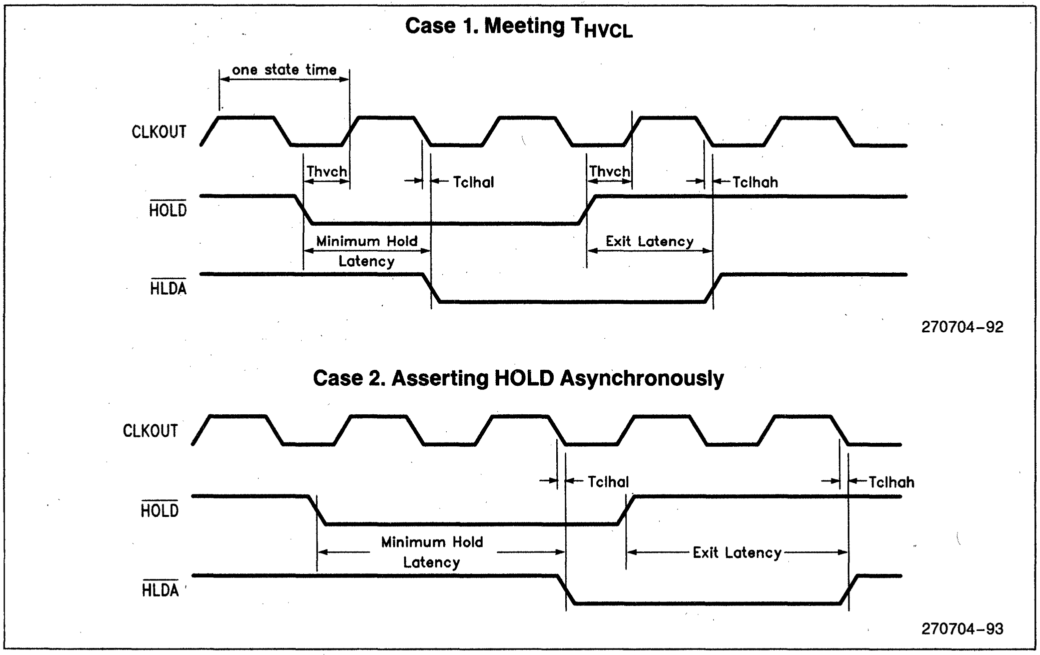
Min = T_{HVCL} + 0.5 states + T_{CLHAL} if T_{HVCL} is met
= T_{HVCL} + 1.5 states + T_{CLHAL} for asynchronous HOLD

Figure 16-11. Maximum HOLD Latency

REGAINING BUS CONTROL

There is no delay from the time the 80C196KC removes $\overline{\text{HLDA}}$ to the time it takes control of the bus. After $\overline{\text{HOLD}}$ is removed, the 80C196KC drops $\overline{\text{HLDA}}$ in the following state and resumes control of the bus.

$\overline{\text{BREQ}}$ is asserted when the part is in hold and needs to perform an external memory cycle. An external memory cycle can be a data access or a request from the prefetch queue for a code request. A request comes from the queue when it contains two bytes or less. Once asserted, it remains asserted until $\overline{\text{HOLD}}$ is removed. At the earliest, $\overline{\text{BREQ}}$ can be asserted with $\overline{\text{HLDA}}$.



Case 1. Meeting T_{HVCL}

270704–92

Case 2. Asserting HOLD Asynchronously

270704–93

Figure 16-12

Hold requests do not freeze the 80C196KC when executing out of internal memory. The device continues executing as long as the resources it needs are located internal to the 80C196KC. As soon as the device needs to access external memory, it asserts BREQ and waits for the HOLD to be removed. At this time, the device cannot respond to any interrupt requests until HOLD is removed.

When executing out of external memory during a HOLD, the 80C196KC keeps running until the queue is empty or it needs to perform an external data cycle. The 80C196KC cannot service any interrupts until HOLD is removed.

The 80C196KC will also respond to hold requests in the Idle Mode. The latency for entering bus hold from the Idle Mode is the same as when executing out of internal memory.

Special consideration must be given to the bus arbiter design if the 80C196KC can be reset while in HOLD. For example, a CPU device would try and fetch the CCR from external memory after RESET is brought high. Now there would be two parts attempting to access 80C196KC memory. The simplest solution is to make the RESET pin of the 80C196KC a system reset. This way the other bus master would also be reset. Examples of system reset circuits are given in Section 13.

DISABLING HOLD REQUESTS

Clearing the HLDEN bit (WSR.7), can disable HOLD requests when consecutive memory cycles are required. Clearing the HDLEN bit, however, does not cause the 80C196KC to take over the bus immediately. The 80C196KC waits for the current HOLD request to finish. Then it disables the bus hold feature, causing any new requests to be ignored until the HLDEN bit is set again. Since there is a delay from the time the code for clearing this bit is fetched to the time it is actually executed, the code that clears HLDEN needs to be a few instructions ahead of the block that needs to be protected from HOLD requests.

The safest way is to add a JBC instruction to check the status of the HLDA pin after the code that clears the HLDEN bit. Figure 16-13 is an example of code that prevents the part from executing a new instruction until both current HOLD requests are serviced and the hold feature is disabled.

16.5 AC Timing Explanations

Figure 16-14 shows the timing of the ADDR/DATA bus and control signals. Refer to the latest data sheet for the AC timings to make sure your system meets specifications. The major timing specifications are explained in English in Figure 16-15.

```
        DI                  ; disable interrupts
        ANDB WSR, #0EFH     ; disable hold request
WAIT:   JBC PORT1, 6, WAIT; Check the HLDA pin
            •               ; If set, execute
            •               ; protected instructions
            •
        ORB WSR,#80h        ; enable HOLD requests
        EI                  ; enable interrupts
```

NOTE:
Interrupts should be disabled to prevent code interruption

Figure 16-13. HOLD code

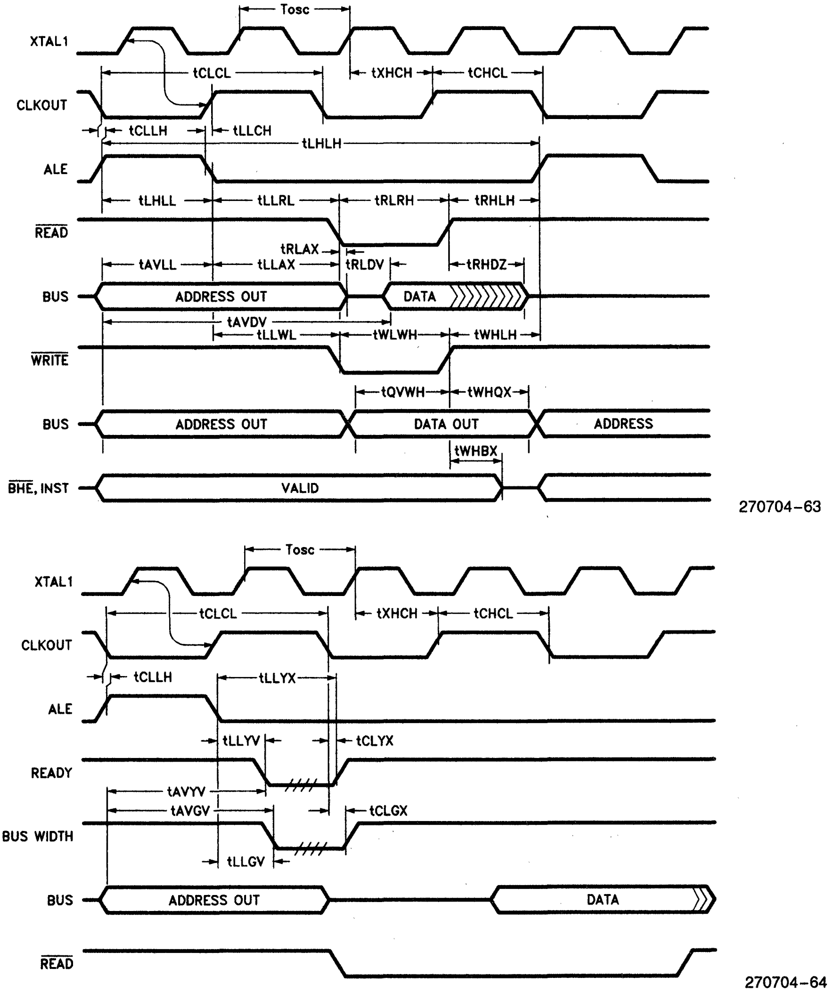


Figure 16-14. AC Timing Diagrams

TIMINGS THE MEMORY SYSTEM MUST MEET:

T_{AVYV} — **ADDRESS Valid to READY Setup:** Maximum time the memory system has to decode READY after ADDRESS is output by the 80C196KC to guarantee at least one-wait state will occur.

T_{LLYV} — **ALE Low to READY Setup:** Maximum time the memory system has to decode READY after ALE falls to guarantee at least one wait state will occur.

T_{YLYH} — **READY Low to READY HIGH:** Maximum amount of nonREADY time or the maximum number of wait states that can be inserted into a bus cycle. Since the 80C196KC is a completely static part, T_{YLYH} is unbounded.

T_{CLYX} — **READY Hold after CLKOUT Low:** Minimum time the level on the READY pin must be valid after CLKOUT falls. The minimum hold time is always 0 ns. If maximum value is exceeded, additional wait states will occur.

T_{LLYX} — **READY Hold AFTER ALE Low:** Minimum time the level on the READY pin must be valid after ALE falls. If maximum value is exceeded, additional wait states will occur.

T_{AVGV} — **ADDRESS Valid to BUSWIDTH Valid:** Maximum time the memory system has to decode BUSWIDTH after ADDRESS is output by the 80C196KC. If exceeded, it is not guaranteed the 80C196KC will respond with an 8- or 16-bit bus cycle.

T_{LLGV} — **ALE Low to BUSWIDTH Valid:** Maximum time after ALE/$\overline{ADV}$ falls until BUSWIDTH must be valid. If exceeded, it is not guaranteed the 80C196KC will respond with an 8- or 16-bit bus cycle.

T_{CLGX} — **BUSWIDTH Hold after CLKOUT Low:** Minimum time BUSWIDTH must be held valid after CLKOUT falls. Always 0 ns of the 80C196KC.

T_{AVDV} — **ADDRESS Valid to Input Data Valid:** Maximum time the memory system has to output valid data after the 80C196KC outputs a valid address.

T_{RLDV} — **$\overline{RD}$ Low to Input Data Valid:** Maximum time the memory system has to output valid data after the 80C196KC asserts $\overline{RD}$.

T_{CLDV} — **CLKOUT Low to Input Data Valid:** Maximum time the memory system has to output valid data after the CLKOUT falls.

T_{RHDZ} — **$\overline{RD}$ High to Input Data Float:** Time after $\overline{RD}$ is inactive until the memory system must float the bus. If this timing is not met, bus contention will occur.

T_{RXDX} — **Data Hold after $\overline{RD}$ Inactive:** Time after $\overline{RD}$ is inactive that the memory system must hold Data on the bus. Always 0 ns on the 80C196KC.

TIMINGS THE 80C196KC WILL PROVIDE:

F_{XTAL} — **Frequency on XTAL1:** Frequency of signal input into the 80C196KC. The 80C196KC runs internally at ½ F_{XTAL}.

T_{OSC} — **1/F_{XTAL}:** All A.C. Timings are referenced to T_{OSC}.

T_{XHCH} — **XTAL1 High to CLKOUT High or Low:** Needed in systems where the signal driving XTAL1 is also a clock for external devices.

T_{CLCL} — **CLKOUT Cycle Time:** Nominally 2 T_{OSC}.

T_{CHCL} — **CLKOUT High Period:** Needed in systems which use CLKOUT as clock for external devices.

T_{CLLH} — **CLKOUT Falling Edge to ALE/$\overline{ADV}$ Rising:** A help in deriving other timings.

T_{LLCH} — **ALE/$\overline{ADV}$ Falling Edge to CLKOUT Rising:** A help in deriving other timings.

T_{LHLH} — **ALE Cycle Time:** Time between ALE pulses.

T_{LHLL} — **ALE/$\overline{ADV}$ High Period:** Useful in determining ALE/$\overline{ADV}$ rising edge to ADDRESS valid. External latches must also meet this spec.

T_{AVLL} — **ADDRESS Setup to ALE/$\overline{ADV}$ Falling Edge:** Length of time ADDRESS is valid before ALE/$\overline{ADV}$ falls. External latches must meet this spec.

T_{LLAX} — **ADDRESS Hold after ALE/$\overline{ADV}$ Falling Edge:** Length of Time ADDRESS is valid after ALE/$\overline{ADV}$ falls. External latches must meet this spec.

T_{LLRL} — **ALE/$\overline{ADV}$ Low to $\overline{RD}$ Low:** Length of time after ALE/$\overline{ADV}$ falls before $\overline{RD}$ is asserted. Could be needed to insure proper memory decoding takes place before a device is enabled.

Figure 16-15. AC Timing Explanations

T_{RLCL} — **$\overline{RD}$ Low to CLKOUT Falling Edge:** Length of time from $\overline{RD}$ asserted to CLKOUT falling edge: Useful for systems based on CLKOUT.

T_{RLRH} — **$\overline{RD}$ Low to $\overline{RD}$ High:** $\overline{RD}$ pulse width.

T_{RHLH} — **$\overline{RD}$ High to ALE/$\overline{ADV}$ Asserted:** Time between $\overline{RD}$ going inactive and next ALE/$\overline{ADV}$, also used to calculate time between inactive and next ADDRESS valid.

T_{RLAZ} — **$\overline{RD}$ Low to ADDRESS Float:** Used to calculate when the 80C196KC stops driving ADDRESS on the bus.

T_{LLWL} — **ALE/$\overline{ADV}$ Low Edge to $\overline{WR}$ Low:** Length of time ALE/$\overline{ADV}$ falls before $\overline{WR}$ is asserted. Could be needed to ensure proper memory decoding takes place before a device is enabled.

T_{CLWL} — **CLKOUT Falling Edge to $\overline{WR}$ Low:** Time between CLKOUT going low and $\overline{WR}$ being asserted. Useful in systems based on CLKOUT.

T_{QVWH} — **Data Valid to $\overline{WR}$ Rising Edge:** Time between data being valid on the bus and $\overline{WR}$ going inactive. Memory devices must meet this spec.

T_{CHWH} — **CLKOUT High to $\overline{WR}$ Rising Edge:** Time between CLKOUT going high and $\overline{WR}$ going inactive. Useful in systems based on CLKOUT.

T_{WLWH} — **$\overline{WR}$ Low to $\overline{WR}$ High:** $\overline{WR}$ pulse width. Memory devices must meet this spec.

T_{WHQX} — **Data Hold after $\overline{WR}$ Rising Edge:** Amount of time data is valid on the bus after $\overline{WR}$ going inactive. Memory devices must meet this spec.

T_{WHLH} — **$\overline{WR}$ Rising Edge to ALE/$\overline{ADV}$ Rising Edge:** Time between $\overline{WR}$ going inactive and next ALE/$\overline{ADV}$. Also used to calculate $\overline{WR}$ inactive and next ADDRESS valid.

T_{WHBX} — **$\overline{BHE}$, INST, Hold after $\overline{WR}$ Rising Edge:** Minimum time these signals will be valid after $\overline{WR}$ inactive.

T_{RHBX} — **$\overline{BHE}$, INST, Hold after $\overline{RD}$ Rising Edge:** Minimum time these signals will be valid after $\overline{RD}$ inactive.

T_{WHAX} — **AD8-15 Hold after $\overline{WR}$ Rising Edge:** Minimum time the high byte of the address in 8-bit mode will be valid after $\overline{WR}$ inactive.

T_{RHAX} — **AD8-15 Hold after $\overline{RD}$ Rising Edge:** Minimum time the high byte of the address in 8-bit mode will be valid after $\overline{RD}$ inactive.

Figure 16-15. AC Timing Explanations (Continued)

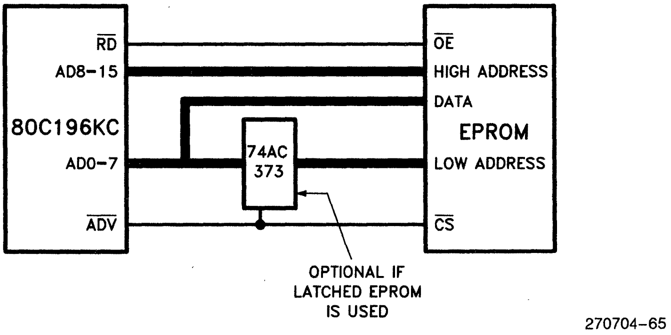
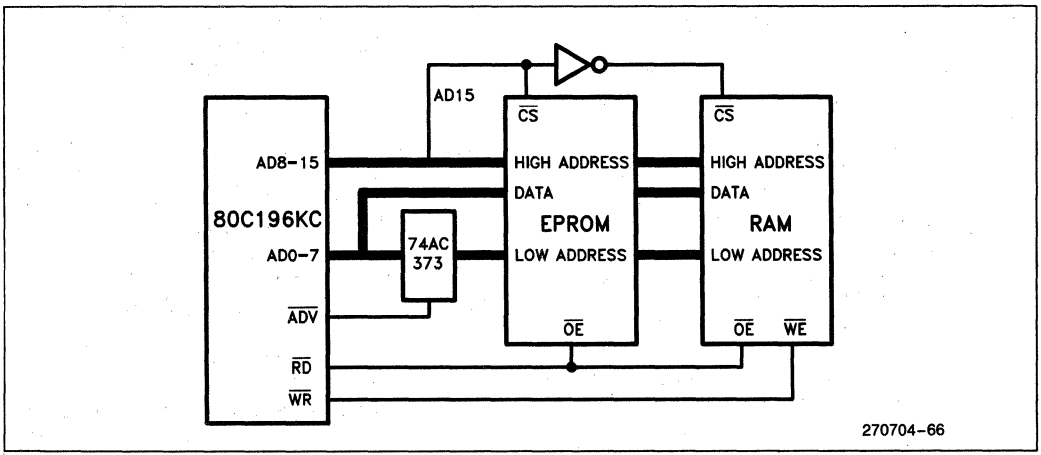


Figure 16-16. 8-Bit System with EPROM

16.6 Memory System Examples

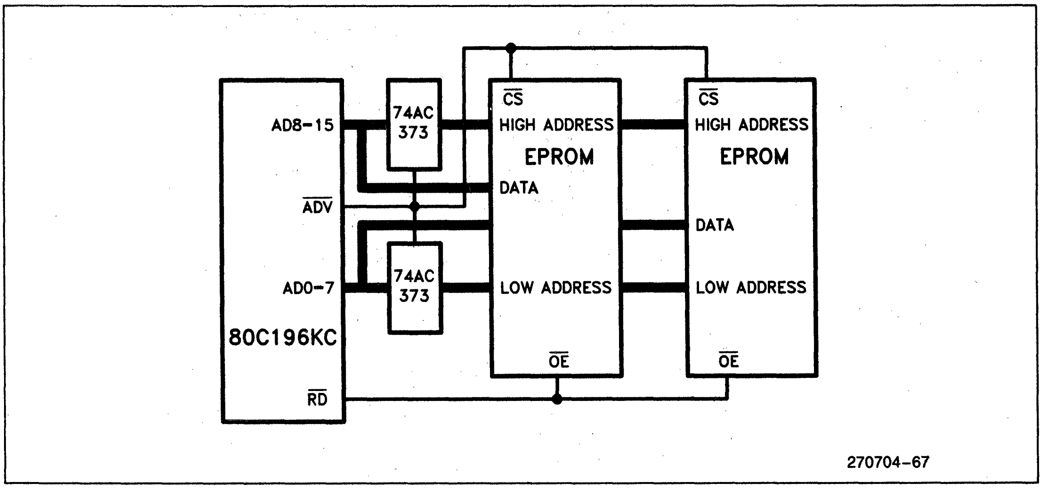
External memory systems for the 80C196KC can be set up in many different ways. Figure 16-16 shows a simple 8 bit system with a single EPROM. The $\overline{\text{ADV}}$ Mode can be selected to provide a chip select to the memory. By setting bit CCR.1 to 0, the system is locked into the eight bit mode. An eight bit system with EPROM and RAM is shown in Figure 16-17. The EPROM is decoded in the lower half of memory,and the RAM in the upper half.

Figure 16-18 shows a 16 bit system with 2 EPROMs. Again, $\overline{\text{ADV}}$ is used to chip select the memory. Figure 16-19 shows a system with dynamic bus width. Code is executed from the two EPROMs and data is stored in the single RAM. Note the Chip Select of the RAM also is input to the BUSWIDTH pin to select an eight bit cycle.



270704-66

Figure 16-17. 8-Bit System with EPROM and RAM



270704-67

Figure 16-18. 16-Bit System with EPROM

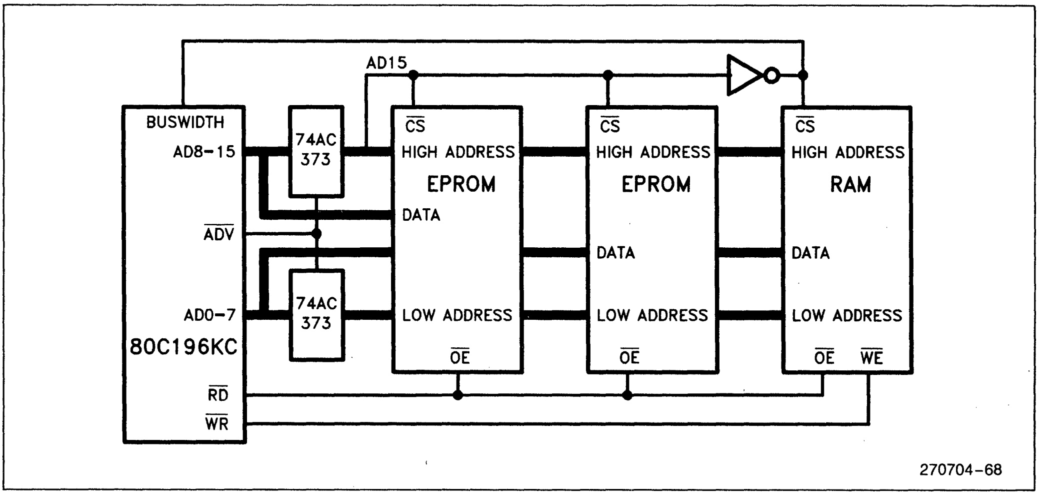


Figure 16-19. 16-Bit System with Dynamic Buswidth

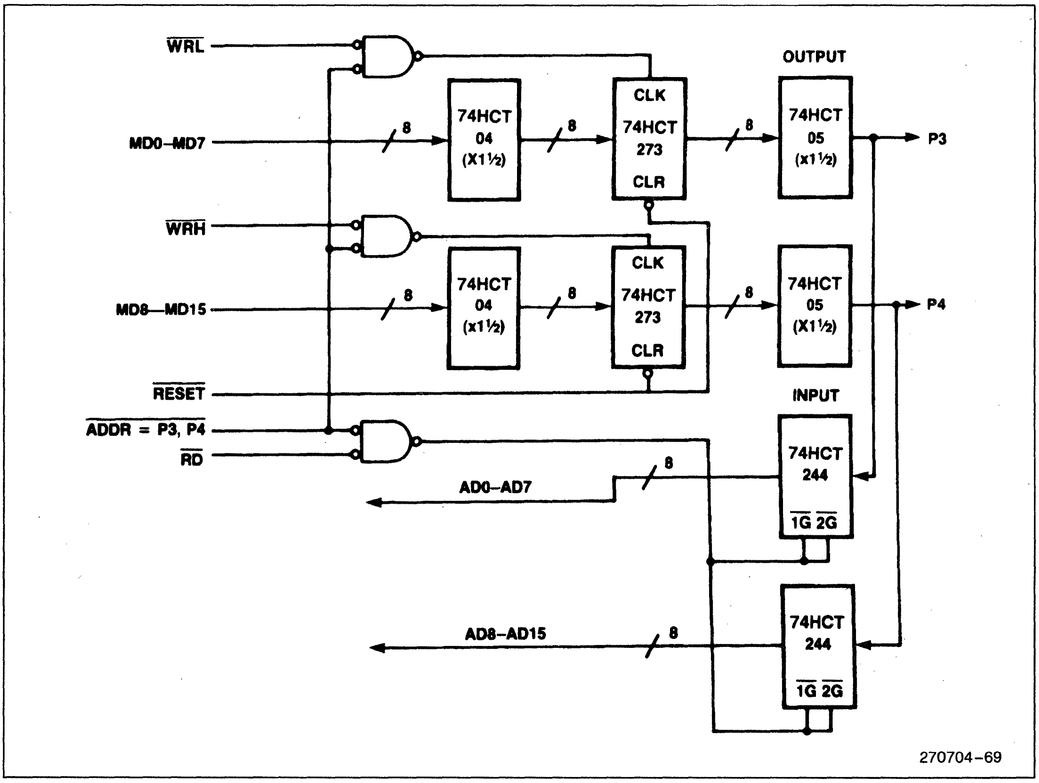


Figure 16-20. I/O Port Reconstruction

16.7 I/O Port Reconstruction

When a single-chip system is being designed using a multiple chip system as a prototype, it may be necessary to reconstruct I/O Ports 3 and 4 using a memory mapped I/O technique. The circuit to reconstruct the Ports is shown in Figure 16-20. It can be attached to a 80C196KC system which has the required address decoding and bus demultiplexing.

The output circuitry is a latch that operates when 1FFEH or 1FFFH are placed on the MA lines. The inverters surrounding the latch create an open-collector output to emulate the open-drain output found on the 80C196KC. The $\overline{\text{RESET}}$ line sets the ports to all 1s when the chip is reset. The voltage and current specifications of the port will be different from the 80C196KC, but the functionality will be the same.

The input circuitry is a bus transceiver that is addressed at 1FFEH and 1FFFH. If the ports are going to be either inputs or outputs, but not both, some of the circuitry can be eliminated.

17.0 USING ROM AND EPROM PARTS

The 87C196KC contains 16K bytes of ultraviolet Erasable and Electrically Programmable Read Only Memory (EPROM). The 83C196KC contains 16K bytes of Read Only Memory (ROM).

Three flexible EPROM programming modes are available on the 87C196KC; Auto, Slave, and Run-time. These modes can program the 87C196KC in a stand alone or run-time environment.

For ROM parts, a ROM dump mode allows the ROM contents to be verified by the user.

17.1 Programming the 87C196KC

The EPROM is mapped into memory locations 2000H–5FFFH if $\overline{\text{EA}}$ is at logical 1. By applying +12.50V to $\overline{\text{EA}}$ when $\overline{\text{RESET}}$ is asserted places the 87C196KC in Programming Mode. The Programming Mode supports programming and verification of 87C196KC EPROMs.

The Auto Programming Mode enables an 87C196KC to program itself with the 16K bytes of code beginning at address 4000H on its external bus. The Slave Mode provides a standard interface for an EPROM programmer. The Run-time Mode allows individual EPROM locations to be programmed at run-time under complete software control.

In the Programming Mode, some I/O pins have new functions. These new pin functions determine and support the different programming modes. Figure 17-1 shows how the pins are renamed and Figure 17-2 describes in detail each new pin function.

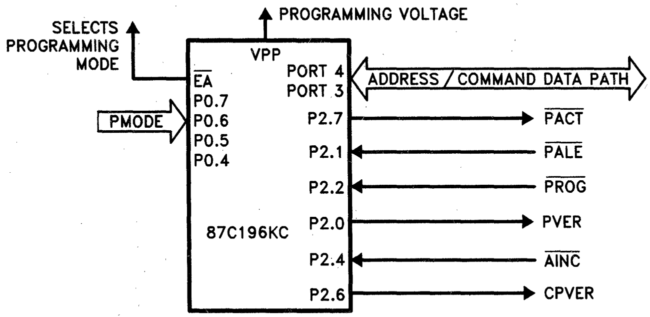


Figure 17-1. Programming Mode Pin Functions

Name	Function
PMODE	Programming Input Mode Select. Determines the EPROM programming algorithm that is performed. PMODE is sampled after a chip reset and should be static while the part is operating.
$\overline{\text{PALE}}$	Programming ALE Input. Accepted by an 87C196KC that is in Slave Programming Mode. Used to indicate that Ports 3 and 4 contain a command/address.
$\overline{\text{PROG}}$	Programming Input. Falling edge indicates valid data on PBUS and the beginning of programming. Rising edge indicates end of programming.
$\overline{\text{PACT}}$	Programming Active. Output indicates when programming activity is complete.
PVER	Program Verification. Output signal is low after rising edge of PROG if the programming was not successful.
$\overline{\text{AINC}}$	Auto Increment. Active low input signal indicates that the auto increment mode is enabled. Auto Increment will allow reading or writing of sequential EPROM locations without address transactions across the PBUS for each read or write.
PORTS	Address/Command/Data Bus. Used to pass commands, addresses and data to and from 87C196KCs. Also used in the Auto Programming Mode as a regular system bus to access external memory.
CPVER	Cummulative Program Output Verification. Pin is high if all locations since entering a programming mode have programmed correctly.

Figure 17-2. Programming Mode Pin Definitions

While in Programming Mode, the PMODE value selects the programming function (see Figure 17-3). Runtime programming can be done at any time during normal execution.

PMODE	Programming Mode
0–4	Reserved
5	Slave Programming
6	ROM Dump
7–8	Reserved
9	UPROM Programming
0AH–0BH	Reserved
0CH	Auto Programming
0DH	PCCB Programming
0EH–0FH	Reserved

Figure 17-3. Programming Function Pmode Values

To guarantee proper functionality, the pins of PMODE must be in their desired state before $\overline{\text{RESET}}$ rises. Once the part is reset, it should not be switched to another mode without a new reset sequence.

When $\overline{\text{EA}}$ selects the Programming Mode, the chip reset sequence loads the CCR from the PCCB (Programming Chip Configuration Byte). The PCCB is a separate EPROM location that is not mapped under normal operation. PCCB has implications in some of the programming modes, and also for the memory protection options, discussed later. Therefore, the PCCB must correctly correspond to the memory system in the programming setup, which is not necessarily the memory system of the application.

The following sections describe the 87C196KC programming modes in detail.

17.2 Auto Programming Mode

The Auto Programming Mode allows an 87C196KC EPROM to be programmed without a special EPROM programmer. In this mode, the 87C196KC simply programs itself with the data found at external locations 4000H–7FFFH. Figure 17-4 shows a minimum configuration using an 16K × 8 EPROM to program an 87C196KC in the Auto Programming Mode.

To start the Auto Programming Mode, $\overline{\text{PACT}}$ is asserted and the word at external location 4014H is read to determine the programming pulse width. This allows Auto Programming to be done at several different frequencies. The formula for calculating the pulse width is:

$$\text{Pulse Width} = \text{XTAL1}/1{,}600{,}000 - 1$$

The MSB of this word also must be set. For example, with a 10 MHz Clock, the word at 4014H should be loaded with 8005H for the correct pulse width.

The 87C196KC then reads a word from external memory, and the Modified Quick-Pulse Programming Algorithm (described later) programs the corresponding EPROM location. Since the erased state of a byte is 0FFH, the Auto Programming Mode skips locations containing 0FFH. PVER will go low anytime there is a programming error. When all 16K have been programmed, $\overline{\text{PACT}}$ goes high.

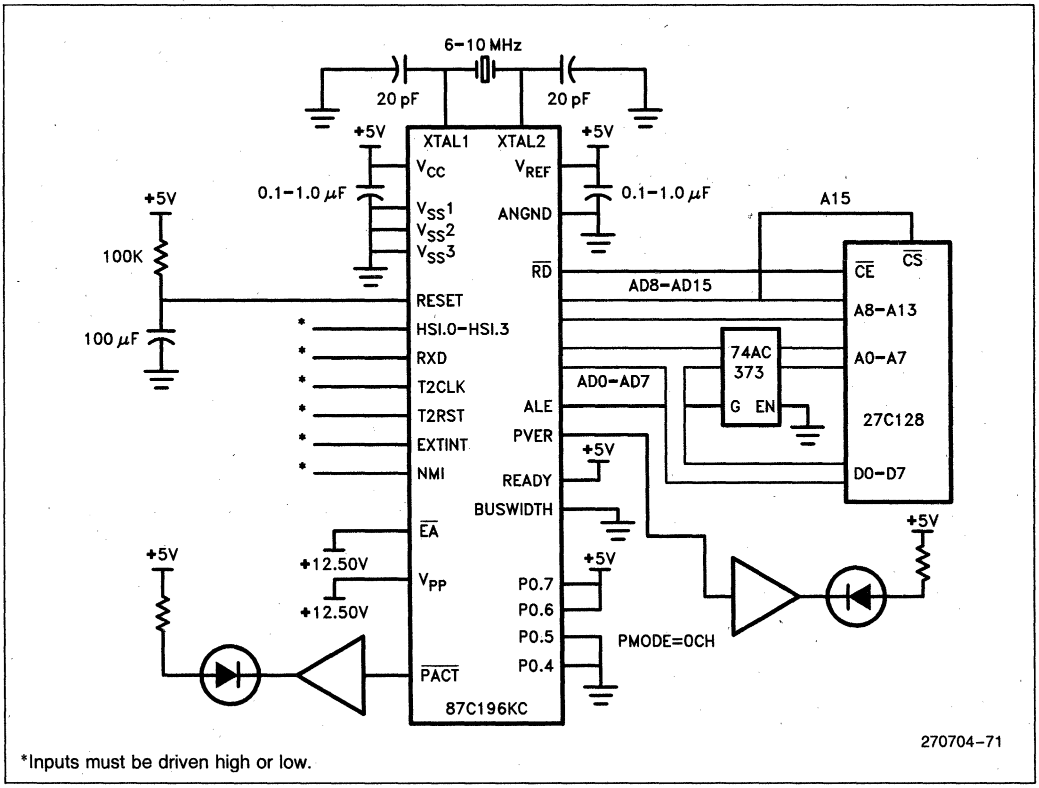
In the Auto Programming Mode, CCR is loaded with the PCCB. The PCCB must correspond to the memory system of the programming setup, which is not necessarily the same as the memory system of the actual application.

V_{PP} and EA must be kept noise-free and must never go above 12.75V at any time.

PCCB Programming Mode

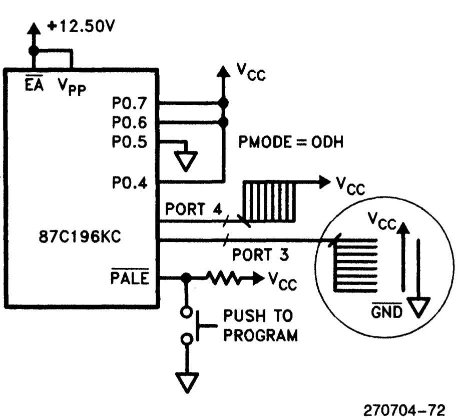
The PCCB (Programming Chip Configuration Byte) can be treated just like any other EPROM location. When in the Slave Programming Mode, the PCCB can be programmed at location 0218H. But the PCCB Programming Mode is a simple way to program the PCCB when no other locations need be programmed. Figure 17-5 shows a block diagram for using the PCCB Programming Mode.

With PMODE = 0DH and 0FFH on Port 4, the PCCB will be programmed with the data on Port 3 when a 0 is placed on PALE. After programming is complete, PVER is driven high if the PCCB was programmed correctly, and low if the programming failed. PALE can be pulsed to repeat programming.



*Inputs must be driven high or low.

Figure 17-4. Auto Programming Mode



NOTES:

Tie Port 3 to the value desired to be programmed into PCCB.

Make all necessary minimum connections for power, ground and clock.

Figure 17-5. The PCCB Programming Mode

17.3 Slave Programming Mode

A 87C196KC can be programmed by a master programmer through the Slave Programming Mode.

In this mode, the 87C196KC programs like a simple EPROM device. The 87C196KC responds to two different commands while in this mode: data program, and word dump. These commands, along with the transfer of appropriate data and addresses are selected using Ports 3 and 4 and five other pins for handshaking. The least significant bit selects a Data Program or Word Dump and the upper 15 bits contain the address to be programmed or dumped. The address ranges from 2000H-5FFFH and refers to internal EPROM space.

Data Program Command

A Data Program Command is illustrated in Figure 17-6. The data program command is selected by setting the LSB of the address to a 1. For example, an address of 3501H would program the word location at 3500H. Asserting PALE latches the command and address to be programmed from Ports 3 and 4. PROG also asserts to latch the data. The width of the PROG pulse determines the length of the programming pulse.

After the rising edge of PROG, the 87C196KC automatically performs a verification of the address just programmed. PVER is asserted if the location programmed correctly. This gives verification information to programmers which can not use the Word Dump Command. CPVER is a cummulative program verify that will remain low if a location did not verify correctly. The AINC pin can optionally be toggled in order to increment to the next location or a new Data Program Command can be issued.

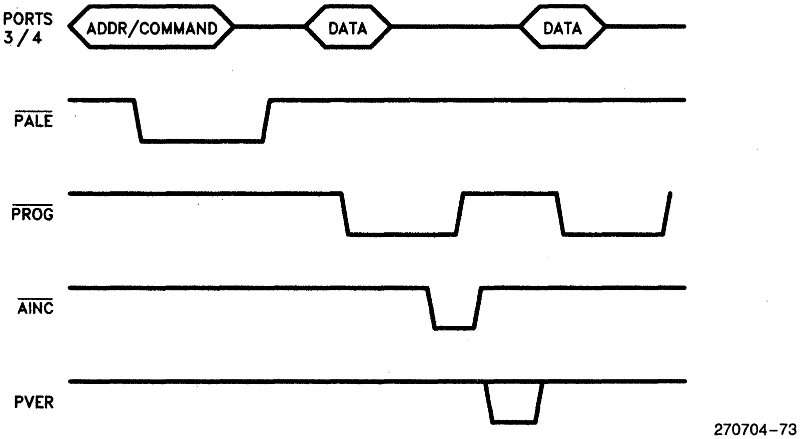


Figure 17-6. Data Program Command

Figure 17-6 shows the relationship of $\overline{\text{PALE}}$, $\overline{\text{PROG}}$, PVER, $\overline{\text{AINC}}$ and CPVER to the Command/Data path on Ports 3 and 4 for the Data Program Command.

Word Dump Command

When the Word Dump Command is issued, the 87C196KC places the value at the requested address on Ports 3 and 4. A Word Dump command is selected by a 0 on Port pin 3.0. For example, sending the command 2100H to a slave results in the slave placing the word at internal address 2100H on Ports 3 and 4. $\overline{\text{PROG}}$ governs when the 87C196KC drives the bus. The Timings are shown in Figure 17-7. In the Word Dump mode, the $\overline{\text{AINC}}$ pin can remain active and toggling. The $\overline{\text{PROG}}$ pin will automatically increment the address.

17.4 Run-Time Programming

In Run-Time Programming, the user can program an EPROM location during the normal execution of code. The only additional requirement of Run-Time Programming is programming voltage is applied V$_{PP}$. Run-Time Programming is done with $\overline{\text{EA}}$ at a TTL high.

To Run-Time Program, the user writes to the location to be programmed. Figure 17-8 is the recommended code sequence for Run-Time Programming. The EPROM cannot be accessed during Run-Time Programming. Therefore, the part must enter the IDLE mode immediately after writing to the EPROM or begin executing immediately from external memory. The Modified Quick Pulse algorithm guarantees the programmed EPROM cell for the life of the part.

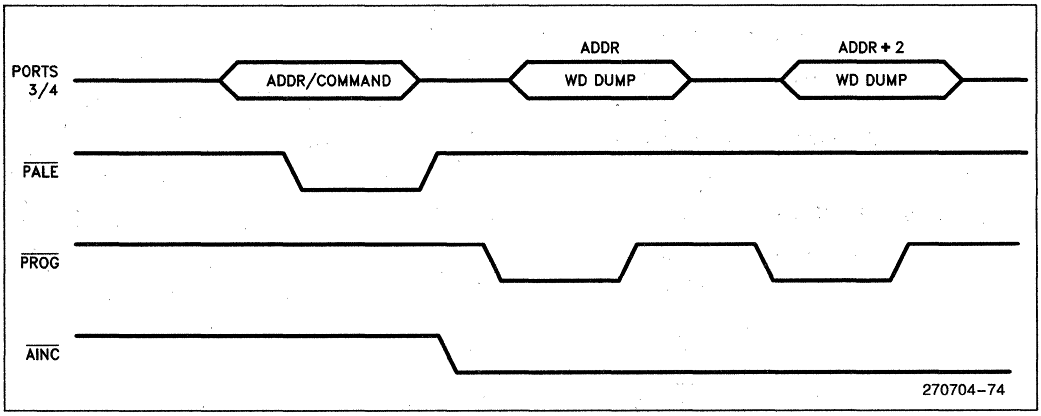


Figure 17-7. Word Dump Command

```
PROGRAM:     POP ADDRESS_TEMP                      ;Load program data
             POP DATA_TEMP                         ;and address
             PUSHF
             LD COUNT, #25T                        ;program using Modified
                                                    Quick Pulse
    LOOP:    LDB INT_MASK,#ENABLE_SWT              ;program SWT for
             LDB HSO_COMMAND,#SWTO_OVF             ;program pulse width
             ADD HSO_TIME,TIMER1,#PROGRAM_PULSE
             EI
             ST DATA_TEMP,[ADDR_TEMP]              ;enter idle mode until
             IDLPD 1                               ;swt expires
             DJNZ COUNT, LOOP                      ;loop 25 times
             POPF
             RET

SWT_EXPIRED:                                       ;service swt and return
             RET
```

Figure 17-8. Run-Time Programming Algorithm Example

17.5 ROM/EPROM Memory Protection Options

Write protection is available for EPROM parts, and read protection is provided for both ROM and EPROM parts.

Write protection is enabled by setting the LOC0 bit in either CCR or the PCCB to zero. Figure 17-9 summarizes the different write protection options. When write protection is enabled, the bus controller will cycle through the write sequence but will not actually drive data to the EPROM or enable V_{PP}. This protects the entire EPROM 2000–5FFFH from inadvertent or unauthorized programming. With write protection enabled, the PTS cannot read the internal EPROM.

CCB LOC0	PCCB LOC0	Protection
1	1	Unprotected EPROM. Writes to Internal Memory always allowed.
1	0	Run-Time Programming Allowed. No programming modes allowed
0	1	Run-Time programming not allowed. Programming modes allowed after key verification (if needed).
0	0	All programming unconditionally disabled.

Figure 17-9. Write Protection Options

Read protection is enabled by setting the LOC1 bit of CCR. When read protection is selected, the bus controller will only perform a data read from the address range 2020H–2020FH and 2050H–5FFFH if the Slave Program Counter is in the range 2000H–5FFFH. Since the Slave PC can be as many as 4 bytes ahead of the CPU program counter, an instruction after address 5FFAH may not access protected memory. The interrupt vectors and CCB are not read protected since interrupts can occur even when executing out of external memory. LOC1 in the PCCB can be programmed, but has no memory protection implications.

Also, two UPROM (Unerasable PROM) bits are implemented on the 87C196KC for additional memory protection. The DEI (Disable External Instruction fetch) bit disables the bus controller from executing external instruction fetches if the bit is a 0. If an attempt is made to load the Slave PC with an external address, the part will Reset itself. The automatic Reset also gives extra protection against runaway code.

Because of the prefetch queue in the bus controller, code cannot be executed from the last four bytes of internal memory if this feature is enabled.

The DED (disable external Data fetch) bit disables the bus controller from executing external data reads and writes if the bit is 0. If a data access is requested from the bus controller, the part will Reset itself.

For more information on UPROMs, see Section 17.5.

Authorized Access of Protected Memory

A "Security Key" mechanism has been implemented for authorized access of protected internal memory to test internal ROM/EPROM. This allows users to verify the EPROM array and still keep their code protected.

The security key is a 128-bit number located in internal memory at locations 2020H–202FH. The user programs his own security key into these locations. Figure 17-10 shows the different programming modes and whether a security key verification is performed to allow entry into the programming mode.

Programming Mode	Key Verification
Slave Programming	If Protected*
ROM Dump	Always
UPROM Programming	Never
AUTO Programming	If Protected*
PCCB Programming	Never

*If read or write protected in CCR.

Figure 17-10. Security Key Verification

The ROM Dump Mode is an easy way to verify the contents of the EPROM or ROM array. The ROM Dump writes out the entire EPROM or ROM to locations 4000H–7FFFH in external memory. If the DED (Disable External Data Fetches) bit has been programmed in the USFR, this mode is disabled entirely (see Section 17.5).

The ROM Dump Mode is selected with PMODE = 06H and always begins with a security key verification. The user puts the same security key at external locations 4020H–402FH that he has programmed in internal locations 2020H-202FH. Before doing a ROM dump, the 80C196KC compares the two security keys. If they match, the 80C196KC dumps the array to external memory. If they do not match, the 80C196KC enters an endless loop of internal execution which can be exited only by a chip reset.

When using the Auto Programming Mode, a security key verification is done if the CCB has read and or write protection enabled. The security key must reside in external memory locations 4020H-402FH and the two keys must match or the 80C196KC enters an endless internal loop.

In the Slave Programming Mode, a security key verification is also performed if CCB has read and or write protection enabled. The key verify is done somewhat differently in the Slave Programming Mode. The user must "program" the correct key into the array at locations 2020H–202FH using the Data Program Command described in Section 17.3. The locations are not actually programmed, but the data is compared to the internal security key. The key must be entered sequentially, and if any of the locations is "programmed" incorrectly, the 80C196KC will enter the elusive endless internal loop.

If CCB has any protection enabled, the security key is write protected to keep unauthorized users from overwriting the key with a known security key.

If the PCCB is programmed with any read or write protection, there is no way to enter any of the programming modes. So the last thing that should be done to protect the part from unauthorized access, is to program the PCCB.

17.6 UPROMs

Unerasable PROM (UPROM) devices are implemented on the 87C196KC for some additional security features. The USFR (UPROM Special Function Register) along with what bits are available for the ROM and EPROM versions is shown in Figure 17-11. The UPROM bits act as fuses, i.e., they can be programmed, but not erased.

	7							0
*	RSV	RSV	RSV	RSV	DEI	DED	RSV	RSV

USFR

	DEI	DED
87C196KC	UPROM Bit	UPROM Bit
83C196KC	N/A	N/A

*Always program reserved (RSV) bits to 1

Figure 17-11. USFR

Because the UPROM bits cannot be erased, the bits cannot be tested by Intel prior to shipment. Intel does, however, test the different features enabled by UPROM bits. Therefore, only defects within the UPROM cells themselves (rather unlikely) will be undetectable. When programming a UPROM bit, it can be verified to make sure that it did indeed program. Customers are advised that devices with programmed UPROM bits cannot be returned to Intel for failure analysis.

Programming UPROMs

The UPROM bits in the USFR can be programmed using the Slave Programming Mode, or with the separate UPROM Programming Mode. Figure 17-12 shows the locations to program a UPROM bit and the data in the Slave Programming Mode.

Bit	Address	Data
DED	0758H	0004H
DEI	0718H	0008H

Figure 17-12. Programming UPROMs in the Slave Programming Mode

The UPROM Programming mode works much the same way as the PCCB Programming Mode (see Figure 17-5). PMODE must be 09H, the value to program the USFR is forced onto port 3, 0FFH is forced on port 4, and PALE is pulled low to begin programming. A 1 in a bit location means to program the associated UPROM bit. $\overline{\text{PACT}}$ remains low while programming and PVER indicates if the data was programmed correctly.

The Modified Quick Pulse Algorithm

The Modified Quick Pulse Algorithm must be used to guarantee programming over the life of the EPROM in Run-Time and Slave Programming Modes.

The Modified Quick-Pulse Algorithm calls for each EPROM location to receive 25 separate 100 μs (± 5 μs) programming cycles. Verification is done after the 25th pulse. If the location verifies, the next location is programmed. If the location fails to verify, the location fails the programming sequence.

Once all locations are programmed and verified, the entire EPROM is again verified.

Programming of 87C196KC EPROMs is done with V_{PP} = 12.50V ±0.25V and V_{CC} = 5.0V ±0.5V.

Signature Word

The 87C196KC contains a signature word. The word can be accessed in the Slave Mode by executing a Word Dump Command at memory location 70H. The programming voltages are determined by reading locations 72H and 73H in Slave Programming Mode. The voltages are calculated by using the following equation.

Voltage = 20/256 × (test ROM data)

The values for the signature word and voltage levels are shown in Figure 17-13.

Description	Location	Value
Signature Word	70H	879CH
Programming V_{CC}	72H	040H 5.0V
Programming V_{PP}	73H	0A0H 12.50V

Figure 17-13. Signature Word and Voltage Levels

Erasing the 87C196KC

After each erasure, all bits of the 87C196KC are logical "1s". Data is introduced by selectively programming "0s". The only way to change a "0" to a "1" is by exposure to ultraviolet light.

The erasure characteristics of the 87C196KC are so that erasure begins upon exposure to light with wavelengths shorter than approximately 4000Å. It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000Å–4000Å range.

Opaque labels must always be placed over the Window to prevent unintentional erasure. In the Powerdown Mode, the part will draw more current than specified if the EPROM window is exposed to light.

The recommended erasure procedure for the 87C196KC is exposure to ultraviolet light which has a wavelength of 2537Å.

18.0 80C196KC to 80C196KB

18.1 New Features of the 80C196KC

1. The 80C196KC has 488 bytes of RAM and is available with 16K of EPROM/ROM. The 80C196KB has 232 bytes of RAM and 8K of EPROM/ROM. A Vertical Windowing scheme has been implemented to allow the extra 256 bytes of RAM to be accessed as Registers. See Section 3.3.
2. The 80C196KC has an additional 2 PWM outputs over the 80C196KB. See Section 8.0.
3. The 80C196KC has the Peripheral Transaction Server (PTS), which can greatly reduce interrupt servicing overhead. See Section 6.0.
4. Timer2 can now be clocked internally as well as externally. Timer2 can clock internally every 1 or 8 states selectable by software. See Section 7.0.
5. The HSO has one new command that allows all of the pins to be addressed simultaneously. See Section 10.0.
6. The A/D on the 80C196KC has an 8-bit conversion mode as well as 10-bit. Also, the A/D has selectable sample and convert times. See Section 12.0.
7. The 80C196KC has 2 UPROM (unerasable programmable read only memory) for additional security enhancements. See Section 17.5.

18.2 Converting 80C196KB Designs to 80C196KC Designs

1. **Clock Detect Enable Pin (CDE)** —The CDE pin on the 80C196KB is a V_{SS} pin on the 80C196KC. An extra V_{SS} pin was needed on the 80C196KC to support the higher clock rates.

2. **DJNZW Instruction** —The DJNZW will work properly on the 80C196KC. However, it will take 6/10 state times rather than 5/9 state times as on the 80C196KB.
3. **Internal Reset Pulse** —The Reset pulse on the 80C196KC has been increased to 16 state times. There are 2 ways a 80C196KB or 80C196KC can internally assert the $\overline{\text{RESET}}$ pin; Watchdog Timer Overflow, and a RST instruction. On the KB, the Reset Pulse is 4 state times. Because the 80C196KC and future proliferations will run at much higher frequencies, a 4 state time reset pulse was deemed unreasonably small. This should cause no problems in most applications.
4. **Memory Map** —Because the 80C196KC has 16K of EPROM/ROM and 512 bytes of RAM, twice that of the 80C196KB, the memory map is different to accommodate the extra memory. Locations 100–1FFH contain the additional 256 bytes of RAM on the 80C196KC. On the EPROM/ROM versions, locations 4000–5FFFH contain the extra 8K of EPROM.
5. **ONCE Mode Entry** —The 80C196KC enters the ONCE mode by holding the TXD pin low on the rising edge of RESET. See Section 15.3.
6. **EPROM Programming Modes** —Gang Programming with the Slave Programming Mode is no longer supported. Also, the Auto-Programming and Slave Programming Modes have been modified to support the 16K of onboard memory. This should only affect manufacturers of EPROM programmers, and should not have an impact on the end user.
7. **A/D Converter** —An A/D conversion takes 1.5 state times less on the 80C196KC in both the fast and slow conversion modes.
8. **HOLD/HLDA** —The $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WRC}}$, INST, $\overline{\text{BHE}}$, $\overline{\text{ADV}}$ and ALE pins are weakly held in their inactive states during HOLD on the 80C196KC. On the 80C196KB, only ALE is weakly held during HOLD.
9. **The PSW** —The PSW on the 80C196KC has an extra bit (PSE) to support the PTS. This bit was reserved on the 80C196KB.
10. **HSO** —HSO commands 0CH and 0DH were reserved on the 80C196KB. On the 80C196KC, 0CH becomes a new command.
11. **WSR** —WSR bits that were reserved to 0 on the 80C196KB must be 0 to be 80C196KC compatible.

ADVANCE INFORMATION

8XC196KC
16-BIT HIGH PERFORMANCE CHMOS MICROCONTROLLER

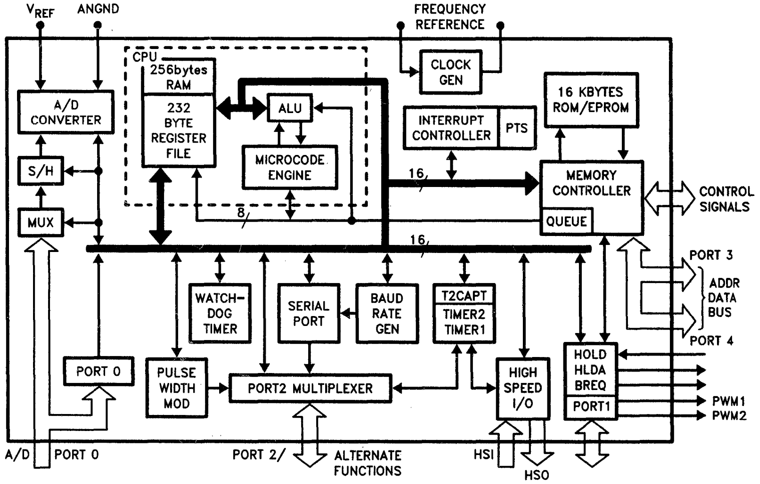
87C196KC—16 Kbytes of On-Chip EPROM
80C196KC—ROMless
83C196KC—16 Kbytes of On-Chip ROM

- **16 MHz Operation**
- **232 Byte Register File**
- **256 Bytes of Additional RAM**
- **Register-to-Register Architecture**
- **28 Interrupt Sources/16 Vectors**
- **Peripheral Transaction Server**
- **1.75 μs 16 x 16 Multiply (16 MHz)**
- **3.0 μs 32/16 Divide (16 MHz)**
- **Powerdown and Idle Modes**
- **Five 8-Bit I/O Ports**
- **16-Bit Watchdog Timer**
- **Dynamically Configurable 8-Bit or 16-Bit Buswidth**
- **Full Duplex Serial Port**
- **High Speed I/O Subsystem**
- **16-Bit Timer**
- **16-Bit Up/Down Counter with Capture**
- **3 Pulse-Width-Modulated Outputs**
- **Four 16-Bit Software Timers**
- **8- or 10-Bit A/D Converter with Sample/Hold**
- **$\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ Bus Protocol**
- **OTP One-Time Programmable Version**

The 80C196KC 16-bit microcontroller is a high performance member of the MCS®-96 microcontroller family. The 80C196KC is an enhanced 80C196KB device with 488 bytes RAM, 16 MHz operation and an optional 16 Kbytes of ROM/EPROM. Intel's CHMOS IV process provides a high performance processor along with low power consumption.

The 87C196KC is an 80C196KC with 16 Kbytes on-chip EPROM. In this document, the 80C196KC will refer to all products unless otherwise stated.

Four high-speed capture inputs are provided to record times when events occur. Six high-speed outputs are available for pulse or waveform generation. The high-speed output can also generate four software timers or start an A/D conversion. Events can be based on the timer or up/down counter.



270741-1

Figure 1. 80C196KC Block Diagram

MCS®-96 is a registered trademark of Intel Corporation.

October 1989
Order Number: 270741-002

80C196KC Enhanced Feature Set over the 80C196KB

1. The 80C196KC has twice the RAM and twice the ROM/EPROM of the 80C196KB. Also, a Vertical Register Windowing Scheme allows the extra 256 bytes of RAM to be used as registers. This greatly reduces the context switching time.
2. Peripheral Transaction Server (PTS). The PTS is an alternative way to service an interrupt, reducing latency and overhead. Each interrupt can be mapped to its PTS channel, which acts like a DMA channel. Each interrupt can now do a single or block transfer, without executing an Interrupt service routine. Special PTS modes exist for the A/D converter, HSI, and HSO.
3. Two extra Pulse Width Modulated outputs. The 80C196KC has added 2 PWM outputs that are functionally compatible to the 80C196KB PWM.
4. Timer2 Internal Clocking. Timer2 can now be clocked with an internal source, every 1 or 8 state times.
5. The A/D can now perform an 8- as well as a 10-bit conversion. This trades off resolution for a faster conversion time.
6. Additional On-chip Memory Security. Two UPROM (Uneraseable Programmable Read Only Memory) bits can be programmed to disable the bus controller for external code and data fetches. Once programmed, a UPROM bit cannot be erased. By shutting off the bus controller for external fetches, no one can try and gain access to your code by executing from external memory.
7. New Instructions. The 80C196KC has 5 new instructions. An exchange (XCHB/XCHW) instruction swaps two memory locations, an Interruptable Block Move Instruction (BMOVI), a Table Indirect Jump (TIJMP) instruction, and two instructions for enabling and disabling the PTS (EPTS/DPTS).

The 80C196KC User's Guide contains a complete description of the feature set, order #270704.

PACKAGING

Pin	Description	Pin	Description	Pin	Description
9	ACH7/P0.7	54	AD6/P3.6	31	P1.6/$\overline{\mathrm{HLDA}}$
8	ACH6/P0.6	53	AD7/P3.7	30	P1.5/$\overline{\mathrm{BREQ}}$
7	ACH2/P0.2	52	AD8/P4.0	29	HSO.1
6	ACH0/P0.0	51	AD9/P4.1	28	HSO.0
5	ACH1/P0.1	50	AD10/P4.2	27	HSO.5/HSI.3
4	ACH3/P0.3	49	AD11/P4.3	26	HSO.4/HSI.2
3	NMI	48	AD12/P4.4	25	HSI.1
2	$\overline{\mathrm{EA}}$	47	AD13/P4.5	24	HSI.0
1	V_{CC}	46	AD14/P4.6	23	P1.4/PWM2
68	V_{SS}	45	AD15/P4.7	22	P1.3/PWM1
67	XTAL1	44	T2CLK/P2.3	21	P1.2
66	XTAL2	43	READY	20	P1.1
65	CLKOUT	42	T2RST/P2.4	19	P1.0
64	BUSWIDTH	41	$\overline{\mathrm{BHE}}/\overline{\mathrm{WRH}}$	18	TXD/P2.0
63	INST	40	$\overline{\mathrm{WR}}/\overline{\mathrm{WRL}}$	17	RXD/P2.1
62	ALE/$\overline{\mathrm{ADV}}$	39	PWM0/P2.5	16	$\overline{\mathrm{RESET}}$
61	$\overline{\mathrm{RD}}$	38	P2.7/T2CAPTURE	15	EXTINT/P2.2
60	AD0/P3.0	37	V_{PP}	14	V_{SS}
59	AD1/P3.1	36	V_{SS}	13	V_{REF}
58	AD2/P3.2	35	HSO.3	12	ANGND
57	AD3/P3.3	34	HSO.2	11	ACH4/P.04
56	AD4/P3.4	33	P2.6/T2UP-DN	10	ACH5/P.05
55	AD5/P3.5	32	P1.7/$\overline{\mathrm{HOLD}}$		

Figure 2. Pin Definitions

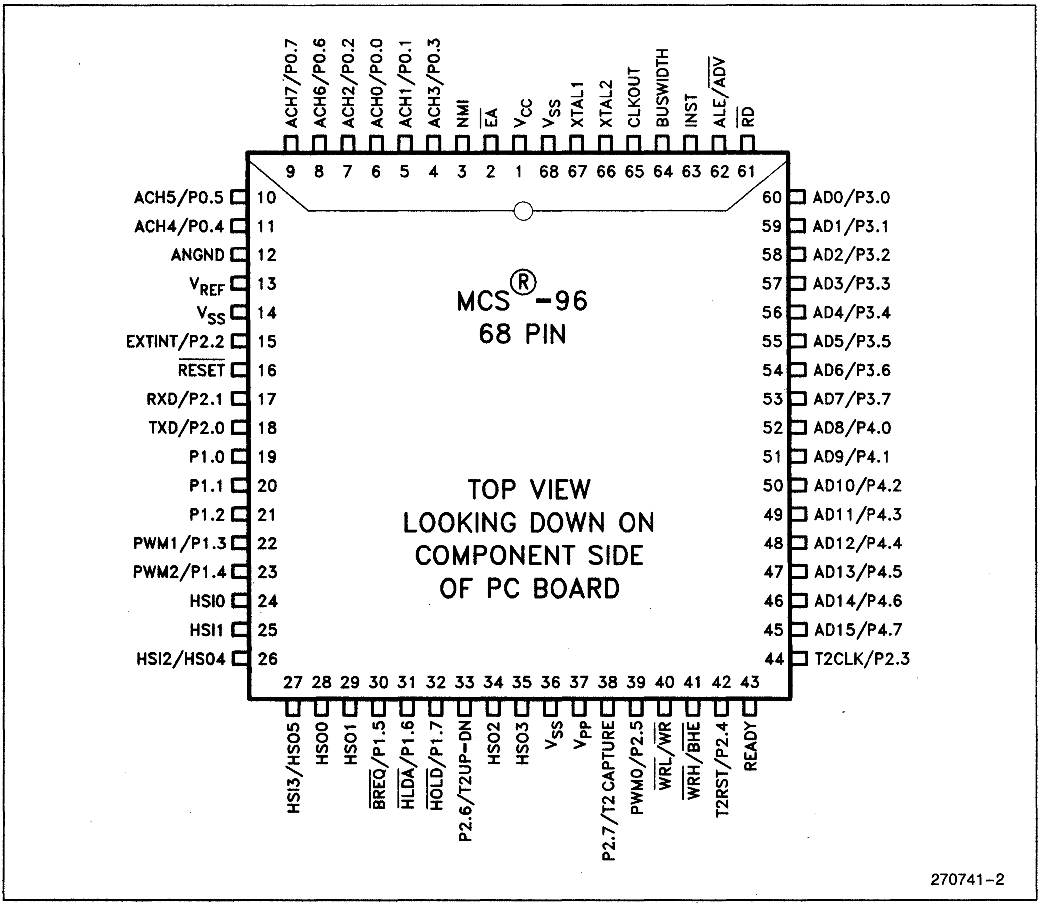


Figure 3. 68-Pin Cerquad and PLCC Package

Table 1. Prefix Identification

	CERQUAD	PLCC
80C196KC		N80C196KC
83C196KC		N83C196KC
87C196KC	CJ87C196KC	N87C196KC *

*OTP Version

PIN DESCRIPTIONS

Symbol	Name and Function
V_{CC}	Main supply voltage (5V).
V_{SS}	Digital circuit ground (0V). There are three V_{SS} pins, all of which must be connected.
V_{REF}	Reference voltage for the A/D converter (5V). V_{REF} is also the supply voltage to the analog portion of the A/D converter and the logic used to read Port 0. Must be connected for A/D and Port 0 to function.
ANGND	Reference ground for the A/D converter. Must be held at nominally the same potential as V_{SS}.
V_{PP}	Timing pin for the return from powerdown circuit. Connect this pin with a 1 μF capacitor to V_{SS} and a 1 MΩ resistor to V_{CC}. If this function is not used V_{PP} may be tied to V_{CC}. This pin is the programming voltage on the EPROM device.
XTAL1	Input of the oscillator inverter and of the internal clock generator.
XTAL2	Output of the oscillator inverter.
CLKOUT	Output of the internal clock generator. The frequency of CLKOUT is ½ the oscillator frequency.
$\overline{\text{RESET}}$	Reset input to the chip.
BUSWIDTH	Input for buswidth selection. If CCR bit 1 is a one, this pin selects the bus width for the bus cycle in progress. If BUSWIDTH is a 1, a 16-bit bus cycle occurs. If BUSWIDTH is a 0 an 8-bit cycle occurs. If CCR bit 1 is a 0, the bus is always an 8-bit bus.
NMI	A positive transition causes a vector through 203EH.
INST	Output high during an external memory read indicates the read is an instruction fetch. INST is valid throughout the bus cycle. INST is activated only during external memory accesses and output low for a data fetch.
$\overline{\text{EA}}$	Input for memory select (External Access). $\overline{\text{EA}}$ equal to a TTL-high causes memory accesses to locations 2000H through 5FFFH to be directed to on-chip ROM/EPROM. $\overline{\text{EA}}$ equal to a TTL-low causes accesses to those locations to be directed to off-chip memory.
ALE/$\overline{\text{ADV}}$	Address Latch Enable or Address Valid output, as selected by CCR. Both pin options provide a signal to demultiplex the address from the address/data bus. When the pin is $\overline{\text{ADV}}$, it goes inactive high at the end of the bus cycle. ALE/$\overline{\text{ADV}}$ is activated only during external memory accesses.
$\overline{\text{RD}}$	Read signal output to external memory. $\overline{\text{RD}}$ is activated only during external memory reads.
$\overline{\text{WR}}$/$\overline{\text{WRL}}$	Write and Write Low output to external memory, as selected by the CCR. $\overline{\text{WR}}$ will go low for every external write, while $\overline{\text{WRL}}$ will go low only for external writes where an even byte is being written. $\overline{\text{WR}}$/$\overline{\text{WRL}}$ is activated only during external memory writes.
$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	Bus High Enable or Write High output to external memory, as selected by the CCR. $\overline{\text{BHE}}$ = 0 selects the bank of memory that is connected to the high byte of the data bus. A0 = 0 selects the bank of memory that is connected to the low byte of the data bus. Thus accesses to a 16-bit wide memory can be to the low byte only (A0 = 0, $\overline{\text{BHE}}$ = 1), to the high byte only (A0 = 1, $\overline{\text{BHE}}$ = 0), or both bytes (A0 = 0, $\overline{\text{BHE}}$ = 0). If the $\overline{\text{WRH}}$ function is selected, the pin will go low if the bus cycle is writing to an odd memory location. $\overline{\text{BHE}}$/$\overline{\text{WRH}}$ is valid only during 16-bit external memory write cycles.

PIN DESCRIPTIONS (Continued)

Symbol	Name and Function
READY	Ready input to lengthen external memory cycles, for interfacing to slow or dynamic memory, or for bus sharing. When the external memory is not being used, READY has no effect.
HSI	Inputs to High Speed Input Unit. Four HSI pins are available: HSI.0, HSI.1, HSI.2, and HSI.3. Two of them (HSI.2 and HSI.3) are shared with the HSO Unit.
HSO	Outputs from High Speed Output Unit. Six HSO pins are available: HSO.0, HSO.1, HSO.2, HSI.3, HSO.4, and HSO.5. Two of them (HSO.4 and HSO.5) are shared with the HSI Unit.
Port 0	8-bit high impedance input-only port. These pins can be used as digital inputs and/or as analog inputs to the on-chip A/D converter.
Port 1	8-bit quasi-bidirectional I/O port.
Port 2	8-bit multi-functional port. All of its pins are shared with other functions in the 80C196KC.
Ports 3 and 4	8-bit bi-directional I/O ports with open drain outputs. These pins are shared with the multiplexed address/data bus which has strong internal pullups.
$\overline{\text{HOLD}}$	Bus Hold input requesting control of the bus.
$\overline{\text{HLDA}}$	Bus Hold acknowledge output indicating release of the bus.
$\overline{\text{BREQ}}$	Bus Request output activated when the bus controller has a pending external memory cycle.

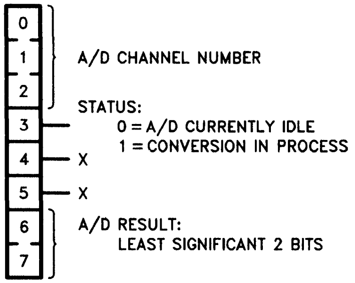
MEMORY MAP

	0FFFFH
EXTERNAL MEMORY OR I/O	
	6000H
INTERNAL ROM/EPROM OR EXTERNAL MEMORY	
	2080H
RESERVED	
	205EH
PTS VECTORS	
	2040H
UPPER INTERRUPT VECTORS	
	2030H
ROM/EPROM SECURITY KEY	
	2020H
RESERVED	
	2019H
CHIP CONFIGURATION BYTE 0	
	2018H
RESERVED	
	2014H
LOWER INTERRUPT VECTORS	
	2000H
PORT 3 AND PORT 4	
	1FFEH
EXTERNAL MEMORY	
	200H
ADDITIONAL RAM	
	100H
REGISTER FILE AND EXTERNAL PROGRAM MEMORY	
	0

	HWINDOW 0 when Read		HWINDOW 0 when Written		HWINDOW 1 Read/Write		HWINDOW 15
19H	SP (HI)	19H	SP (HI)	19H	SP (HI)	19H	SP (HI)
18H	SP (LO)	18H	SP (LO)	18H	SP (LO)	18H	SP (LO)
17H	IOS2	17H	PWM0_CONTROL	17H	PWM1_CONTROL	17H	
16H	IOS1	16H	IOC1	16H	PWM2_CONTROL	16H	
15H	IOS0	15H	IOC0	15H	RESERVED	15H	
14H	WSR	14H	WSR	14H	WSR	14H	WSR
13H	INT_MASK_1	13H	INT_MASK_1	13H	INT_MASK_1	13H	INT_MASK_1
12H	INT_PEND_1	12H	INT_PEND_1	12H	INT_PEND_1	12H	INT_PEND_1
11H	SP_STAT	11H	SP_CON	11H	RESERVED	11H	
10H	PORT2	10H	PORT2	10H	RESERVED	10H	RESERVED
0FH	PORT1	0FH	PORT1	0FH	RESERVED	0FH	RESERVED
0EH	PORT0	0EH	BAUD RATE	0EH	RESERVED	0EH	RESERVED
0DH	TIMER2 (HI)	0DH	TIMER2 (HI)	0DH	RESERVED	0DH	T2CAP (LO)
0CH	TIMER2 (LO)	0CH	TIMER2 (LO)	0CH	T2CONTROL	0CH	T2CAP (HI)
0BH	TIMER1 (HI)	0BH	IOC2	0BH	RESERVED	0BH	
0AH	TIMER1 (LO)	0AH	WATCHDOG	0AH	RESERVED	0AH	
09H	INT_PEND	09H	INT_PEND	09H	INT_PEND	09H	INT_PEND
08H	INT_MASK	08H	INT_MASK	08H	INT_MASK	08H	INT_MASK
07H	SBUF (RX)	07H	SBUF (TX)	07H	PTSSRV (HI)	07H	
06H	HSI_STATUS	06H	HSO_COMMAND	06H	PTSSRV (LO)	06H	
05H	HSI_TIME (HI)	05H	HSO_TIME (HI)	05H	PTSSEL(HI)	05H	
04H	HSI_TIME (LO)	04H	HSO_TIME (LO)	04H	PTSSEL(LO)	04H	
03H	AD_RESULT (HI)	03H	HSI_MODE	03H	A/D_TIME	03H	
02H	AD_RESULT (LO)	02H	AD_COMMAND	02H	RESERVED	02H	
01H	ZERO_REG (HI)	01H	ZERO_REG (HI)	01H	ZERO_REG (HI)	01H	ZERO_REG (HI)
00H	ZERO_REG (LO)	00H	ZERO_REG (LO)	00H	ZERO_REG (LO)	00H	ZERO_REG (LO)

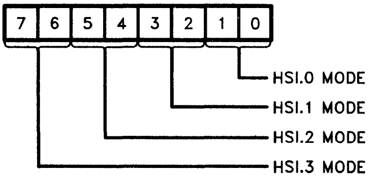
SFR BIT SUMMARY

A/D RESULT LO (02H)



270741–3

HSI_Mode (03H)

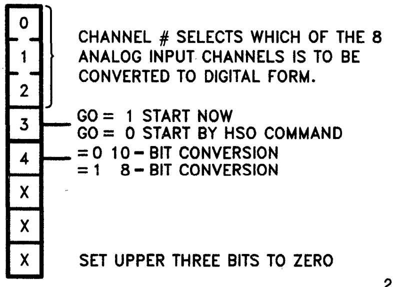


WHERE EACH 2-BIT MODE CONTROL FIELD DEFINES ONE OF 4 POSSIBLE MODES:

00	8 POSITIVE TRANSITIONS
01	EACH POSITIVE TRANSITION
10	EACH NEGATIVE TRANSITION
11	EVERY TRANSITION (POSITIVE AND NEGATIVE)

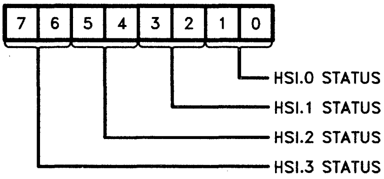
270741–6

A/D COMMAND (02H)


0
1
2
3
4
X
X
X
CHANNEL # SELECTS WHICH OF THE 8 ANALOG INPUT CHANNELS IS TO BE CONVERTED TO DIGITAL FORM.
GO = 1 START NOW
GO = 0 START BY HSO COMMAND
= 0 10-BIT CONVERSION
= 1 8-BIT CONVERSION
SET UPPER THREE BITS TO ZERO

270741–4

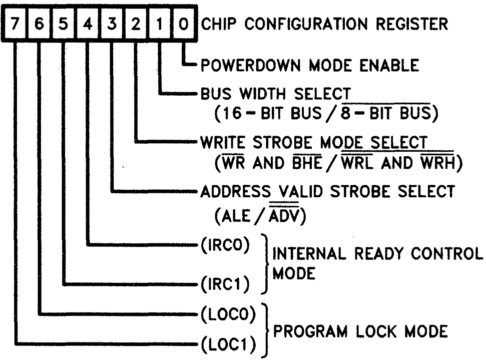
HSI_Status (06H)



WHERE FOR EACH 2-BIT STATUS FIELD THE LOWER BIT INDICATES WHETHER OR NOT AN EVENT HAS OCCURED ON THIS PIN AND THE UPPER BIT INDICATES THE CURRENT STATUS OF THE PIN.

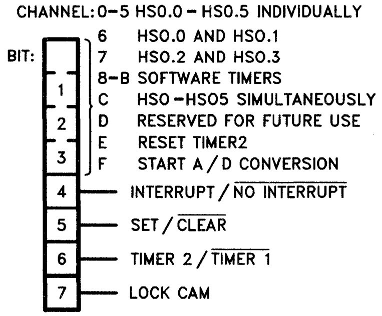
270741–7

Chip Configuration (2018H)

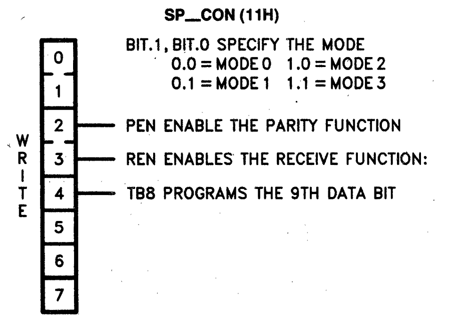

7
6
5
4
3
2
1
0
CHIP CONFIGURATION REGISTER
POWERDOWN MODE ENABLE
BUS WIDTH SELECT
(16-BIT BUS / $\overline{8\text{-BIT BUS}}$)
WRITE STROBE MODE SELECT
($\overline{WR}$ AND $\overline{BHE}$ / $\overline{WRL}$ AND $\overline{WRH}$)
ADDRESS VALID STROBE SELECT
(ALE / $\overline{ADV}$)
(IRC0)
(IRC1)
INTERNAL READY CONTROL MODE
(LOC0)
(LOC1)
PROGRAM LOCK MODE

270741–5

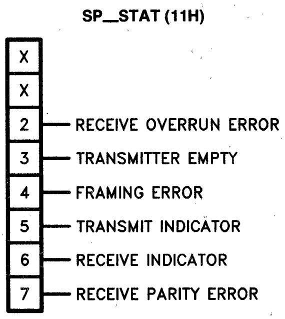
HSO COMMAND (06H)


CHANNEL: 0–5 HSO.0 – HSO.5 INDIVIDUALLY
6 HSO.0 AND HSO.1
7 HSO.2 AND HSO.3
8–B SOFTWARE TIMERS
C HSO-HSO5 SIMULTANEOUSLY
D RESERVED FOR FUTURE USE
E RESET TIMER2
F START A / D CONVERSION
BIT:
1
2
3
4
5
6
7
INTERRUPT / $\overline{\text{NO INTERRUPT}}$
SET / $\overline{\text{CLEAR}}$
TIMER 2 / $\overline{\text{TIMER 1}}$
LOCK CAM

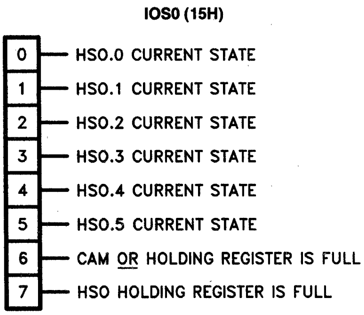
270741–8


SP_CON (11H)
0
1
2
3
4
5
6
7
WRITE
BIT.1, BIT.0 SPECIFY THE MODE
0.0 = MODE 0 1.0 = MODE 2
0.1 = MODE 1 1.1 = MODE 3
PEN ENABLE THE PARITY FUNCTION
REN ENABLES THE RECEIVE FUNCTION:
TB8 PROGRAMS THE 9TH DATA BIT

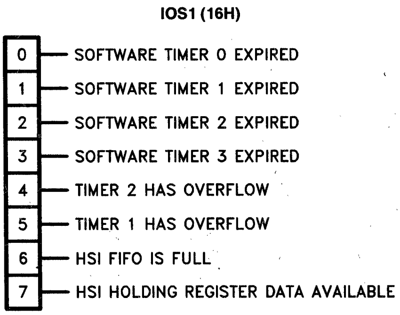
270741-9



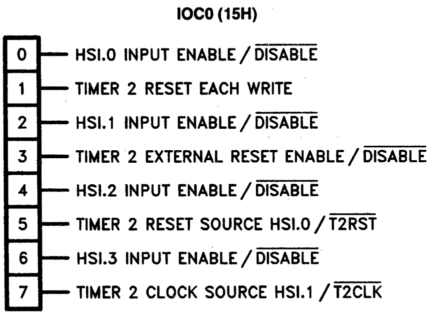
270741-30



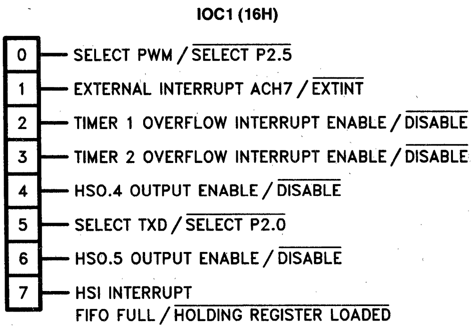
270741-10


IOS1 (16H)
0 SOFTWARE TIMER 0 EXPIRED
1 SOFTWARE TIMER 1 EXPIRED
2 SOFTWARE TIMER 2 EXPIRED
3 SOFTWARE TIMER 3 EXPIRED
4 TIMER 2 HAS OVERFLOW
5 TIMER 1 HAS OVERFLOW
6 HSI FIFO IS FULL
7 HSI HOLDING REGISTER DATA AVAILABLE

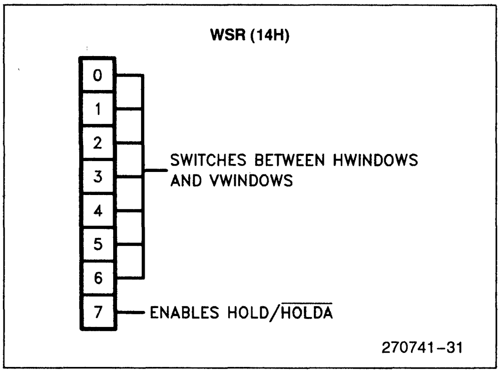
270741-12


IOC0 (15H)
0 HSI.0 INPUT ENABLE / $\overline{\text{DISABLE}}$
1 TIMER 2 RESET EACH WRITE
2 HSI.1 INPUT ENABLE / $\overline{\text{DISABLE}}$
3 TIMER 2 EXTERNAL RESET ENABLE / $\overline{\text{DISABLE}}$
4 HSI.2 INPUT ENABLE / $\overline{\text{DISABLE}}$
5 TIMER 2 RESET SOURCE HSI.0 / $\overline{\text{T2RST}}$
6 HSI.3 INPUT ENABLE / $\overline{\text{DISABLE}}$
7 TIMER 2 CLOCK SOURCE HSI.1 / $\overline{\text{T2CLK}}$

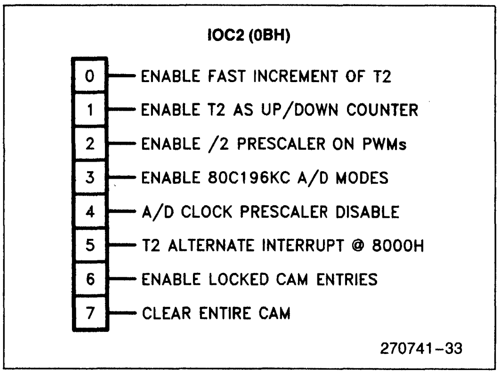
270741-11


IOC1 (16H)
0 SELECT PWM / $\overline{\text{SELECT P2.5}}$
1 EXTERNAL INTERRUPT ACH7 / $\overline{\text{EXTINT}}$
2 TIMER 1 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
3 TIMER 2 OVERFLOW INTERRUPT ENABLE / $\overline{\text{DISABLE}}$
4 HSO.4 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
5 SELECT TXD / $\overline{\text{SELECT P2.0}}$
6 HSO.5 OUTPUT ENABLE / $\overline{\text{DISABLE}}$
7 HSI INTERRUPT
FIFO FULL / $\overline{\text{HOLDING REGISTER LOADED}}$

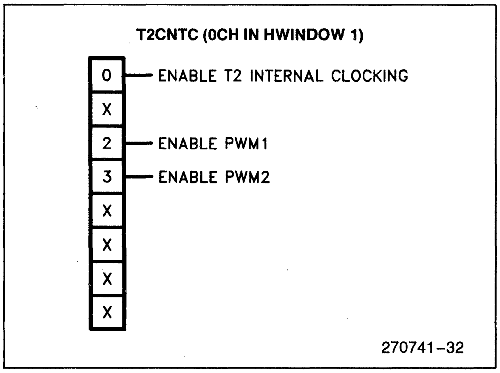
270741-13


WSR (14H)
0
1
2
3
4
5
6
7
SWITCHES BETWEEN HWINDOWS AND VWINDOWS
ENABLES HOLD/$\overline{\text{HOLDA}}$

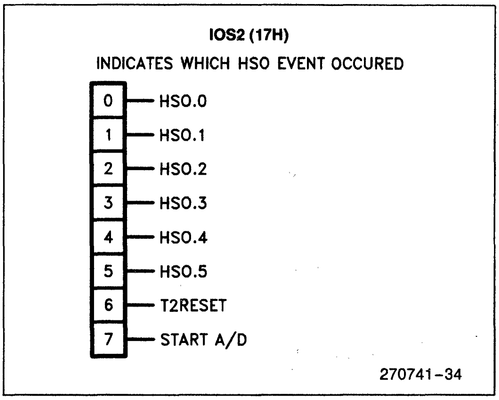
270741-31


IOC2 (0BH)
0 ENABLE FAST INCREMENT OF T2
1 ENABLE T2 AS UP/DOWN COUNTER
2 ENABLE /2 PRESCALER ON PWMs
3 ENABLE 80C196KC A/D MODES
4 A/D CLOCK PRESCALER DISABLE
5 T2 ALTERNATE INTERRUPT @ 8000H
6 ENABLE LOCKED CAM ENTRIES
7 CLEAR ENTIRE CAM

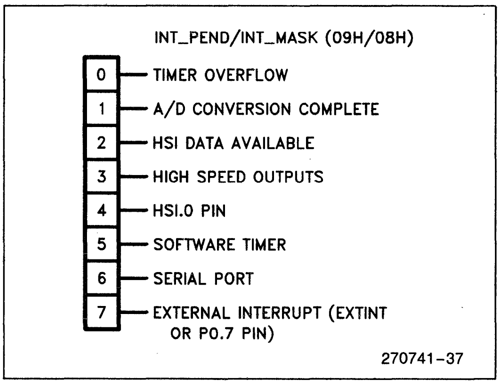
270741-33



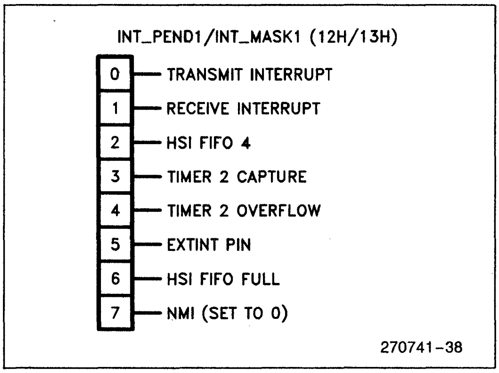
270741-32



270741-34



270741-37



270741-38

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings*

Ambient Temperature
Under Bias .... 0°C to +70°C
Storage Temperature .... −65°C to +150°C
Voltage On Any Pin to V_{SS} .... −0.5V to +7.0V
Power Dissipation 1.5W

**Notice: Stresses 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 specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

NOTICE: Specifications contained within the following tables are subject to change.

Operating Conditions

Symbol	Description	Min	Max	Units
T_A	Ambient Temperature Under Bias	0	+70	°C
V_{CC}	Digital Supply Voltage	4.50	5.50	V
V_{REF}	Analog Supply Voltage	4.50	5.50	V
f_{OSC}	Oscillator Frequency	3.5	16	MHz

NOTE:
ANGND and V_{SS} should be nominally at the same potential.

This is an Advance Data Sheet. It is expected that parameters may change before Intel releases this product for sale. Contact your local sales office before finalizing the Timing and D.C. Characteristics section of a design to verify you have the latest information.

D.C. Characteristics (Over specified operating conditions)

Symbol	Description	Min	Max	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage (Note 1)	0.2 V_{CC} + 1.0	V_{CC} + 0.5	V	
V_{IH1}	Input High Voltage on XTAL 1, EA	0.7 V_{CC}	V_{CC} + 0.5	V	
V_{IH2}	Input High Voltage on RESET	2.2	V_{CC} + 0.5	V	
V_{OL}	Output Low Voltage		0.3 0.45 1.5	V V V	I_{OL} = 200 μA I_{OL} = 2.8 mA I_{OL} = 7 mA
V_{OL1}	Output Low Voltage in RESET on P2.5 (Note 2)		0.8	V	I_{OL} = +0.4 mA
V_{OH}	Output High Voltage (Standard Outputs)	V_{CC} − 0.3 V_{CC} − 0.7 V_{CC} − 1.5		V V V	I_{OH} = −200 μA I_{OH} = −3.2 mA I_{OH} = −7 mA
V_{OH1}	Output High Voltage (Quasi-bidirectional Outputs)	V_{CC} − 0.3 V_{CC} − 0.7 V_{CC} − 1.5		V V V	I_{OH} = −10 μA I_{OH} = −30 μA I_{OH} = −60 μA
V_{OH2}	Output High Voltage in RESET on P2.0 (Note 2)	2.0		V	I_{OH} = −0.8 mA
I_{LI}	Input Leakage Current (Std. Inputs)		±10	μA	$0 < V_{IN} < V_{CC} - 0.3V$
I_{LI1}	Input Leakage Current (Port 0)		±3	μA	$0 < V_{IN} < V_{REF}$

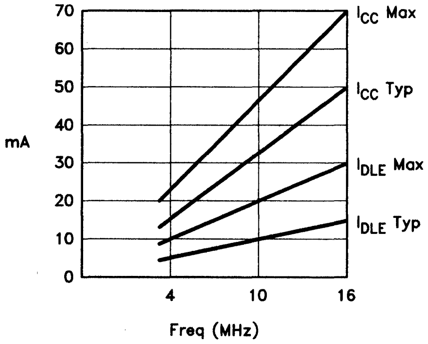
NOTE:
1. All pins except RESET, XTAL1 and EA.
2. Violating these specifications in Reset may cause the part to enter test modes.

D.C. Characteristics (Over specified operating conditions) (Continued)

Symbol	Description	Min	Typ	Max	Units	Test Conditions
I_{TL}	1 to 0 Transition Current (QBD Pins)			−650	μA	$V_{IN} = 2.0V$
I_{IL}	Logical 0 Input Current (QBD Pins)			−70	μA	$V_{IN} = 0.45V$
I_{CC}	Active Mode Current in Reset		50	70	mA	XTAL1 = 16 MHz $V_{CC} = V_{PP} = V_{REF} = 5.5V$
I_{REF}	A/D Converter Reference Current		2	5	mA	
I_{IDLE}	Idle Mode Current		15	30	mA	
I_{PD}	Powerdown Mode Current		50	T.B.D.	μA	$V_{CC} = V_{PP} = V_{REF} = 5.5V$
R_{RST}	Reset Pullup Resistor	6K		65K	Ω	$V_{CC} = 5.5V, V_{IN} = 4.0V$
C_S	Pin Capacitance (Any Pin to V_{SS})			10	pF	

NOTES:
(Notes apply to all specifications)
1. QBD (Quasi-bidirectional) pins include Port 1, P2.6 and P2.7.
2. Standard Outputs include AD0–15, $\overline{RD}$, $\overline{WR}$, ALE, $\overline{BHE}$, INST, HSO pins, PWM/P2.5, CLKOUT, RESET, Ports 3 and 4, TXD/P2.0, and RXD (in serial mode 0). The V_{OH} specification is not valid for RESET. Ports 3 and 4 are open-drain outputs.
3. Standard Inputs include HSI pins, READY, BUSWIDTH, NMI, RXD/P2.1, EXTINT/P2.2, T2CLK/P2.3, and T2RST/P2.4.
4. Maximum current per pin must be externally limited to the following values if V_{OL} is held above 0.45V or V_{OH} is held below V_{CC} − 0.7V:
   I_{OL} on Output pins: 10 mA
   I_{OH} on quasi-bidirectional pins: self limiting
   I_{OH} on Standard Output pins: 10 mA
5. Maximum current per bus pin (data and control) during normal operation is ±3.2 mA.
6. During normal (non-transient) conditions the following total current limits apply:

Port 1, P2.6	I_{OL}: 29 mA	I_{OH} is self limiting
HSO, P2.0, RXD, $\overline{RESET}$	I_{OL}: 29 mA	I_{OH}: 26 mA
P2.5, P2.7, $\overline{WR}$, $\overline{BHE}$	I_{OL}: 13 mA	I_{OH}: 11 mA
AD0–AD15	I_{OL}: 52 mA	I_{OH}: 52 mA
$\overline{RD}$, ALE, INST–CLKOUT	I_{OL}: 13 mA	I_{OH}: 13 mA



I_{CC} Max = 3.88 × FREQ + 8.43
I_{IDLE} Max = 1.65 × FREQ + 2.2

270741–17

Figure 4. I_{CC} and I_{IDLE} vs Frequency

A.C. Characteristics

For use over specified operating conditions.

Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, f_{OSC} = 16 MHz

The system must meet these specifications to work with the 80C196KC:

Symbol	Description	Min	Max	Units	Notes
T_{AVYV}	Address Valid to READY Setup		$2\,T_{OSC} - 75$	ns	
T_{LLYV}	ALE Low to READY Setup		$T_{OSC} - 70$	ns	
T_{YLYH}	Non READY Time	No upper limit		ns	
T_{CLYX}	READY Hold after CLKOUT Low	0	$T_{OSC} - 30$	ns	(Note 1)
T_{LLYX}	READY Hold after ALE Low	$T_{OSC} - 15$	$2\,T_{OSC} - 40$	ns	(Note 1)
T_{AVGV}	Address Valid to Buswidth Setup		$2\,T_{OSC} - 75$	ns	
T_{LLGV}	ALE Low to Buswidth Setup		$T_{OSC} - 60$	ns	
T_{CLGX}	Buswidth Hold after CLKOUT Low	0		ns	
T_{AVDV}	Address Valid to Input Data Valid		$3\,T_{OSC} - 55$	ns	(Note 2)
T_{RLDV}	$\overline{RD}$ Active to Input Data Valid		$T_{OSC} - 22$	ns	(Note 2)
T_{CLDV}	CLKOUT Low to Input Data Valid		$T_{OSC} - 50$	ns	
T_{RHDZ}	End of $\overline{RD}$ to Input Data Float		T_{OSC}	ns	
T_{RXDX}	Data Hold after $\overline{RD}$ Inactive	0		ns	

NOTE:
1. If max is exceeded, additional wait states will occur.
2. If wait states are used, add $2\,T_{OSC} * N$, where N = number of wait states.

A.C. Characteristics (Continued)

For user over specified operating conditions.

Test Conditions: Capacitive load on all pins = 100 pF, Rise and fall times = 10 ns, f_{OSC} = 16 MHz

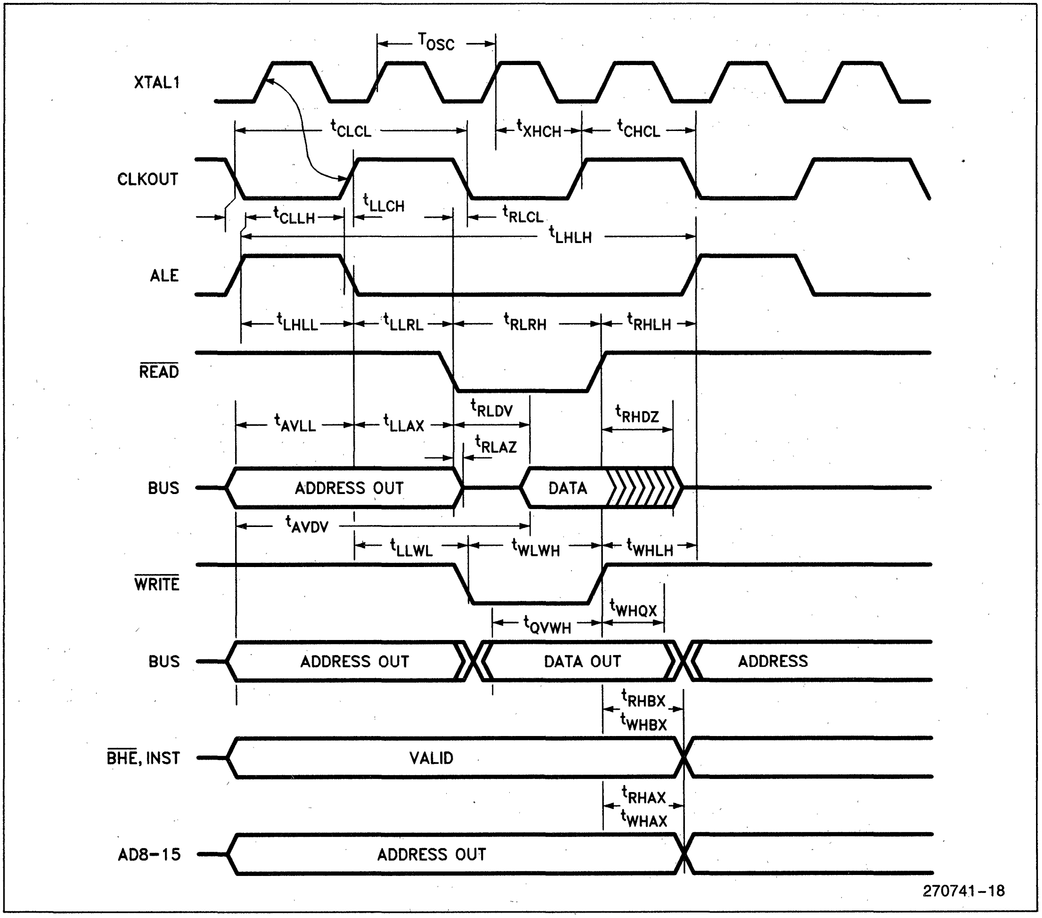
The 80C196KC will meet these specifications:

Symbol	Description	Min	Max	Units	Notes
F_{XTAL}	Frequency on $XTAL_1$	3.5	16	MHz	(Note 1)
T_{OSC}	I/F_{XTAL}	62.5	286	ns	
T_{XHCH}	XTAL1 High to CLKOUT High or Low	30	110	ns	
T_{CLCL}	CLKOUT Cycle Time	$2T_{OSC}$		ns	
T_{CHCL}	CLKOUT High Period	$T_{OSC} - 10$	$T_{OSC} + 15$	ns	
T_{CLLH}	CLKOUT Falling Edge to ALE Rising	−5	15	ns	
T_{LLCH}	ALE Falling Edge to CLKOUT Rising	−20	+15	ns	
T_{LHLH}	ALE Cycle Time	$4T_{OSC}$		ns	(Note 4)
T_{LHLL}	ALE High Period	$T_{OSC} - 10$	$T_{OSC} + 10$	ns	
T_{AVLL}	Address Setup to ALE Falling Edge	$T_{OSC} - 15$			
T_{LLAX}	Address Hold after ALE Falling Edge	$T_{OSC} - 40$		ns	
T_{LLRL}	ALE Falling Edge to $\overline{RD}$ Falling Edge	$T_{OSC} - 30$		ns	
T_{RLCL}	$\overline{RD}$ Low to CLKOUT Falling Edge	4	30	ns	
T_{RLRH}	$\overline{RD}$ Low Period	$T_{OSC} - 5$		ns	(Note 4)
T_{RHLH}	$\overline{RD}$ Rising Edge to ALE Rising Edge	T_{OSC}	$T_{OSC} + 25$	ns	(Note 2)
T_{RLAZ}	$\overline{RD}$ Low to Address Float		5	ns	
T_{LLWL}	ALE Falling Edge to $\overline{WR}$ Falling Edge	$T_{OSC} - 10$		ns	
T_{CLWL}	CLKOUT Low to $\overline{WR}$ Falling Edge	0	25	ns	
T_{QVWH}	Data Stable to $\overline{WR}$ Rising Edge	$T_{OSC} - 23$			
T_{CHWH}	CLKOUT High to $\overline{WR}$ Rising Edge	−10	15	ns	
T_{WLWH}	$\overline{WR}$ Low Period	$T_{OSC} - 30$		ns	(Note 4)
T_{WHQX}	Data Hold after $\overline{WR}$ Rising Edge	$T_{OSC} - 25$		ns	
T_{WHLH}	$\overline{WR}$ Rising Edge to ALE Rising Edge	$T_{OSC} - 10$	$T_{OSC} + 15$	ns	(Note 2)
T_{WHBX}	$\overline{BHE}$, INST after $\overline{WR}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{WHAX}	AD8–15 HOLD after $\overline{WR}$ Rising	$T_{OSC} - 30$		ns	(Note 3)
T_{RHBX}	$\overline{BHE}$, INST after $\overline{RD}$ Rising Edge	$T_{OSC} - 10$		ns	
T_{RHAX}	AD8–15 HOLD after $\overline{RD}$ Rising	$T_{OSC} - 30$		ns	(Note 3)

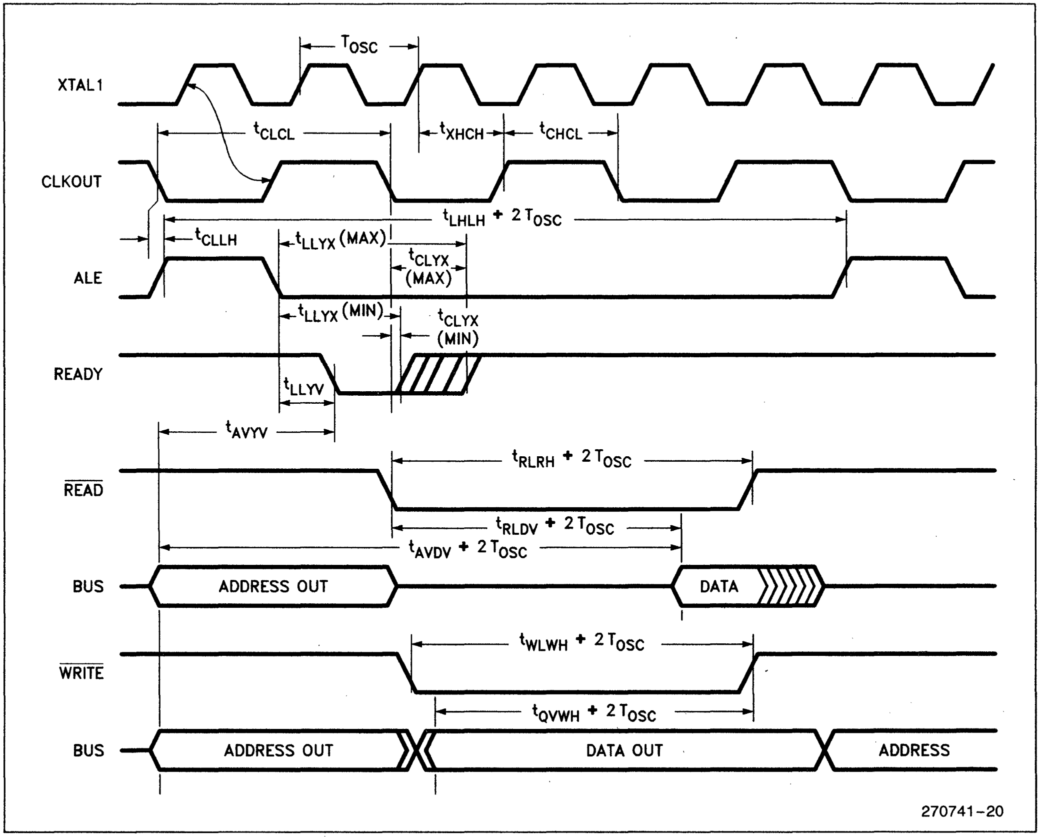
NOTES:

1. Testing performed at 3.5 MHz. However, the device is static by design and will typically operate below 1 Hz.
2. Assuming back-to-back bus cycles.
3. 8-Bit bus only.
4. If wait states are used, add 2 T_{OSC} * N, where N = number of wait states.

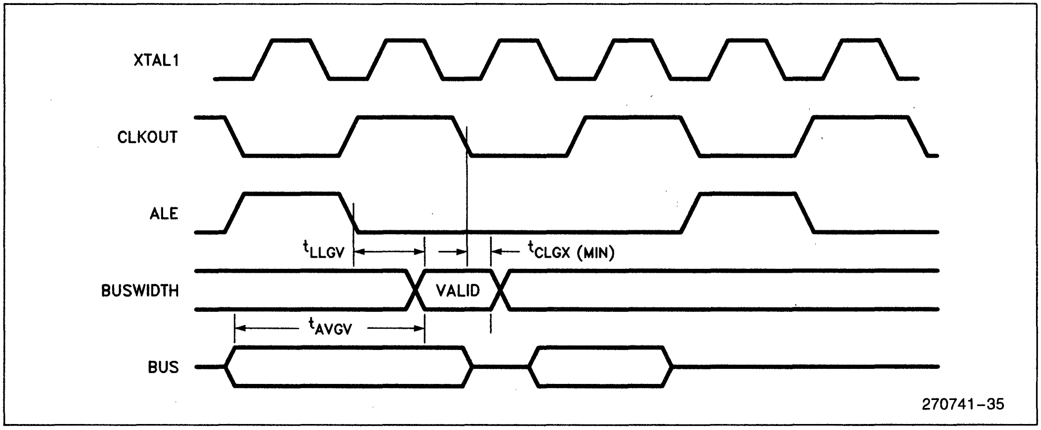
System Bus Timings



READY Timings (One Waitstate)



Buswidth Timings



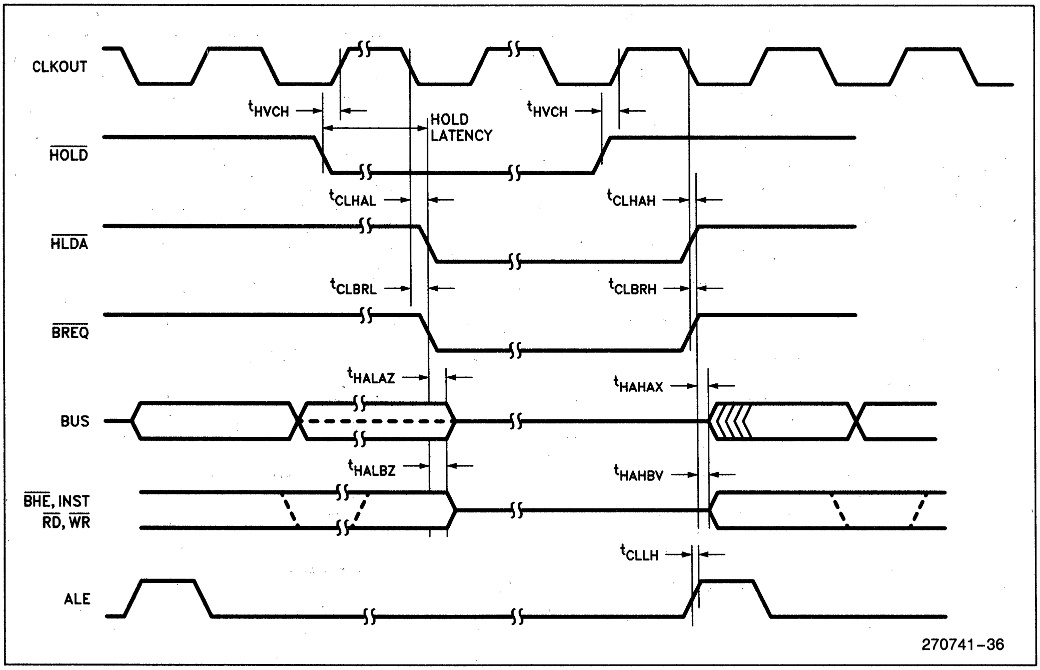
$\overline{\text{HOLD}}/\overline{\text{HLDA}}$ Timings

Symbol	Description	Min	Max	Units	Notes
T_{HVCH}	HOLD Setup	55		ns	(Note 1)
T_{CLHAL}	CLKOUT Low to HLDA Low	−15	15	ns	
T_{CLBRL}	CLKOUT Low to BREQ Low	−15	15	ns	
T_{AZHAL}	$\overline{\text{HLDA}}$ Low to Address Float		10	ns	
T_{BZHAL}	$\overline{\text{HLDA}}$ Low to $\overline{\text{BHE}}$, INST, $\overline{\text{RD}}$, $\overline{\text{WR}}$ Weakly Driven		10	ns	
T_{CLHAH}	CLKOUT Low to HLDA High	−15	15	ns	
T_{CLBRH}	CLKOUT Low to BREQ High	−15	15	ns	
T_{HAHAX}	HLDA High to Address No Longer Float	−15		ns	
T_{HAHBV}	HLDA High to BHE, INST, RD, WR Valid	−10		ns	
T_{CLLH}	CLKOUT Low to ALE High	−5	15	ns	

NOTE:
1. To guarantee recognition at next clock.

D.C. SPECIFICATIONS IN HOLD

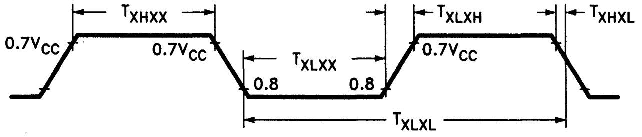
	Min	Max	Units
Weak Pullups on $\overline{\text{ADV}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WRL}}$, $\overline{\text{BHE}}$	50K	250K	$V_{CC} = 5.5V$, $V_{IN} = 0.45V$
Weak Pulldowns on ALE, INST	10K	50K	$V_{CC} = 5.5V$, $V_{IN} = 2.4$



EXTERNAL CLOCK DRIVE

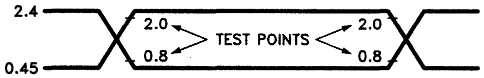
Symbol	Parameter	Min	Max	Units
$1/T_{XLXL}$	Oscillator Frequency	3.5	16.0	MHz
T_{XLXL}	Oscillator Frequency	62.5	286	ns
T_{XHXX}	High Time	22		ns
T_{XLXX}	Low Time	22		ns
T_{XLXH}	Rise Time		10	ns
T_{XHXL}	Fall Time		10	ns

EXTERNAL CLOCK DRIVE WAVEFORMS



270741-21

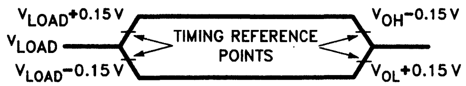
A.C. TESTING INPUT, OUTPUT WAVEFORM



270741-22

A.C. Testing inputs are driven at 2.4V for a Logic "1" and 0.45V for a Logic "0" Timing measurements are made at 2.0V for a Logic "1" and 0.8V for a Logic "0".

FLOAT WAVEFORM



270741-23

For Timing Purposes a Port Pin is no Longer Floating when a 100 mV change from Load Voltage Occurs and Begins to Float when a 100 mV change from the Loaded V_{OH}/V_{OL} Level occurs I_{OL}/I_{OH} = ±15 mA.

EXPLANATION OF AC SYMBOLS

Each symbol is two pairs of letters prefixed by "T" for time. The characters in a pair indicate a signal and its condition, respectively. Symbols represent the time between the two signal/condition points.

Conditions:

H—	High
L—	Low
V—	Valid
X—	No Longer Valid
Z—	Floating

Signals:

A—	Address	L—	ALE/$\overline{\text{ADV}}$
B—	$\overline{\text{BHE}}$	BR—	$\overline{\text{BREQ}}$
C—	CLKOUT	R—	$\overline{\text{RD}}$
D—	DATA	W—	$\overline{\text{WR}}$/$\overline{\text{WRH}}$/$\overline{\text{WRL}}$
G—	Buswidth	X—	XTAL1
H—	$\overline{\text{HOLD}}$	Y—	READY
HA—	$\overline{\text{HLDA}}$	Q—	Data Out

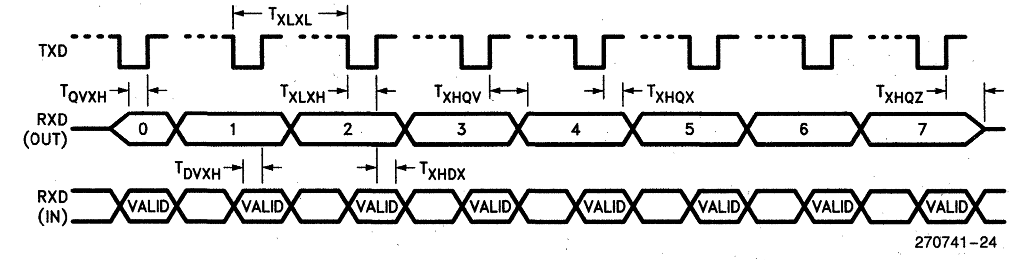
A.C. CHARACTERISTICS—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT TIMING—SHIFT REGISTER MODE

Symbol	Parameter	Min	Max	Units
T_{XLXL}	Serial Port Clock Period (BRR ≥ 8002H)	6 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR ≥ 8002H)	4 T_{OSC} ±50		ns
T_{XLXL}	Serial Port Clock Period (BRR = 8001H)	4 T_{OSC}		ns
T_{XLXH}	Serial Port Clock Falling Edge to Rising Edge (BRR = 8001H)	2 T_{OSC} ±50		ns
T_{QVXH}	Output Data Setup to Clock Rising Edge	2 T_{OSC} − 50		ns
T_{XHQX}	Output Data Hold after Clock Rising Edge	2 T_{OSC} − 50		ns
T_{XHQV}	Next Output Data Valid after Clock Rising Edge		2 T_{OSC} + 50	ns
T_{DVXH}	Input Data Setup to Clock Rising Edge	T_{OSC} +50		ns
T_{XHDX}	Input Data Hold after Clock Rising Edge	0		ns
T_{XHQZ}	Last Clock Rising to Output Float		1 T_{OSC}	ns

WAVEFORM—SERIAL PORT—SHIFT REGISTER MODE

SERIAL PORT WAVEFORM—SHIFT REGISTER MODE



EPROM SPECIFICATIONS

A.C. EPROM Programming Characteristics

Operating Conditions: Load Capacitance = 150 pF, T_A = +25°C ±5°C, V_{CC}, V_{REF} =5V, V_{SS}, ANGND = 0V, V_{PP} = 12.50V ±0.25V, EA = 12.50V ±0.25V

Symbol	Description	Min	Max	Units
T_{SHLL}	Reset High to First $\overline{\text{PALE}}$ Low	1100		Tosc
T_{LLLH}	$\overline{\text{PALE}}$ Pulse Width	50		Tosc
T_{AVLL}	Address Setup Time	0		Tosc
T_{LLAX}	Address Hold Time	100		Tosc
T_{PLDV}	$\overline{\text{PROG}}$ Low to Word Dump Valid		50	Tosc
T_{PHDX}	Word Dump Data Hold		50	Tosc
T_{DVPL}	Data Setup Time	0		Tosc
T_{PLDX}	Data Hold Time	400		Tosc
T_{PLPH}(2)	$\overline{\text{PROG}}$ Pulse Width	50		Tosc
T_{PHLL}	$\overline{\text{PROG}}$ High to Next $\overline{\text{PALE}}$ Low	220		Tosc
T_{LHPL}	$\overline{\text{PALE}}$ High to $\overline{\text{PROG}}$ Low	220		Tosc
T_{PHPL}	$\overline{\text{PROG}}$ High to Next $\overline{\text{PROG}}$ Low	220		Tosc
T_{PHIL}	$\overline{\text{PROG}}$ High to AINC Low	0		Tosc
T_{ILIH}	$\overline{\text{AINC}}$ Pulse Width	240		Tosc
T_{ILVH}	PVER Hold after $\overline{\text{AINC}}$ Low	50		Tosc
T_{ILPL}	$\overline{\text{AINC}}$ Low to $\overline{\text{PROG}}$ Low	170		Tosc
T_{PHVL}	$\overline{\text{PROG}}$ High to $\overline{\text{PVER}}$ Valid		220	Tosc

NOTES:

1. Run Time Programming is done with Fosc = 6.0 MHz to 12.0 MHz, V_{REF} = 5V ±0.50V. T_A = +25°C to ±5°C and V_{PP} = 12.50V. For run-time programming over a full operating range, contact the factory.
2. This specification is for the Word Dump Mode. For programming pulses, use 300 T_{OSC} + 100 μs.

D.C. EPROM Programming Characteristics

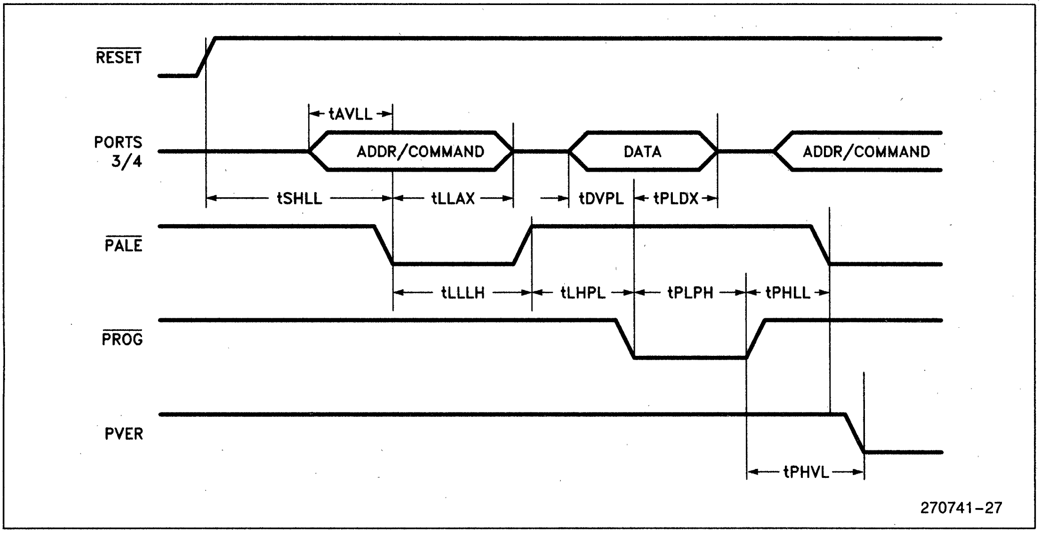
Symbol	Description	Min	Max	Units
I_{PP}	V_{PP} Supply Current (When Programming)		100	mA

NOTE:

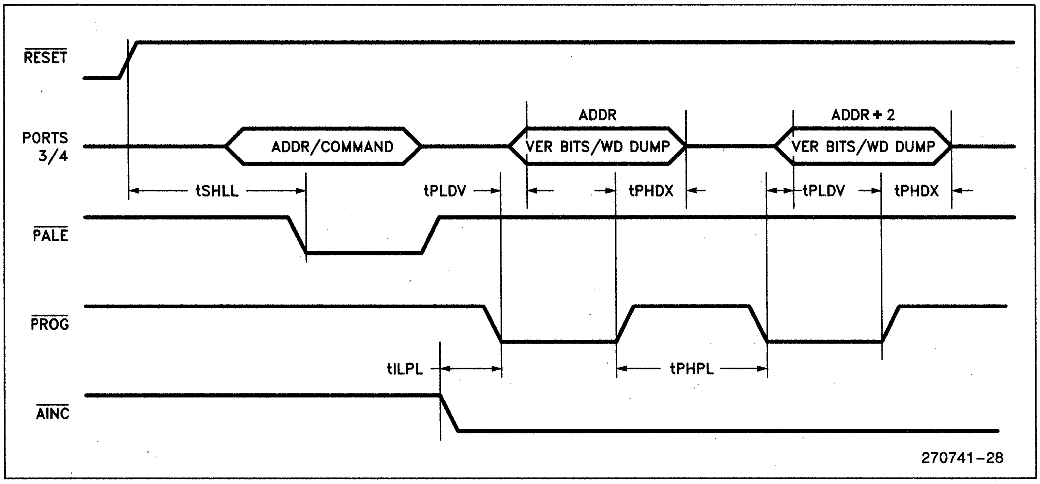
V_{PP} must be within 1V of V_{CC} while V_{CC} < 4.5V. V_{PP} must not have a low impedance path to ground of V_{SS} while V_{CC} > 4.5V.

EPROM PROGRAMMING WAVEFORMS

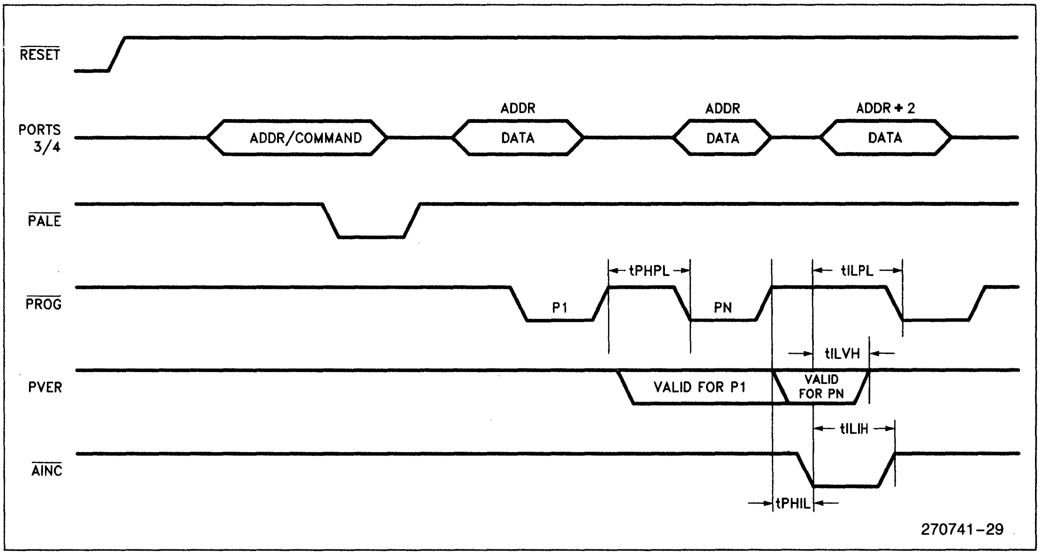
SLAVE PROGRAMMING MODE DATA PROGRAM MODE WITH SINGLE PROGRAM PULSE



SLAVE PROGRAM MODE IN WORD DUMP WITH AUTO INCREMENT



SLAVE PROGRAMMING MODE TIMING IN DATA PROGRAM WITH REPEATED PROG PULSE AND AUTO INCREMENT



10-BIT A/D CHARACTERISTICS

The speed of the A/D converter in the 10-bit mode can be adjusted by setting a clock prescaler on or off. At high frequencies more time is needed for the comparator to settle. The maximum frequency with the clock prescaler disabled is 6 MHz. The conversion times with the prescaler turned on or off is shown in the table below. The AD_TIME register has not been characterized for the 10-bit mode.

The converter is ratiometric, so the absolute accuracy is dependent on the accuracy and stability of V_{REF}. V_{REF} must be close to V_{CC} since it supplies both the resistor ladder and the digital section of the converter.

A/D CONVERTER SPECIFICATIONS

The specifications given below assume adherence to the Operating Conditions section of this data sheet. Testing is performed with $V_{REF} = 5.12V$.

Clock Prescaler On IOC2.4 = 0	Clock Prescaler Off IOC2.4 = 1
156.5 States 19.5 μs @ 16 MHz	89.5 States 29.8 μs @ 6 MHz

Parameter	Typical(3)	Minimum	Maximum	Units*	Notes
Resolution		1024 10	1024 10	Levels Bits	
Absolute Error		0	±8	LSBs	
Full Scale Error	±3			LSBs	
Zero Offset Error	±3			LSBs	
Non-Linearity		0	±8	LSBs	
Differential Non-Linearity Error		> −1	+2	LSBs	
Channel-to-Channel Matching		0	±1	LSBs	
Repeatability	±0.25			LSBs	
Temperature Coefficients: Offset Full Scale Differential Non-Linearity	 0.009 0.009 0.009			 LSB/°C LSB/°C LSB/°C	
Off Isolation		−60		dB	1, 2
Feedthrough	−60			dB	1
V_{CC} Power Supply Rejection	−60			dB	1
Input Resistance		750	1.2K	Ω	
D.C. Input Leakage		0	3.0	μA	
Sample Time: Prescaler On Prescaler Off	16 8			States States	
Input Capacitance	3			pF	

NOTES:
*An "LSB", as used here, has a value of approximately 5 mV.
1. DC to 100 KHz.
2. Multiplexer Break-Before-Make Guaranteed.
3. Typical values are expected for most devices at 25°C.

8-BIT MODE A/D CHARACTERISTICS

The 8-bit mode trades off resolution for a faster conversion time. The AD_TIME register must be used when performing an 8-bit conversion.

The following specifications are tested @ 16 MHz with OA6H in AD_TIME. The actual AD_TIME register is tested with all possible values, to ensure functionality, but the accuracy of the A/D converter is not.

Sample Time 20 States	Convert Time 56 States
A6H in AD_TIME 9.8 µs @ 16 MHz	

Parameter	Typical	Minimum	Maximum	Units*	Notes
Resolution		256 8	256 8	Levels Bits	
Absolute Error		0	±2	LSBs	
Full Scale Error	±1			LSBs	
Zero Offset Error	±2			LSBs	
Non-Linearity		0	±2	LSBs	
Differential Non-Linearity Error		>−1	+1	LSBs	
Channel-to-Channel Matching			±1	LSBs	
Repeatablity	±0.25			LSBs	
Temperature Coeffecients: Offset Full Scale Differential Non-Linearity	 0.003 0.003 0.003			 LSB/°C LSB/°C LSB/°C	

Notes:
* An "LSB", as used here, has a value of approximately 20 mV.
1. Typical values are expected for most devices at 25°C.

80C196KB TO 80C196KC DESIGN CONSIDERATIONS

1. Memory Map. The 80C196KC has 512 bytes of RAM/SFRs and 16K of ROM/EPROM. The extra 256 bytes of RAM will reside in locations 100H-1FFH and the extra 8K of ROM/EPROM will reside in locations 4000H-5FFFH. These locations are external memory on the 80C196KB.
2. The CDE pin on the KB has become a V_{SS} pin on the KC to support 16 MHz operation.
3. EPROM programming. The 80C196KC has a different programming algorithm to support 16K of onboard memory. When performing Run-Time Programming, use the section of code on page 3-91 of the 1989 16-bit Embedded Controller Handbook.
4. ONCE™ Mode Entry. The ONCE mode is entered on the 80C196KC by driving the TXD pin low on the rising edge of RESET. The TXD pin is held high by a pullup that is specified at 1.4 mA and remain at 2.0V. This Pullup must not be over-ridden or the 80C196KC will enter the ONCE mode.
5. During the bus HOLD state, the 80C196KC weakly holds RD, WR, ALE, BHE and INST in their inactive states. The 80C196KB only holds ALE in its inactive state.
6. A RESET pulse from the 80C196KC is 16 states rather than 4 states as on the 80C196KB (i.e., a watchdog timer overflow). This provides a longer RESET pulse for other devices in the system.

80C196KC ERRATA

1. A weak pullup (100K) needs to be put on either the ALE, RD or WR pin or the device may enter a factory test mode.
2. When using the A/D converter, V_{REF} must be within ±0.25V of V_{CC}.
3. Immediately after RESET is asserted, these pins are driven to their RESET state with a strong pullup or pulldown for a short period. These devices can sink or source about 2 mA. After this, the pins are only weakly held. The Pins are RD, WR, ALE, BHE, INST, Port1, P2.6, P2.7, Port3, Port4 and HSO0-3.
4. During the RESET sequence, the 80C196KC will do a fetch from location 201AH after reading the CCB from 2018H. After the fetch from 201AH, the KC begins executing from 2080H. Location 201AH is reserved by Intel and must contain OFFFH for the 80C196KC to function correctly.
5. The INST pin is weakly driven low during the CCB fetch. The weak pulldown will sink about 100 μa and remain at V_{OL}. Customers using the INST pin may have to use a 10K pulldown.
6. When reading the CCB, the 80C196KC automatically inserts 3 waitstates reregardless of the READY pin.
7. Voltage must not be applied to ACH6-7 before V_{CC} is stable or the A/D converter will latch-up.
8. Absolute maximum voltage on Port 0 is -0.5V to 6.0V relative to AV_{SS}.
9. BMOVi Instruction. The CNTREG for this instruction must be located between addresses 0-7EH. Otherwise, the CNTREG will be written back to the location of the address +100H. EX. Location 82H will be written back to location 182H.
10. The HSI unit has two errata: one dealing with resolution and the other with first entries into the FIFO.

    The HSI resolution is 9 states instead of 8 states. Events on the same line may be lost if they occur faster than once every 9 state times.

    There is a mismatch between the 9 state time HSI resolution and the 8 state time timer. This causes one time value to be unused every 9 timer counts. Events may receive a time-tag one count later than expected because of this "skipped" time value.

    If the first two events into an empty FIFO (not including the Holding Register) occur in the same internal phase, both are recorded with one time-tag. Otherwise, if the second event occurs within 9 states after the first, its time-tag is one count later than the first's. If this is the "skipped" time value, the second event's time-tag is 2 counts later than the first's.

    If the FIFO and Holding Register are empty, the first event will transfer into the Holding Register after 8 state times, leaving the FIFO empty again. If the second event occurs after this time, it will act as a new first event into an empty FIFO.
11. The serial port Framing Error flag fails to indicate an error if the bit preceding the stop bit is a 1. This is the case in both the 8-bit and 9-bit modes. False framing errors are never generated.

DATA SHEET REVISION HISTORY

The following are the important differences between the -001 and -002 versions of this data sheet. Please review this revision history carefully.

1. The 83C196KC (ROM) was added to the product line.
2. The OTP version of the EPROM was added to the product line.
3. $\overline{HOLD}/\overline{HLDA}$ Specifications were added.
4. The I_{OL} test condition on V_{OL1} has changed to -0.5 mA from -0.4 mA.
5. The I_{OH} test condition V_{OH2} has changed to 0.8 mA from 1.4 mA.
6. BMOVi errata was added.
7. Errata was added for the HSI resolution and first event anomalies.
8. Errata was added for the serial port Framing Error anomaly.

8XC196KC
16-BIT MICROCONTROLLER
EXPRESS

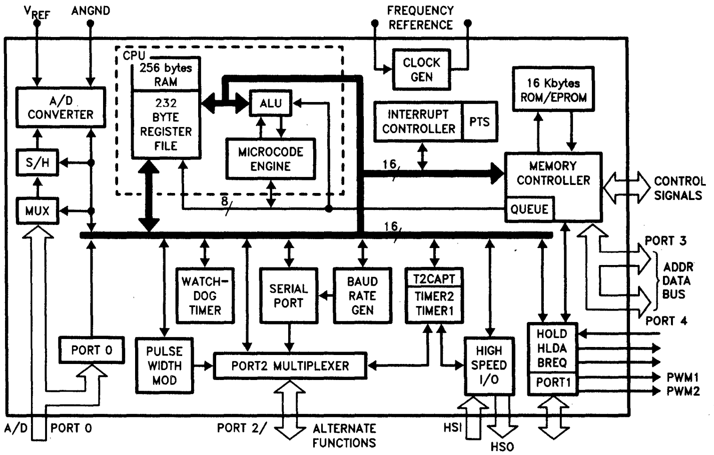
87C196KC—16 Kbytes of On-Chip EPROM
80C196KC—ROMless
83C196KC—16 Kbytes of On-Chip ROM

- **Extended Temperature Range (−40°C to +85°C)**
- **16 MHz Operation**
- **232 Byte Register File**
- **256 Bytes of Additional RAM**
- **Register-to-Register Architecture**
- **28 Interrupt Sources/16 Vectors**
- **Peripheral Transaction Server**
- **1.75 μs 16 x 16 Multiply (16 MHz)**
- **3.0 μs 32/16 Divide (16 MHz)**
- **Powerdown and Idle Modes**
- **Five 8-Bit I/O Ports**
- **16-Bit Watchdog Timer**
- **Dynamically Configurable 8-Bit or 16-Bit Buswidth**
- **Full Duplex Serial Port**
- **High Speed I/O Subsystem**
- **16-Bit Timer**
- **16-Bit Up/Down Counter with Capture**
- **3 Pulse-Width-Modulated Outputs**
- **Four 16-Bit Software Timers**
- **8- or 10-Bit A/D Converter with Sample/Hold**
- **$\overline{\text{HOLD}}$/$\overline{\text{HLDA}}$ Bus Protocol**
- **OTP One-Time Programmable Version**

With the commercial standard temperature range operational characteristics are guaranteed over the temperature range of 0°C to + 70°C. With the extended temperature range option, operational characteristics are guaranteed over the range of −40°C to +85°C.

Package types and EXPRESS versions are identified by a one- or two-letter prefix to the part number. The prefixes are listed in Table 1.

All A.C. and D.C. parameters in the commercial data sheets apply to the Express devices.



270794–1

Figure 1. 8XC196KC Block Diagram

MCS®-96 is a registered trademark of Intel Corporation.

September 1989
Order Number: 270794-001

PACKAGING

The 80C196KC is available in a 68-pin PLCC package and 87C196KC is available in a 68-pin Cerquad package. Contact your local sales office to determine the exact ordering code for the part desired.

Pin	Description	Pin	Description	Pin	Description
9	ACH7/P0.7	54	AD6/P3.6	31	P1.6/$\overline{\text{HLDA}}$
8	ACH6/P0.6	53	AD7/P3.7	30	P1.5/$\overline{\text{BREQ}}$
7	ACH2/P0.2	52	AD8/P4.0	29	HSO.1
6	ACH0/P0.0	51	AD9/P4.1	28	HSO.0
5	ACH1/P0.1	50	AD10/P4.2	27	HSO.5/HSI.3
4	ACH3/P0.3	49	AD11/P4.3	26	HSO.4/HSI.2
3	NMI	48	AD12/P4.4	25	HSI.1
2	$\overline{\text{EA}}$	47	AD13/P4.5	24	HSI.0
1	V_{CC}	46	AD14/P4.6	23	P1.4/PWM2
68	V_{SS}	45	AD15/P4.7	22	P1.3/PWM1
67	XTAL1	44	T2CLK/P2.3	21	P1.2
66	XTAL2	43	READY	20	P1.1
65	CLKOUT	42	T2RST/P2.4	19	P1.0
64	BUSWIDTH	41	$\overline{\text{BHE}}$/$\overline{\text{WRH}}$	18	TXD/P2.0
63	INST	40	$\overline{\text{WR}}$/$\overline{\text{WRL}}$	17	RXD/P2.1
62	ALE/$\overline{\text{ADV}}$	39	PWM0/P2.5	16	$\overline{\text{RESET}}$
61	$\overline{\text{RD}}$	38	P2.7/T2CAPTURE	15	EXTINT/P2.2
60	AD0/P3.0	37	V_{PP}	14	V_{SS}
59	AD1/P3.1	36	V_{SS}	13	V_{REF}
58	AD2/P3.2	35	HSO.3	12	ANGND
57	AD3/P3.3	34	HSO.2	11	ACH4/P.04
56	AD4/P3.4	33	P2.6/T2UP-DN	10	ACH5/P.05
55	AD5/P3.5	32	P1.7/$\overline{\text{HOLD}}$		

Figure 2. Pin Definitions

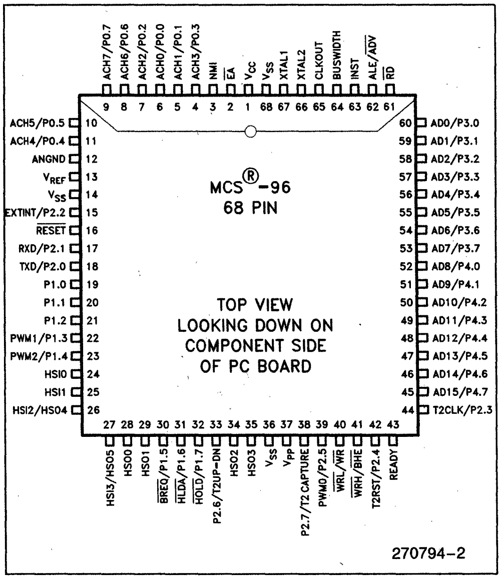


Figure 3. 68-Pin Cerquad and PLCC Package

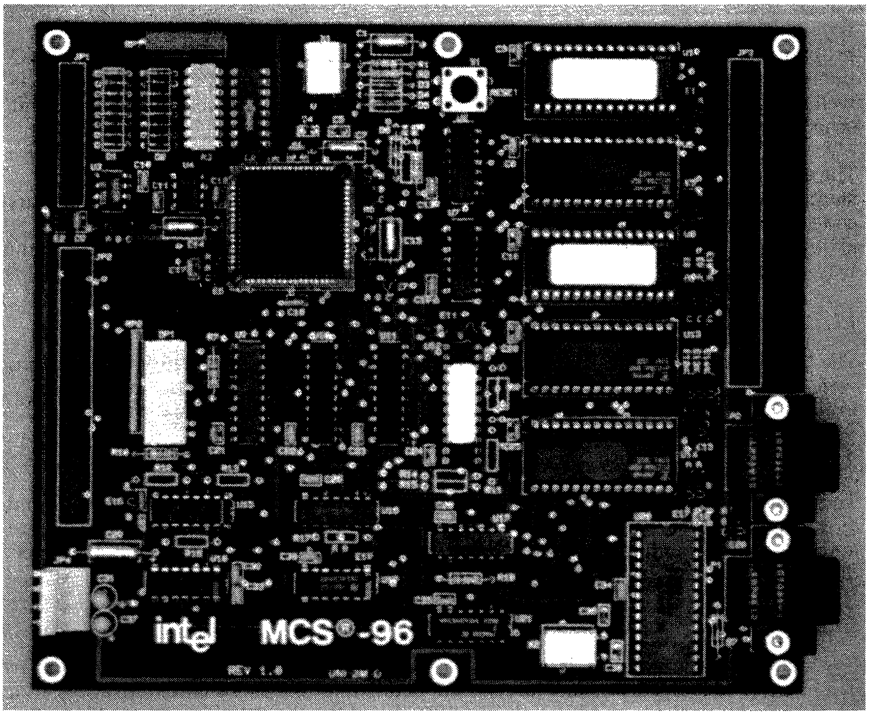
Table 1. Prefix Identification

	Cerquad	PLCC
80C196KC		TN80C196KC
83C196KC		TN83C196KC
87C196KC	TJ87C196KC	TN87C196KC*

*OTP Version

EV80C196KC EVALUATION BOARD

PRELIMINARY



EV80C196KC FEATURES

- **Zero Wait-State 16 MHz Execution Speed**
- **24K Bytes of ROMsim**
- **Flexible Wait-State, Buswidth, Chip-Select Controller**
- **Totally CMOS, Low Power Board**
- **Concurrent Interrogation of Memory and Registers**
- **Sixteen Software Breakpoints**
- **Two Single Step Modes**
- **High-Level Language Support**
- **Symbolic Debug**
- **RS-232-C Communication Link**

LOW COST CODE EVALUATION TOOL

Intel's EV80C196KC evaluation board provides a hardware environment for code execution and software debugging at a relatively low cost. The board features the 80C196KC advanced, CHMOS*, 16-bit microcontroller, the newest member of the industry standard MCS®-96 family. The board allows the user to take full advantage of the power of the MCS-96. The EV80C196KB provides zero wait-state, 16 MHz execution of a user's code. Plus, its memory (ROMsim) can be reconfigured to match the user's planned memory system, allowing for exact analysis of code execution speeds in a particular application.

*CHMOS is a patented Intel process.
**IBM PC, XT, AT and DOS are registered trademarks of International Business Machines Corporation.

intel®

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodies in an Intel product. No other circuit patent licenses are implied. Information contained herein supersedes previously published specifications on these devices from Intel.

APRIL 1989
Order number: 270802-001

Popular features such as a symbolic single line assembler/disassembler, single-step program execution, and sixteen software breakpoints are standard on the EV80C196KC. Intel provides a complete code development environment using assembler (ASM-96) as well as high-level languages such as Intel's iC-96 or PL/M-96 to accelerate development schedules.

The evaluation board is hosted on an IBM PC** or BIOS-compatible clone, already a standard development solution in most of today's engineering environments. The source code for the on-board monitor (written in ASM-96) is public domain. The program is about 1K, and can be easily modified to be included in the user's target hardware. In this way, the provided PC host software can be used throughout the development phase.

FULL SPEED EXECUTION

The EV80C196KC executes the user's code from on-board ROMsim at 16 MHz with zero wait-states. By changing crystals on the 80C196KC any slower execution speed can be evaluated. The boards host interface timing is not affected by this crystal change.

24K BYTES OF ROMSIM

The board comes with 24K bytes of SRAM to be used as ROMsim for the user's code and as data memory if needed. 16K bytes of this memory are configured as sixteen bits wide, and 8K bytes are configured as eight bits wide. The user can therefore evaluate the speed of the part executing from either buswidth.

FLEXIBLE MEMORY DECODING

By changing the Programable Logic Device (PLD) on the board, the memory on the board can be made to look like the memory system planned for the user's hardware application. The PLD controls the buswidth of the 80C196KC and the chip-select inputs on the board. It also controls the number of wait states (zero to three) generated by the 80C196KC during a memory cycle. These features can all be selected with 256 byte boundaries of resolution.

TOTALLY CMOS BOARD

The EV80C196KC board is built totally with CMOS components. Its power consumption is therefore very low, requiring 5 volts at only 300 mA. If the on board LED's are disabled, the current drops to only 165 mA. The board also requires +/- 12 volts at 15 mA.

CONCURRENT INTERROGATION OF MEMORY AND REGISTERS

The monitor for the EV80C196KC allows the user to read and modify internal registers and external memory while the user's code is running in the board.

SIXTEEN SOFTWARE BREAKPOINTS

There are sixteen breakpoints available which automatically substitute a TRAP instruction for a user's instruction at the breakpoint location. The substitution occurs when execution is started. If the code is halted or a breakpoint is reached, the user's code is restored in the ROMsim.

TWO STEP MODES

There are two single-step modes available. The first stepping mode locks out all interrupts which might occur during the step. The second mode enables interrupts, and treats subroutine calls and interrupt routines as one indivisible instruction.

HIGH LEVEL LANGUAGE SUPPORT

The host software for the EV80C196KC board is able to load absolute object code generated by ASM-96, iC-96, PL/M-96 or RL-96 all of which are available from Intel.

SYMBOLIC DEBUG

The host has a Single Line Assembler, and a Disassmbler, which recognize symbolics generated by Intel software tools.

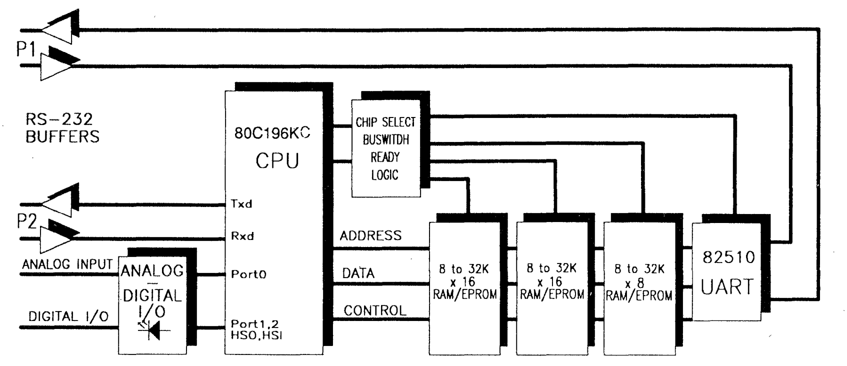
RS-232-C COMMUNICATION LINK

The EV80C196KC communicates with the host using an Intel 82510 UART provided on board. This frees the on-chip UART of the 80C196KC for the user's application.

PERSONAL COMPUTER REQUIREMENTS

The EV80C196KC Evaluation Board is hosted on an IBM PC, XT, AT** or BIOS compatible clone. The PC must meet the following minimum requirements:

- 512K Bytes of Memory
- One 360K Byte floppy Disk Drive
- PC DOS** 3.1 or Later
- A Serial Port (COM1 or COM2) at 9600 Baud
- ASM-96, iC-96 or PL/M-96
- A text editor such as AEDIT



Block Diagram of the 80C196KC Board

MCS®-96 Development Support Tools

6

ace196™ SOFTWARE

ACE196™ SOFTWARE MAKES YOU AN ARCHITECTURAL WIZARD - INSTANTLY.

If you want to learn 80C196KB architecture as fast as possible, so that you can develop hardware and software in parallel, Intel has the perfect solution.

We call it ACE196™ Software.

PC-BASED SOFTWARE TRAINING SPEEDS LEARNING

ACE196™ Software is a PC-Based Expert System that uses artificial intelligence technology and your PC's high-resolution monitor to guide you through detailed documentation training.

Its easy to use and highly graphic, designed to speed up your learning curve – and reduce your total design time, no matter what level of MCS-96 experience you have.

ACE196™ Software includes:

- A hypertext manual
- Peripheral design modules
- An assembler editor

It uses "Hypertext" to efficiently present 80C196KB documentation by providing highlighted links to related topics. You can follow these links several layers into the documentation – without having to search through hundreds of cross-referenced pages.

CONCENTRATE ON APPLICATIONS INSTEAD OF BIT-BY-BIT PROGRAMMING

After learning the basics of the architecture, you can use the ACE196™ design module to program peripherals. So, you can concentrate on application needs versus bit by bit programming materials. You'll save design time and minimize programming errors.

Also, ACE196™ Software generates fully commented initialization code and features scoreboards to document just how each peripheral has been programmed.

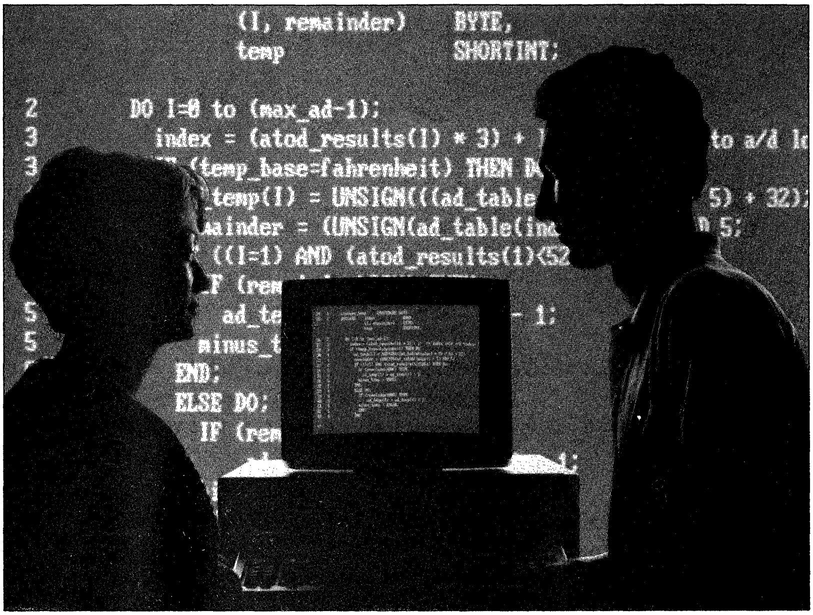
ACE196™ Software's Assembler editor makes you syntax-literate right away. It provides templates of over 100 instructions, on line help and automated register programming and testing.

System requirements: IBM compatible XT or AT*, EGA Monitor, hard disk. 1.2 meg floppy drive, 640K memory.

intel®

*IBM PC, XT, AT and DOS are registered trademarks of International Business Machines Corporation.
Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodies in an Intel product. No other circuit patent licenses are implied. Information contained herein supersedes previously published specifications on these devices from Intel.

8096/196 SOFTWARE DEVELOPMENT PACKAGES

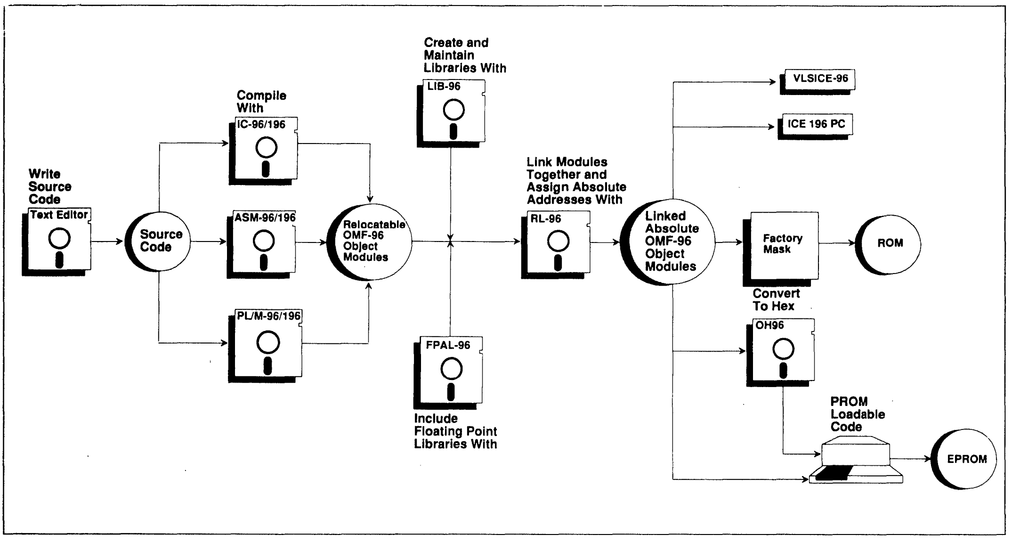

(I, remainder) BYTE,
temp SHORTINT;
2 DO I=0 to (max_ad-1);
3 index = (atod_results(I) * 3) +
3 (temp_base=fahrenheit) THEN
temp(I) = UNSIGN((ad_table
ainder = (UNSIGN(ad_table(in
((I=1) AND (atod_results(I)<52
END;
ELSE DO;
IF (rem

COMPLETE SOFTWARE DEVELOPMENT SUPPORT FOR THE 8096/196 FAMILY OF MICROCONTROLLERS

Intel supports application development for its 8096 and 80C196 family of microcontrollers with a complete set of development languages and utilities. These tools include a macroassembler, a PL/M compiler, a C compiler, linker/relocator program, floating point arithmetic library, a librarian utility, and an object-to-hex utility. Develop code in the language(s) you desire, then combine object modules from different languages into a single, fast program.

FEATURES

- Software Tools support all members of Intel's MCS®-96 family
- ASM-96/196 macroassembler for speed critical code
- PL/M-96/196 package for the maintainability and reliability of a high-level language with support for many low-level hardware functions
- iC-96/196 package for structured C language programming, with many hardware specific extensions
- Linker/Relocator program for linking modules generated in assembler, PL/M or C and assigning absolute addresses to relocatable code. RL-96 prepares your code for execution in target with a simple, one-step operation
- 32-bit Floating Point Arithmetic Library to reduce your development effort and to allow fast, highly optimized numerics-intensive processing
- Library utility for creating and maintaining software object module libraries
- PROM building utility that converts object modules into standard hexadecimal format for easy download into a non-Intel PROM Programmer
- Hosted on IBM PC XT/AT with PC-DOS 3.0 or above



Write Source Code
Text Editor
Source Code
Compile With
iC-96/196
ASM-96/196
PL/M-96/196
Relocatable OMF-96 Object Modules
Create and Maintain Libraries With
LIB-96
FPAL-96
Include Floating Point Libraries With
Link Modules Together and Assign Absolute Addresses With
RL-96
Linked Absolute OMF-96 Object Modules
VLSICE-96
ICE 196 PC
Factory Mask
ROM
Convert To Hex
OH96
PROM Loadable Code
EPROM

Figure 1. MCS®-96 Application Development Process

ASM-96/196 MACROASSEMBLER

ASM-96/196 is the macroassembler for the MCS-96 family of microcontrollers, including the 80C196. ASM-96/196 translates symbolic assembly language mnemonics into relocatable object code.

The macro facility in ASM-96/196 saves development and maintenance time, since common code sequences need only be developed once. The assembler also supports symbolic access to the many features of the 8096/196 and provides an "include" file with all 8096/196 registers defined.

PL/M-96/196 SOFTWARE PACKAGE

PL/M-96/196 is a high-level programming language designed to support the software requirements of advanced 16-bit microcontrollers. The PL/M-96/196 compiler translates PL/M high-level language statements into 8096/196 relocatable object code. Major features of the PL/M-96/196 compiler include:

- **Structured programming.** The PL/M language supports modular and structured programming, making programs easier to understand, maintain, and debug.
- **Built-in functions.** PL/M-96/196 includes an extensive list of functions, including TYPE CONVERSION functions, STRING manipulations, and functions for interrogating MCS-96 hardware flags.
- **!nterrupt handling.** The INTERRUPT attribute allows you to define interrupt handling procedures. The compiler generates code to save and restore the program status word for INTERRUPT procedures.
- **Compiler controls.** Compile-time options increase the flexibility of the PL/M-96/196 compiler. These controls include: optimization, conditional compilation, the inclusion of common PL/M source files from disk, cross-reference of symbols, and optional assembly language code in the listing file.
- **Data types.** PL/M-96/196 supports seven data types, allowing PL/M-96/196 to perform three different kinds of arithmetic: signed, unsigned, and floating point.
- **Language compatibility.** PL/M-96/196 object modules are compatible with all other object modules generated by Intel MCS-96 translators.

iC-96/196 SOFTWARE PACKAGE

Intel's iC-96/196 is a structured programming language designed to support applications for the 16-bit family of MCS-96 microcontrollers. iC-96/196 implements the C language as described in the Kernighan and Ritchie book, *The C Programming Language*, and includes many of the enhancements as defined by the proposed ANSI C standard. Major features of the iC-96/196 compiler include:

- **Symbolics.** The iC-96/196 compiler boosts programmer productivity by providing extensive debug information, including symbols. The debug information can be used to debug the code using either the VLSiCE™-96 emulator or the ICE™-196PC emulator.
- **Architecture Support.** iC-96/196 generates code which is fully optimized for the MCS-96 architecture. iC-96/196 provides an INTERRUPT attribute, allowing you to define interrupt handling functions in C, and library routines which allow you to enable and disable interrupts directly from C (mid-1989). A REENTRANT/NOREENTRANT control is also included, allowing the compiler to identify non-reentrant procedures. This gives you full access to the large MCS-96 register set.
- **Standard language.** iC-96/196 accepts standard C source code. iC-96/196 code is fully linkable with both PL/M-96/196 and ASM-96/196 modules via an "alien" attribute, allowing programmers to utilize the optimal language for any application. In addition, programmers can quickly begin programming with iC-96/196 because it conforms to accepted C language standards.

RL-96/196 LINKER/RELOCATOR

Intel's RL-196 utility is used to link multiple MCS-96 object modules into a single program and then assign absolute addresses to all relocatable addresses in the new program. Modules can be written in ASM-96/196, PL/M-96/196, or iC-96/196.

The RL-96/196 utility also promotes programmer productivity by encouraging modular programming. Because applications can be broken into separate modules, they're easier to design, test and maintain. Standard modules can be reused in different applications, saving software development time.

FPAL-96/196 FLOATING POINT ARITHMETIC LIBRARY

FPAL-96/196 is a library of single-precision 32-bit floating point arithmetic functions. These functions are compatible with the IEEE floating point standard for accuracy and reliability and include an error-handler library.

LIB-96/196

The Intel LIB-96/196 utility creates and maintains libraries of software object modules. Standard modules can be placed in a library, and linked into your applications programs using RL-96/196.

OH-96/196

The OH-96/196 utility converts Intel OMF-96 object modules into standard hexadecimal format. This allows the code to be loaded directly into a PROM via non-Intel PROM programmers.

SERVICE, SUPPORT, AND TRAINING

Intel augments its MCS-96 architecture family development tools with a full array of seminars, classes, and workshops; on-site consulting services; field application engineering expertise; telephone hot-line support; and software and hardware maintenance contracts. This full line of services will ensure your design success.

ORDERING INFORMATION

D86ASM96NL*	96/196 Assembler for PC XT or AT system (or compatible), running DOS 3.0 or higher
D86PLM96NL*	PL/M-96/196 Software Package for PC XT or AT system (or compatible), running DOS 3.0 or higher
D86C96NL*	iC-96/196 Software Package for PC XT or AT system (or compatible), running DOS 3.0 or higher

*Also Includes: Relocator/Linker, Object-to-hex converter, Floating Point Arithmetic Library, and Librarian.

ORDERING INFORMATION

D86ASM96* 96/196 Assembler for PC XT or AT system (or compatible), running DOS 3.0 or higher

D86PLM96* PL/M-96/196 Software Package for PC XT or AT system (or compatible), running DOS 3.0 or higher

D86C96* iC-96/196 Software Package for PC XT or AT system (or compatible), running DOS 3.0 or higher

*Also Includes: Relocator/Linker, Object-to-hex converter, Floating Point Arithmetic Library, and Librarian.

For direct information on Intel's Development Tools, or for the number of your nearest sales office or distributor, call 800-874-6835 (U.S.). For information or literature on additional Intel products, call 800-548-4725 (U.S. and Canada).

VLSiCE™-96 IN-CIRCUIT EMULATOR

IN-CIRCUIT EMULATOR FOR THE 8X9X FAMILY OF MICROCONTROLLERS

The VLSiCE™-96 emulator is a complete hardware/software debug environment for developing systems based on the Intel 8x9x family of microcontrollers. The VLSiCE-96 emulator supports all NMOS members of Intel's MCS-96 microcontrollers, including the 8096BH, the 8098, the 8095, the 8097, and the 8096-90. With high performance 12 MHz emulation, symbolic debugging, and flexible memory mapping, the VLSiCE-96 emulator expedites all stages of development: software development, hardware development, system integration and system test.

FEATURES

- Real-time transparent emulation, up to 12 MHz
- 64K of mappable memory to allow early software debug and (EP)ROM simulation, even before any target hardware is available.
- Trace contains execution address, opcode, symbolics, and bus information
- 4K frame trace buffer for storing real-time execution history
- Ability to break or trace on execution addresses, opcodes, data values, or flags values
- Symbolic debugging for faster and easier access to memory location and program variables.
- Fast breaks and dynamic trace to allow the user to modify and interrogate memory, and access the trace buffer without stopping emulation.
- On-line Help file to speed development
- Shadow Registers can read many write-only registers and write to many read-only registers, allowing enhanced debugging over component features
- Includes 68-pin PGA adaptor; optional 68-pin PLCC and 48-pin DIP adaptors are also available
- Serially hosted on IBM PC AT/XT or compatibles with DOS 3.0 or greater.

October, 1989
Order Number: 280794-001

ONE TOOL FOR ENTIRE DEVELOPMENT CYCLE

The VLSiCE-96 emulator speeds target system development by allowing hardware and software design to proceed simultaneously. You can develop software even before prototype hardware is finished. And because the VLSiCE-96 emulator precisely matches the component's electrical and timing characteristics, it's a valuable tool for hardware development and debug.

The VLSiCE-96 emulator also simplifies and expedites system integration and test. As each section of the hardware is completed, it is simply added to the prototype and tested in real-time. When the prototype is complete, it is tested with the final version of the system software. The VLSiCE-96 emulator can then be used to verify or debug the target system as a completed unit.

Because it supports the ROMless, ROM and EPROM versions of Intel's microcontrollers, the VLSiCE-96 emulator can debug a prototype or production product at any stage in its development without introducing extraneous hardware or software test tools.

SPECIFICATIONS

HOST REQUIREMENTS

An IBM PC AT/XT or compatible with 512K bytes RAM and hard disk. Intel recommends an IBM PC AT or compatible with 640K bytes of RAM, one floppy drive and one hard disk running PC-DOS 3.1 or later.

System Performance

Mappable zero wait state (up to 12 MHz) Min 0K bytes, Max 64K bytes	Mappable to user memory or ICE memory in 1K blocks on 1K boundaries
Trace Buffer	4K bytes × 48 bits
Virtual Symbol Table	A maximum of 61K bytes of host memory space is available for the virtual symbol table (VST). The rest of the VST resides on disk and is paged in and out of host memory as needed.

Electrical Characteristics

Power Supply
100V-120V or 200V-240V (selectable)
50 Hz-60 Hz
2 amps (AC max) @ 120V
1 amp (AC max) @ 240V

Physical Characteristics

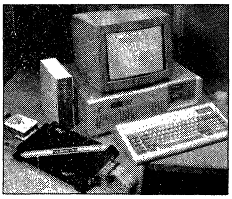
Controller Pod

Width:	8¼″ (21 cm)
Height:	1½″ (4 cm)
Depth:	13½″ (34 cm)
Weight:	4 lbs (2 kg)

Power Supply

Width:	7⅝″ (18 cm)
Height:	4″ (10 cm)
Depth:	11″ (28 cm)
Weight:	15 lbs (7 kg)

User Cable:	3″ (1 m)



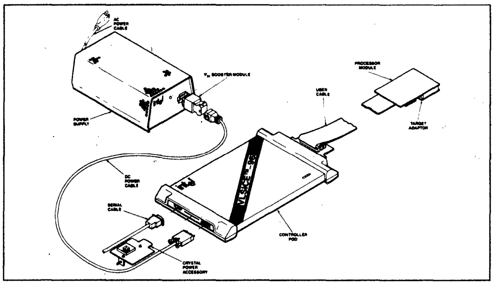


FIGURE 1. The VLSiCE-96™ Emulator

Environmental Characteristics

Operating Temperature:	0°C to +40°C (−32°F to + 104°F)
Operating Humidity:	Maximum to 85% relative humidity, non condensing

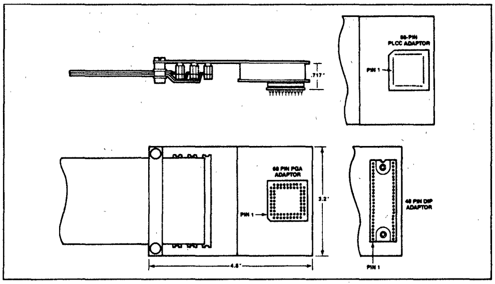


FIGURE 2. Dimensions for the Emulator Processor Board and Adaptors

SERVICE SUPPORT AND TRAINING

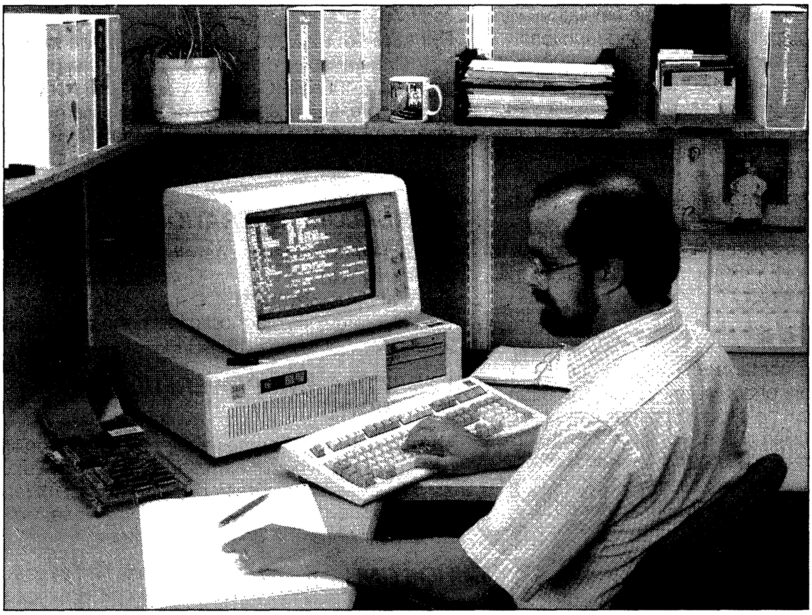
Intel augments its MCS-96 architecture family development tools with a full array of seminars, classes and workshops; on-site consulting services; field application engineering expertise; telephone hot-line support; and software and hardware maintenance contracts. This full line of services will ensure your design success.

ORDERING INFORMATION

V096-KITA	VLSiCE-96 Power supply cable, emulation base, user cable, Crystal Power Accessory (CPA), serial cables for PC AT/XT, a 68-pin PGA target adaptor, ASM-96, AEDIT Text Editor. Host, probe, diagnostic and tutorial software on 5¼″ media for DOS hosts running DOS V3.0 or later. (Requires software license.)
V096KITD	Same as V096KITA without ASM-96 and AEDIT text editor.
TA096E	Optional 68-pin PLCC Target Adaptor Board
TA096B	Optional 48-pin DIP Target Adaptor Board
MSA96	Optional Multi-Synchronous Accessory for multi-ICE capability
SA096D	Software for host, probe, diagnostic and tutorial on 5¼″ media for use with the PC AT/XT under PC-DOS V3.0 or later. (Requires software license.) (Included with V096KITA and V096KITD.)
D86C96NL	C-96 Compiler*
D86PLM96NL	PL/M-96 Compiler*
D86ASM96NL	ASM-96 Macroassembler*

*Also Includes: Relocator/Linker, object-to-hex converter, librarian, and Floating Point Arithmetic Library.

REAL-TIME TRANSPARENT 80C196 IN-CIRCUIT EMULATOR

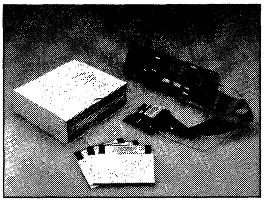


REAL-TIME TRANSPARENT 80C196 IN-CIRCUIT EMULATOR

The ICE™-196KB/PC in-circuit emulator delivers real-time high-level debugging capabilities for developing, integrating and testing 80C196-based designs. Operating at the full speed of the 80C196KB microcontroller, the ICE-196KB/PC provides precise I/O pin timings and functionality. The ICE-196KB/PC also allows you to develop code before prototype hardware is available. The in-circuit emulator represents a low-cost development environment for designing real-time microcontroller-based applications with minimal investment in time and resources.

ICE™-196KB/PC IN CIRCUIT EMULATOR FEATURES

- Real-Time Emulation of the 80C196KB Microcontroller
- 64K Bytes of Mappable Memory
- 2K-entry Trace Buffer
- 3 Breakpoints or 1 Range Break
- Symbolic Support and Source Code Display
- Standalone Operation
- Versatile and Powerful Host Software
- Hosted On IBM PC, XT, AT, or Compatibles With DOS 3.x



September, 1989
Order Number: 280727-004

REAL-TIME EMULATION

The ICE-196KB/PC provides real-time emulation with the precise input/output pin timings and functions across the full operating frequencies of the 80C196KB microcontroller. The ICE-196KB/PC connects to the intended 80C196KB microcontroller socket via a 16″ flex cable, which terminates in a 68-pin PLCC probe.

MAPPABLE MEMORY

The ICE-196KB/PC has 64K bytes of zero wait-state memory that can be enabled or mapped as read-only, write-only or read/write in 4K byte increments to simulate the internal (EP)ROM of the 80C196KB or external program memory.

TRACE BUFFER

The ICE-196KB/PC contains a 2K entry trace buffer for keeping a history of actual instruction execution. The trace buffer can be conditionally turned off to collect a user specified number of trace frames. Trace information can be displayed as disassembled instructions or, optionally, disassembled instructions and the original iC-96 and PL/M-96 source code.

BREAK SPECIFICATION

Three execution address breakpoints or one range of addresses can be active at any time. The ICE-196KB/PC allows any number of breakpoints to be defined and activated when needed.

SYMBOLIC SUPPORT AND SOURCE CODE DISPLAY

Full ASM-96, PL/M-96 and iC-96 language symbolics, including variable typing and scope, are supported by the ICE-196KB/PC memory accesses, trace buffer display, breakpoint specification, and assembler/disassembler. Additionally, iC-96 and PL/M-96 source code can be displayed to make development and debug easier.

STANDALONE OPERATION

Product software can be developed prior to hardware availability with the optional Crystal Power Accessory (CPA) and the ICE-196KB/PC mappable memory. The CPA also provides diagnostic testing to assure full functionality of the ICE-196KB/PC.

VERSATILE AND POWERFUL HOST SOFTWARE

The ICE-196KB/PC comes equipped with an on-line help facility, a dynamic command entry and syntax guide, built-in editor, assembler and disassembler, and the ability to customize the command set via literal definitions and debug procedures.

HOSTING

The ICE-196KB/PC is hosted on the IBM PC XT, AT or compatibles with PC-DOS 3.x.

SPECIFICATIONS

REQUIREMENTS

Host

IBM PC XT, AT (or compatible)
512K bytes RAM, Hard Disk
PC-DOS 3.x
One Unused Peripheral Slot
DC Current 2.5A
Note: ICE-196KB/PC uses two bytes of the user stack.

TARGET INTERFACE BOARD

Length	2.0″ (5.1cm)
Height	1.2″ (3.0cm)
Width	2.3″ (5.8cm)

USER CABLE

Length	15.6″ (39.6cm)

PROBE ELECTRICAL

80C196KB plus per pin	50pf loading 5ns propagation delay
Icc (from target system)	50mA @ 12 MHz
Operating Frequency	3.5 to 12 MHz, 12 MHz only with CPA

ENVIRONMENTAL CHARACTERISTICS

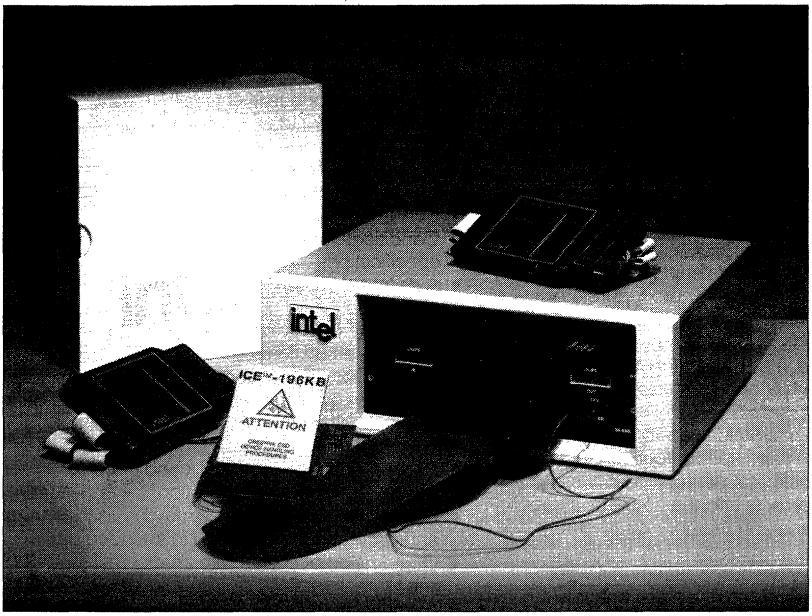
Operating Temperature	10°C to 40°C 37.5°F to 104°F
Operating Humidity	Maximum 55% Relative Humidity, non-condensing

ORDERING INFORMATION

Order Code	Description
ICE196KBPC	Emulation Board, user cable, target interface board (PLCC), host, diagnostic, and tutorial software on 5¼″ DOS diskette, and Crystal Power Accessory with power cable
ICE196KBPCB	Same as above except does not include Crystal Power Accessory
CPA196KAKB	Crystal Power Accessory and power cable only
TA196PLCC68PGA	68-Pin PGA Target Adaptor

For direct information on Intel's Development Tools, or for the number of your nearest sales office or distributor, call 800-874-6835 (U.S.). For information or literature on additional Intel products, call 800-548-4725 (U.S. and Canada).

ICE™-196KB/HX IN-CIRCUIT EMULATOR



MODULAR IN-CIRCUIT EMULATOR FOR THE 8xC196KB FAMILY OF MICROCONTROLLERS

The ICE™-196KB/HX in-circuit emulator delivers a complete, real-time, hardware/software debug environment for developing, integrating, and testing 8xC196KB-based designs. The ICE-196KB/HX emulator is a high-performance modular debugging system featuring real-time and transparent 12 MHz emulation, high-level symbolic debugging, complete execution and bus break/trace capabilities, 128k bytes zero-waitstate mappable memory, and emulation trace. The ICE-196KB/MX emulator, a companion entry-level system, is also available. The ICE-196KB/MX emulator can be upgraded to an ICE-196KB/HX emulator with optional add-in boards. Both systems feature an identical human interface, utilize the same base chassis, and are serially hosted on IBM PC XTs, ATs, and 100% compatibles.

ICE-196KB/xX IN-CIRCUIT EMULATORS CORE FEATURES

- **Precisely** matches the component's electrical and timing characteristics
- Supports the ROMless and (EP)ROM versions of the 8xC196KB
- Does not introduce extraneous hardware or software overhead
- Modular base for future growth and migration

ICE™-196KB/MX IN-CIRCUIT EMULATOR FEATURES

- Real-time transparent 8xC196KB emulation to 12 MHz, including ROM and EPROM versions, either in-target or standalone
- Use of either target system or 64k bytes of zero-waitstate emulator memory for program execution
- Event recognition of up to 255 execution address specifications, either specific or ranges
- Fastbreaks, wherein emulation is immediately broken only for the duration of a requested memory access
- Single-step execution of machine instructions, high-level language statements, or procedure call blocks
- Execution trace, 2k frames deep, including address, opcode in hex and mnemonic formats, and operands in hex and symbolic formats
- Functions to disassemble/assemble memory in the form of machine instructions and to display/modify program variables and special function registers
- Symbolic referencing to memory locations and microcontroller objects and the output of symbolic information in trace and memory disassembly displays
- Automatic display of source statements when memory is disassembled, the trace buffer is displayed, or emulation is broken
- Automatic update of selected variables displayed in a Watch Window
- Dynamic display of the trace buffer during emulation
- A command line user interface with context-sensitive help in pop-up windows
- User-definable function keys and procedures with variables and literal definitions
- On-circuit emulation of surface mount components
- An emulation timer returning the time from entering until leaving emulation
- Synchronized multi-emulator start and break and a trigger out for synchronization with external logic analyzer or other device

ICE™-196KB/HX IN-CIRCUIT EMULATOR FEATURES

Includes all features of the ICE™-196KB/MX emulator *plus the following:*

- Recognition of bus events (either instruction fetch, data read or write at a specific address or range of addresses, or a specific value or range of values)
- OR combinations of execution/bus events or strictly bus events, plus AND combinations of bus events
- Event counters
- Conditional arming and disarming of break specifications
- A deferred Fastbreak option whereby emulation is broken only after reaching a specified address
- Additional 64k bytes (128k bytes total) of zero-waitstate emulation memory
- The addition of bus address/data, processor status bits, and logic clips information in the trace buffer
- Conditional arming and disarming of trace specifications
- Reprogrammability of break and trace specifications during emulation
- Eight logic clips input lines may be used to trigger an action and are captured in the trace buffer
- Qualification of events with an external input SYSIN line
- Asynchronous break based on a signal from an external device
- Eight logic clips output lines are settable to simulate a condition in the target system
- SYSOUT output may be used to stimulate an action in the target system based on a recognized event
- An event timer calculating time between specified conditions

FEATURE COMPARISON OF INTEL'S 8xC196KB IN-CIRCUIT EMULATORS

	ICE-196KB/PC	*ICE-196KB/MX*	*ICE-196KB/HX*
Real-time, transparent	Yes	Yes	Yes
Hosting	PC XT Bus	PC XT, AT Serial	PC XT, AT Serial
Expandable/Upgradable	No	Yes	Yes
Mapped Memory (bytes)	64k	64k	128k
Trace Buffer (frames)	2k	2k	2k
Execution Breaks	Yes	Yes	Yes
Breakpoints	3	255	255
Fastbreaks	No	Abrupt only	Yes
Bus Break/Trace	No	No	Yes
Complex Break/Trace Events	No	No	Yes
Reprogrammable Break/Trace	No	No	Yes
Symbolic Debug	Yes	Yes	Yes
Source Code Display	Yes	Yes	Yes
Watch Windows	Yes	Yes	Yes
Dynamic Trace Display	No	Yes	Yes
Emulation Timer	No	Yes	Yes
Event Timer	No	No	Yes
Logic Analysis Clips	No	No	Yes
Multi-ICE Support	No	Yes	Enhanced

COMPLETE FAMILY OF 8x196 DEVELOPMENT TOOLS

ICE-196KB/MX and ICE-196KB/HX emulators are complemented by Intel's low-cost ICE-196KB/PC emulator. All three emulators utilize an upward-compatible human interface to preserve your learning investment and to allow multiple emulators for large design teams. Each emulator has been designed to work in conjunction with Intel's MCS-96 software tools, including a macro assembler, a PL/M-96 compiler, an iC-96 compiler, and various utilities.

Optional boards are available to upgrade an ICE-196KB/MX emulator with some or all of the functionality of an ICE-196KB/HX emulator. In addition, the ICE-196KB/MX and ICE-196KB/HX emulators have been designed to support future proliferations within the 8xC196 family of microcontrollers.

WORLDWIDE SERVICE AND SUPPORT

Intel augments its MCS-96 architecture family development tools with a full array of seminars, classes, and workshops; on-site consulting services; field application engineering expertise; telephone hot-line support; and software and hardware maintenance contracts. This full line of services will ensure your design success.

SPECIFICATIONS

HOST REQUIREMENTS

Emulators require an IBM PC AT, XT, or 100% compatible with 512k bytes RAM and hard disk running DOS 3.x.

ELECTRICAL CHARACTERISTICS

Power Supply: 100V-120V or 200V-240V
50 Hz–60 Hz
5 amps (AC max) @ 120V
2 amps (AC max) @ 240V

ELECTRICAL CONSIDERATIONS

The emulator processor's user-pin timings and loadings are identical to the 8xC196KB component except as follows:

- Additional pin capacitance:

Target interface board (TIB)	Approximately 12pf (30pf maximum)
Pin 32 (P1.7/HOLD#)	Approximately 70pf
Pin 43 (READY)	Approximately 70pf
Pins 6 (RD#)	Approximately 60pf
Pin 63 (INST)	Approximately 60pf
Pin 16 (RESET#)	Approximately 325pf
All pins when using a hinge cable	10pf

- DC loading: Pin 1(VCC) can draw an additional 5 mA (15mA worst case @ 5.5V) due to power sensing circuitry.

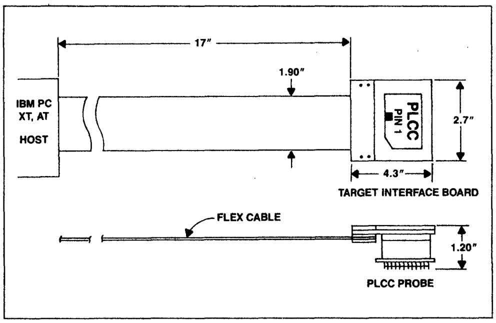
  Sensing circuitry may also draw approximately ±0.1mA (±0.4 mA maximum) DC current from any 8xC196KB output pin.

- AC timings:

Pin 32 (P1.7/HOLD#)	Degraded 1ns
Pin 43 (READY)	Degraded 1ns
Pin 63 (INST) if jumper E1-E3 is installed	Degraded 1ns
Pin 16 (RESET #)	Degraded 15ns

- ONCE support: If ONCE (on-circuit emulation) mode is selected, the RD# and INST pins are driven low while RESET# is active (low).

PROCESSOR MODULE DIMENSIONS



PHYSICAL CHARACTERISTICS

Target Probe

Width: 6.9 cm (2.7")
Height: 3.0 cm (1.2")
Length: 11.0 cm (4.3")
Package: 68-pin PLCC (optional 68-pin PGA adaptor available)

Emulator Chassis		Power Supply	
Width:	34 cm (13³/8")	Width:	18 cm (7¹/2")
Height:	12 cm (4¹/2")	Height:	10 cm (4")
Depth:	28 cm (10⁵/8")	Depth:	28 cm (11")
Weight:	3.2 kg (7 lb)	Weight:	7 kg (15 lb)

Probe Cable Length: 40 cm (17")
Serial Cable Length: 3.65 m (12')

ENVIRONMENTAL CHARACTERISTICS

Operating Temperature:	0°C to 40°C
Operating Humidity:	Maximum 85% relative humidity, non-condensing

ORDERING INFORMATION

ICE196KBHX	ICE in-circuit emulator base chassis, 196 emulation control board (ECB), 196KB target probe, 196KB crystal power accessory (CPA), enhanced break/trace board (BTB), 64k optional memory board (OMB), clips in/out, power supply and cable, serial cables for PC XT/AT, 68-pin PLCC target adapter. Host, 196KB probe, diagnostic, and tutorial software on 5¼″ media for DOS hosts running DOS 3.x. (Requires software license.)
ICE196KBMX	Same as ICE196KBHX except without enhanced break/trace board (BTB), without 64k optional memory board (OMB), and without clips in/out
ICEBTB	Enhanced break/trace board (BTB) for upgrading an ICE-196KB/MX system
ICEOMB	Optional memory board with 64k zero-waitstate mapped memory for upgrading an ICE-196KB/MX system
ICECLIPS	Clips in/out for upgrading an ICE-196KB/MX system (requires an enhanced break/trace board)
ADPTCA68PGA	Hinge cable for 68 pin PGA component packaging
ADPTCA68PLCC	Hinge cable for 68 pin PLCC component packaging
ADPTONC68PLCC	Adaptor to support 68 pin PLCC component packaging ONCE (on-circuit emulation)

For direct information on Intel's Development Tools, or for the number of your nearest sales office or distributor, call 800-874-6835 (U.S.). For information or literature on additional Intel products, call 800-548-4725 (U.S. and Canada).

Index

MCS®-96 INDEX

MCS®-96 INDEX (continued)

MCS®-96 INDEX (continued)

MCS®-96 INDEX (continued)

MCS®-96 INDEX (continued)

MCS®-96 INDEX (continued)

MCS®-96 INDEX (continued)

80C196KB INDEX

80C196KC INDEX

DOMESTIC SALES OFFICES

ALABAMA

†Intel Corp.
5015 Bradford Dr., #2
Huntsville 35805
Tel: (205) 830-4010
FAX: (205) 837-2640

ARIZONA

†Intel Corp.
11225 N. 28th Dr.
Suite D-214
Phoenix 85029
Tel: (602) 869-4980
FAX: (602) 869-4294

Intel Corp.
1161 N. El Dorado Place
Suite 301
Tucson 85715
Tel: (602) 299-6815
FAX: (602) 296-8234

CALIFORNIA

†Intel Corp.
21515 Vanowen Street
Suite 116
Canoga Park 91303
Tel: (818) 704-8500
FAX: (818) 340-1144

†Intel Corp.
2250 E. Imperial Highway
Suite 218
El Segundo 90245
Tel: (213) 640-6040
FAX: (213) 640-7133

Intel Corp.
1510 Arden Way
Suite 101
Sacramento 95815
Tel: (916) 920-8096
FAX: (916) 920-8253

†Intel Corp.
9665 Chesapeake Dr.
Suite 325
San Diego 95123
Tel: (619) 292-8086
FAX: (619) 292-0628

†Intel Corp.*
400 N. Tustin Avenue
Suite 450
Santa Ana 92705
Tel: (714) 835-9642
TWX: 910-595-1114
FAX: (714) 541-9157

†Intel Corp.*
San Tomas 4
2700 San Tomas Expressway
2nd Floor
Santa Clara 95051
Tel: (408) 986-8086
TWX: 910-338-0255
FAX: (408) 727-2620

COLORADO

Intel Corp.
4445 Northpark Drive
Suite 100
Colorado Springs 80907
Tel: (719) 594-6622
FAX: (303) 594-0720

†Intel Corp.*
650 S. Cherry St.
Suite 915
Denver 80222
Tel: (303) 321-8086
TWX: 910-931-2289
FAX: (303) 322-8670

CONNECTICUT

†Intel Corp.
301 Lee Farm Corporate Park
83 Wooster Heights Rd.
Danbury 06810
Tel: (203) 748-3130
FAX: (203) 794-0339

FLORIDA

†Intel Corp.
6363 N.W. 6th Way
Suite 100
Ft. Lauderdale 33309
Tel: (305) 771-0600
TWX: 510-956-9407
FAX: (305) 772-8193

†Intel Corp.
5850 T.G. Lee Blvd.
Suite 340
Orlando 32822
Tel: (407) 240-8000
FAX: (407) 240-8097

Intel Corp.
11300 4th Street North
Suite 170
St. Petersburg 33716
Tel: (813) 577-2413
FAX: (813) 578-1607

GEORGIA

Intel Corp.
20 Technology Parkway, N.W.
Suite 150
Norcross 30092
Tel: (404) 449-0541
FAX: (404) 605-9762

ILLINOIS

†Intel Corp.*
300 N. Martingale Road
Suite 400
Schaumburg 60173
Tel: (312) 605-8031
FAX: (312) 706-9762

INDIANA

†Intel Corp.
8777 Purdue Road
Suite 125
Indianapolis 46268
Tel: (317) 875-0623
FAX: (317) 875-8938

IOWA

Intel Corp.
1930 St. Andrews Drive N.E.
2nd Floor
Cedar Rapids 52402
Tel: (319) 393-1294

KANSAS

†Intel Corp.
10985 Cody St.
Suite 140, Bldg. D
Overland Park 66210
Tel: (913) 345-2727
FAX: (913) 345-2076

MARYLAND

†Intel Corp.*
10010 Junction Dr.
Suite 200
Annapolis Junction 20701
Tel: (301) 206-2860
FAX: (301) 206-3677
(301) 206-3678

MASSACHUSETTS

†Intel Corp.*
Westford Corp. Center
3 Carlisle Road
2nd Floor
Westford 01886
Tel: (508) 692-3222
TWX: 710-343-6333
FAX: (508) 692-7867

MICHIGAN

†Intel Corp.
7071 Orchard Lake Road
Suite 100
West Bloomfield 48322
Tel: (313) 851-8096
FAX: (313) 851-8770

MINNESOTA

†Intel Corp.
3500 W. 80th St.
Suite 360
Bloomington 55431
Tel: (612) 835-6722
TWX: 910-576-2867
FAX: (612) 831-6497

MISSOURI

†Intel Corp.
4203 Earth City Expressway
Suite 131
Earth City 63045
Tel: (314) 291-1990
FAX: (314) 291-4341

NEW JERSEY

†Intel Corp.*
Parkway 109 Office Center
328 Newman Springs Road
Red Bank 07701
Tel: (201) 747-2233
FAX: (201) 747-0983

†Intel Corp.
280 Corporate Center
75 Livingston Avenue
First Floor
Roseland 07068
Tel: (201) 740-0111
FAX: (201) 740-0626

NEW YORK

Intel Corp.*
850 Cross Keys Office Park
Fairport 14450
Tel: (716) 425-2750
TWX: 510-253-7391
FAX: (716) 223-2561

†Intel Corp.*
2950 Expressway Dr., South
Suite 130
Islandia 11722
Tel: (516) 231-3300
TWX: 510-227-6236
FAX: (516) 348-7939

†Intel Corp.
Westage Business Center
Bldg. 300, Route 9
Fishkill 12524
Tel: (914) 897-3860
FAX: (914) 897-3125

NORTH CAROLINA

†Intel Corp.
5800 Executive Center Dr.
Suite 105
Charlotte 28212
Tel: (704) 568-8966
FAX: (704) 535-2236

Intel Corp.
5540 Centerview Dr.
Suite 215
Raleigh 27606
Tel: (919) 851-9537
FAX: (919) 851-8974

OHIO

†Intel Corp.*
3401 Park Center Drive
Suite 220
Dayton 45414
Tel: (513) 890-5350
TWX: 810-450-2528
FAX: (513) 890-8658

†Intel Corp.*
25700 Science Park Dr.
Suite 100
Beachwood 44122
Tel: (216) 464-2736
TWX: 810-427-9298
FAX: (804) 282-0673

OKLAHOMA

Intel Corp.
6801 N. Broadway
Suite 115
Oklahoma City 73162
Tel: (405) 848-8086
FAX: (405) 840-9819

OREGON

†Intel Corp.
15254 N.W. Greenbrier Parkway
Building B
Beaverton 97005
Tel: (503) 645-8051
TWX: 910-467-8741
FAX: (503) 645-8181

PENNSYLVANIA

†Intel Corp.*
455 Pennsylvania Avenue
Suite 230
Fort Washington 19034
Tel: (215) 641-1000
TWX: 510-661-2077
FAX: (215) 641-0785

†Intel Corp.*
400 Penn Center Blvd.
Suite 610
Pittsburgh 15235
Tel: (412) 823-4970
FAX: (412) 829-7578

PUERTO RICO

†Intel Corp.
South Industrial Park
P.O. Box 910
Las Piedras 00671
Tel: (809) 733-8616

TEXAS

Intel Corp.
8911 Capital of Texas Hwy.
Austin 78759
Tel: (512) 794-8086
FAX: (512) 338-9335

†Intel Corp.*
12000 Ford Road
Suite 400
Dallas 75234
Tel: (214) 241-8087
FAX: (214) 484-1180

†Intel Corp.*
7322 S.W. Freeway
Suite 1490
Houston 77074
Tel: (713) 988-8086
TWX: 910-881-2490
FAX: (713) 988-3660

UTAH

†Intel Corp.
428 East 6400 South
Suite 104
Murray 84107
Tel: (801) 263-8051
FAX: (801) 268-1457

VIRGINIA

†Intel Corp.
1504 Santa Rosa Road
Suite 108
Richmond 23288
Tel: (804) 282-5668
FAX: (216) 464-2270

WASHINGTON

†Intel Corp.
155 108th Avenue N.E.
Suite 386
Bellevue 98004
Tel: (206) 453-8086
TWX: 910-443-3002
FAX: (206) 451-9556

Intel Corp.
408 N. Mullan Road
Suite 102
Spokane 99206
Tel: (509) 928-8086
FAX: (509) 928-9467

WISCONSIN

Intel Corp.
330 S. Executive Dr.
Suite 102
Brookfield 53005
Tel: (414) 784-8087
FAX: (414) 796-2115

CANADA

BRITISH COLUMBIA

Intel Semiconductor of
Canada, Ltd.
4585 Canada Way
Suite 202
Burnaby V5G 4L6
Tel: (604) 298-0387
FAX: (604) 298-8234

ONTARIO

†Intel Semiconductor of
Canada, Ltd.
2650 Queensview Drive
Suite 250
Ottawa K2B 8H6
Tel: (613) 829-9714
FAX: (613) 820-5936

†Intel Semiconductor of
Canada, Ltd.
190 Attwell Drive
Suite 500
Rexdale M9W 6H8
Tel: (416) 675-2105
FAX: (416) 675-2438

QUEBEC

Intel Semiconductor of
Canada, Ltd.
620 St. Jean Boulevard
Pointe Claire H9R 3K2
Tel: (514) 694-9130
FAX: 514-694-0064

†Sales and Service Office
*Field Application Location

DOMESTIC DISTRIBUTORS

ALABAMA

Arrow Electronics, Inc.
1015 Henderson Road
Huntsville 35805
Tel: (205) 837-6955

†Hamilton/Avnet Electronics
4940 Research Drive
Huntsville 35805
Tel: (205) 837-7210
TWX: 810-726-2162

Pioneer/Technologies Group, Inc.
4825 University Square
Huntsville 35805
Tel: (205) 837-9300
TWX: 810-726-2197

ARIZONA

†Hamilton/Avnet Electronics
505 S. Madison Drive
Tempe 85281
Tel: (602) 231-5140
TWX: 910-950-0077

Hamilton/Avnet Electronics
30 South McKiemy
Chandler 85226
Tel: (602) 961-6669
TWX: 910-950-0077

Arrow Electronics, Inc.
4134 E. Wood Street
Phoenix 85040
Tel: (602) 437-0750
TWX: 910-951-1550

Wyle Distribution Group
17855 N. Black Canyon Hwy.
Phoenix 85023
Tel: (602) 249-2232
TWX: 910-951-4282

CALIFORNIA

Arrow Electronics, Inc.
10824 Hope Street
Cypress 90630
Tel: (714) 220-6300

Arrow Electronics, Inc.
19748 Dearborn Street
Chatsworth 91311
Tel: (213) 701-7500
TWX: 910-493-2086

†Arow Electronics, Inc.
521 Weddell Drive
Sunnyvale 94086
Tel: (408) 745-6600
TWX: 910-339-9371

Arrow Electronics, Inc.
9511 Ridgehaven Court
San Diego 92123
Tel: (619) 565-4800
TWX: 888-064

†Arrow Electronics, Inc.
2961 Dow Avenue
Tustin 92680
Tel: (714) 838-5422
TWX: 910-595-2860

†Avnet Electronics
350 McCormick Avenue
Costa Mesa 92626
Tel: (714) 754-6071
TWX: 910-595-1928

†Hamilton/Avnet Electronics
1175 Bordeaux Drive
Sunnyvale 94086
Tel: (408) 743-3300
TWX: 910-339-9332

†Hamilton/Avnet Electronics
4545 Ridgeview Avenue
San Diego 92123
Tel: (619) 571-7500
TWX: 910-595-2638

†Hamilton/Avnet Electronics
9650 Desoto Avenue
Chatsworth 91311
Tel: (818) 700-1161

†Hamilton Electro Sales
10950 W. Washington Blvd.
Culver City 20230
Tel: (213) 558-2458
TWX: 910-340-6364

Hamilton Electro Sales
1361B West 190th Street
Gardena 90248
Tel: (213) 217-6700

†Hamilton/Avnet Electronics
3002 'G' Street
Ontario 91761
Tel: (714) 989-9411

†Avnet Electronics
20501 Plummer
Chatsworth 91351
Tel: (213) 700-6271
TWX: 910-494-2207

†Hamilton Electro Sales
3170 Pullman Street
Costa Mesa 92626
Tel: (714) 641-4150
TWX: 910-595-2638

†Hamilton/Avnet Electronics
4103 Northgate Blvd.
Sacramento 95834
Tel: (916) 920-3150

Wyle Distribution Group
124 Maryland Street
El Segundo 90254
Tel: (213) 322-8100

Wyle Distribution Group
7382 Lampson Ave.
Garden Grove 92641
Tel: (714) 891-1717
TWX: 910-348-7140 or 7111

Wyle Distribution Group
11151 Sun Center Drive
Rancho Cordova 95670
Tel: (916) 638-5282

†Wyle Distribution Group
9525 Chesapeake Drive
San Diego 92123
Tel: (619) 565-9171
TWX: 910-335-1590

†Wyle Distribution Group
3000 Bowers Avenue
Santa Clara 95051
Tel: (408) 727-2500
TWX: 910-338-0296

†Wyle Distribution Group
17872 Cowan Avenue
Irvine 92714
Tel: (714) 863-9953
TWX: 910-595-1572

Wyle Distribution Group
26677 W. Agoura Rd.
Calabasas 91302
Tel: (818) 880-9000
TWX: 372-0232

COLORADO

Arrow Electronics, Inc.
7060 South Tucson Way
Englewood 80112
Tel: (303) 790-4444

†Hamilton/Avnet Electronics
8765 E. Orchard Road
Suite 708
Englewood 80111
Tel: (303) 740-1017
TWX: 910-935-0787

†Wyle Distribution Group
451 E. 124th Avenue
Thornton 80241
Tel: (303) 457-9953
TWX: 910-936-0770

CONNECTICUT

†Arrow Electronics, Inc.
12 Beaumont Road
Wallingford 06492
Tel: (203) 265-7741
TWX: 710-476-0162

Hamilton/Avnet Electronics
Commerce Industrial Park
Commerce Drive
Danbury 06810
Tel: (203) 797-2800
TWX: 710-456-9974

†Pioneer Electronics
112 Main Street
Norwalk 06851
Tel: (203) 853-1515
TWX: 710-468-3373

FLORIDA

†Arrow Electronics, Inc.
400 Fairway Drive
Suite 102
Deerfield Beach 33441
Tel: (305) 429-8200
TWX: 510-955-9456

Arrow Electronics, Inc.
37 Skyline Drive
Suite 3101
Lake Marv 32746
Tel: (407) 323-0252
TWX: 510-959-6337

†Hamilton/Avnet Electronics
6801 N.W. 15th Way
Ft. Lauderdale 33309
Tel: (305) 971-2900
TWX: 510-956-3097

†Hamilton/Avnet Electronics
3197 Tech Drive North
St. Petersburg 33702
Tel: (813) 576-3930
TWX: 810-863-0374

†Hamilton/Avnet Electronics
6947 University Boulevard
Winter Park 32792
Tel: (305) 628-3888
TWX: 810-853-0322

†Pioneer/Technologies Group, Inc.
337 S. Lake Blvd.
Alta Monte Springs 32701
Tel: (407) 834-9090
TWX: 810-853-0284

Pioneer/Technologies Group, Inc.
674 S. Military Trail
Deerfield Beach 33442
Tel: (305) 428-8877
TWX: 510-955-9653

GEORGIA

†Arrow Electronics, Inc.
3155 Northwoods Parkway
Suite A
Norcross 30071
Tel: (404) 449-8252
TWX: 810-766-0439

†Hamilton/Avnet Electronics
5825 D Peachtree Corners
Norcross 30092
Tel: (404) 447-7500
TWX: 810-766-0432

Pioneer/Technologies Group, Inc.
3100 F Northwoods Place
Norcross 30071
Tel: (404) 448-1711
TWX: 810-766-4515

ILLINOIS

Arrow Electronics, Inc.
1140 W. Thorndale
Itasca 60143
Tel: (312) 250-0500
TWX: 312-250-0916

†Hamilton/Avnet Electronics
1130 Thorndale Avenue
Bensenville 60106
Tel: (312) 860-7780
TWX: 910-227-0060

MTI Systems Sales
1100 W. Thorndale
Itasca 60143
Tel: (312) 773-2300

†Pioneer Electronics
1551 Carmen Drive
Elk Grove Village 60007
Tel: (312) 437-9680
TWX: 910-222-1834

INDIANA

†Arrow Electronics, Inc.
2495 Directors Row, Suite H
Indianapolis 46241
Tel: (317) 243-9353
TWX: 810-341-3119

Hamilton/Avnet Electronics
485 Gradle Drive
Carmel 46032
Tel: (317) 844-9333
TWX: 810-260-3966

†Pioneer Electronics
6408 Castleplace Drive
Indianapolis 46250
Tel: (317) 849-7300
TWX: 810-260-1794

IOWA

Hamilton/Avnet Electronics
915 33rd Avenue, S.W.
Cedar Rapids 52404
Tel: (319) 362-4757

KANSAS

Arrow Electronics
8208 Melrose Dr., Suite 210
Lenexa 66214
Tel: (913) 541-9542

†Hamilton/Avnet Electronics
9219 Quivera Road
Overland Park 66215
Tel: (913) 888-8900
TWX: 910-743-0005

Pioneer/Tec Gr.
10551 Lockman Rd.
Lenexa 66215
Tel: (913) 492-0500

KENTUCKY

Hamilton/Avnet Electronics
1051 D. Newton Park
Lexington 40511
Tel: (606) 259-1475

MARYLAND

Arrow Electronics, Inc.
8300 Guilford Drive
Suite H, River Center
Columbia 21046
Tel: (301) 995-0003
TWX: 710-236-9005

Hamilton/Avnet Electronics
6822 Oak Hall Lane
Columbia 21045
Tel: (301) 995-3500
TWX: 710-862-1861

†Mesa Technology Corp.
9720 Patuxent Woods Dr.
Columbia 21046
Tel: (301) 290-8150
TWX: 710-828-9702

†Pioneer/Technologies Group, Inc.
9100 Gaither Road
Gaithersburg 20877
Tel: (301) 921-0660
TWX: 710-828-0545

Arrow Electronics, Inc.
7524 Standish Place
Rockville 20855
Tel: 301-424-0244

MASSACHUSETTS

Arrow Electronics, Inc.
25 Upton Dr.
Wilmington 01887
Tel: (617) 935-5134

†Hamilton/Avnet Electronics
10D Centennial Drive
Peabody 01960
Tel: (617) 531-7430
TWX: 710-393-0382

MTI Systems Sales
83 Cambridge St.
Burlington 01813

Pioneer Electronics
44 Hartwell Avenue
Lexington 02173
Tel: (617) 861-9200
TWX: 710-326-6617

MICHIGAN

Arrow Electronics, Inc.
755 Phoenix Drive
Ann Arbor 48104
Tel: (313) 971-8220
TWX: 810-223-6020

Hamilton/Avnet Electronics
2215 29th Street S.E.
Space A5
Grand Rapids 49508
Tel: (616) 243-8805
TWX: 810-274-6921

Pioneer Electronics
4504 Broadmoor S.E.
Grand Rapids 49508
FAX: 616-698-1831

†Hamilton/Avnet Electronics
32487 Schoolcraft Road
Livonia 48150
Tel: (313) 522-4700
TWX: 810-282-8775

†Pioneer/Michigan
13485 Stamford
Livonia 48150
Tel: (313) 525-1800
TWX: 810-242-3271

MINNESOTA

†Arrow Electronics, Inc.
5230 W. 73rd Street
Edina 55435
Tel: (612) 830-1800
TWX: 910-576-3125

†Hamilton/Avnet Electronics
12400 Whitewater Drive
Minnetonka 55434
Tel: (612) 932-0600

†Pioneer Electronics
7625 Golden Triange Dr.
Suite G
Eden Prairi 55343
Tel: (612) 944-3355

MISSOURI

†Arrow Electronics, Inc.
2380 Schuetz
St. Louis 63141
Tel: (314) 567-6888
TWX: 910-764-0882

†Hamilton/Avnet Electronics
13743 Shoreline Court
Earth City 63045
Tel: (314) 344-1200
TWX: 910-762-0684

NEW HAMPSHIRE

†Arrow Electronics, Inc.
3 Perimeter Road
Manchester 03103
Tel: (603) 668-6968
TWX: 710-220-1684

†Hamilton/Avnet Electronics
444 E. Industrial Drive
Manchester 03103
Tel: (603) 624-9400

†Microcomputer System Technical Distributor Center

DOMESTIC DISTRIBUTORS (Contd.)

NEW JERSEY

†Arrow Electronics, Inc.
Four East Stow Road
Unit 11
Marlton 08053
Tel: (609) 596-8000
TWX: 710-897-0829

†Arrow Electronics
6 Century Drive
Parsipanny 07054
Tel: (201) 538-0900

†Hamilton/Avnet Electronics
1 Keystone Ave., Bldg. 36
Cherry Hill 08003
Tel: (609) 424-0110
TWX: 710-940-0262

†Hamilton/Avnet Electronics
10 Industrial
Fairfield 07006
Tel: (201) 575-5300
TWX: 710-734-4388

†MTI Systems Sales
37 Kulick Rd.
Fairfield 07006
Tel: (201) 227-5552

†Pioneer Electronics
45 Route 46
Pinebrook 07058
Tel: (201) 575-3510
TWX: 710-734-4382

NEW MEXICO

Alliance Electronics Inc.
11030 Cochiti S.E.
Albuquerque 87123
Tel: (505) 292-3360
TWX: 910-989-1151

Hamilton/Avnet Electronics
2524 Baylor Drive S.E.
Albuquerque 87106
Tel: (505) 765-1500
TWX: 910-989-0614

NEW YORK

†Arrow Electronics, Inc.
3375 Brighton Henrietta
Townline Rd.
Rochester 14623
Tel: (716) 275-0300
TWX: 510-253-4766

Arrow Electronics, Inc.
20 Oser Avenue
Hauppauge 11788
Tel: (516) 231-1000
TWX: 510-227-6623

Hamilton/Avnet
933 Motor Parkway
Hauppauge 11788
Tel: (516) 231-9800
TWX: 510-224-6166

†Hamilton/Avnet Electronics
333 Metro Park
Rochester 14623
Tel: (716) 475-9130
TWX: 510-253-5470

†Hamilton/Avnet Electronics
103 Twin Oaks Drive
Syracuse 13206
Tel: (315) 437-0288
TWX: 710-541-1560

†MTI Systems Sales
38 Harbor Park Drive
Port Washington 11050
Tel: (516) 621-6200

†Pioneer Electronics
68 Corporate Drive
Binghamton 13904
Tel: (607) 722-9300
TWX: 510-252-0893

Pioneer Electronics
40 Oser Avenue
Hauppauge 11787
Tel: (516) 231-9200

†Pioneer Electronics
60 Crossway Park West
Woodbury, Long Island 11797
Tel: (516) 921-8700
TWX: 510-221-2184

†Pioneer Electronics
840 Fairport Park
Fairport 14450
Tel: (716) 381-7070
TWX: 510-253-7001

NORTH CAROLINA

†Arrow Electronics, Inc.
5240 Greensdairy Road
Raleigh 27604
Tel: (919) 876-3132
TWX: 510-928-1856

†Hamilton/Avnet Electronics
3510 Spring Forest Drive
Raleigh 27604
Tel: (919) 878-0819
TWX: 510-928-1836

Pioneer/Technologies Group, Inc.
9801 A-Southern Pine Blvd.
Charlotte 28210
Tel: (919) 527-8188
TWX: 810-621-0366

OHIO

Arrow Electronics, Inc.
7620 McEwen Road
Centerville 45459
Tel: (513) 435-5563
TWX: 810-459-1611

†Arrow Electronics, Inc.
6238 Cochran Road
Solon 44139
Tel: (216) 248-3990
TWX: 810-427-9409

†Hamilton/Avnet Electronics
954 Senate Drive
Dayton 45459
Tel: (513) 439-6733
TWX: 810-450-2531

Hamilton/Avnet Electronics
4588 Emery Industrial Pkwy.
Warrensville Heights 44128
Tel: (216) 349-5100
TWX: 810-427-9452

†Hamilton/Avnet Electronics
777 Brooksedge Blvd.
Westerville 43081
Tel: (614) 882-7004

†Pioneer Electronics
4433 Interpoint Boulevard
Dayton 45424
Tel: (513) 236-9900
TWX: 810-459-1622

†Pioneer Electronics
4800 E. 131st Street
Cleveland 44105
Tel: (216) 587-3600
TWX: 810-422-2211

OKLAHOMA

Arrow Electronics, Inc.
1211 E. 51st St., Suite 101
Tulsa 74146
Tel: (918) 252-7537

†Hamilton/Avnet Electronics
12121 E. 51st St., Suite 102A
Tulsa 74146
Tel: (918) 252-7297

OREGON

†Almac Electronics Corp.
1885 N.W. 169th Place
Beaverton 97005
Tel: (503) 629-8090
TWX: 910-467-8746

†Hamilton/Avnet Electronics
6024 S.W. Jean Road
Bldg. C, Suite 10
Lake Oswego 97034
Tel: (503) 635-7848
TWX: 910-455-8179

Wyle Distribution Group
5250 N.E. Elam Young Parkway
Suite 600
Hillsboro 97124
Tel: (503) 640-6000
TWX: 910-460-2203

PENNSYLVANIA

Arrow Electronics, Inc.
650 Seco Road
Monroeville 15146
Tel: (412) 856-7000

Hamilton/Avnet Electronics
2800 Liberty Ave.
Pittsburgh 15238
Tel: (412) 281-4150

Pioneer Electronics
259 Kappa Drive
Pittsburgh 15238
Tel: (412) 782-2300
TWX: 710-795-3122

†Pioneer/Technologies Group, Inc.
Delaware Valley
261 Gibralter Road
Horsham 19044
Tel: (215) 674-4000
TWX: 510-665-6778

TEXAS

†Arrow Electronics, Inc.
3220 Commander Drive
Carrollton 75006
Tel: (214) 380-6464
TWX: 910-860-5377

†Arrow Electronics, Inc.
10899 Kinghurst
Suite 100
Houston 77099
Tel: (713) 530-4700
TWX: 910-880-4439

†Arrow Electronics, Inc.
2227 W. Braker Lane
Austin 78758
Tel: (512) 835-4180
TWX: 910-874-1348

†Hamilton/Avnet Electronics
1807 W. Braker Lane
Austin 78758
Tel: (512) 837-8911
TWX: 910-874-1319

†Hamilton/Avnet Electronics
2111 W. Walnut Hill Lane
Irving 75038
Tel: (214) 550-6111
TWX: 910-860-5929

†Hamilton/Avnet Electronics
4850 Wright Rd., Suite 190
Stafford 77477
Tel: (713) 240-7733
TWX: 910-881-5523

†Pioneer Electronics
18260 Kramer
Austin 78758
Tel: (512) 835-4000
TWX: 910-874-1323

†Pioneer Electronics
13710 Omega Road
Dallas 75234
Tel: (214) 386-7300
TWX: 910-850-5563

†Pioneer Electronics
5853 Point West Drive
Houston 77036
Tel: (713) 988-5555
TWX: 910-881-1606

Wyle Distribution Group
1810 Greenville Avenue
Richardson 75081
Tel: (214) 235-9953

UTAH

Arrow Electronics
1946 Parkway Blvd.
Salt Lake City 84119
Tel: (801) 973-6913

†Hamilton/Avnet Electronics
1585 West 2100 South
Salt Lake City 84119
Tel: (801) 972-2800
TWX: 910-925-4018

Wyle Distribution Group
1325 West 2200 South
Suite E
West Valley 84119
Tel: (801) 974-9953

WASHINGTON

†Almac Electronics Corp.
14360 S.E. Eastgate Way
Bellevue 98007
Tel: (206) 643-9992
TWX: 910-444-2067

Arrow Electronics, Inc.
19540 68th Ave. South
Kent 98032
Tel: (206) 575-4420

†Hamilton/Avnet Electronics
14212 N.E. 21st Street
Bellevue 98005
Tel: (206) 643-3950
TWX: 910-443-2469

Wyle Distribution Group
15385 N.E. 90th Street
Redmond 98052
Tel: (206) 881-1150

WISCONSIN

Arrow Electronics, Inc.
200 N. Patrick Blvd., Ste. 100
Brookfield 53005
Tel: (414) 767-6600
TWX: 910-262-1193

Hamilton/Avnet Electronics
2975 Moorland Road
New Berlin 53151
Tel: (414) 784-4510
TWX: 910-262-1182

CANADA

ALBERTA

Hamilton/Avnet Electronics
2816 21st Street N.E.
Calgary T2E 6Z3
Tel: (403) 230-3586
TWX: 03-827-642

Zentronics
Bay No. 1
3300 14th Avenue N.E.
Calgary T2A 6J4
Tel: (403) 272-1021

BRITISH COLUMBIA

†Hamilton/Avnet Electronics
105-2550 Boundary
Burmalay V5M 3Z3
Tel: (604) 437-6667

Zentronics
108-11400 Bridgeport Road
Richmond V6X 1T2
Tel: (604) 273-5575
TWX: 04-5077-89

MANITOBA

Zentronics
60-1313 Border Unit 60
Winnipeg R3H 0X4
Tel: (204) 694-1957

ONTARIO

Arrow Electronics, Inc.
36 Antares Dr.
Nepean K2E 7W5
Tel: (613) 226-6903

Arrow Electronics, Inc.
1093 Meyerside
Mississauga L5T 1M4
Tel: (416) 673-7769
TWX: 06-218213

†Hamilton/Avnet Electronics
6845 Rexwood Road
Units 3-4-5
Mississauga L4T 1R2
Tel: (416) 677-7432
TWX: 610-492-8867

Hamilton/Avnet Electronics
6845 Rexwood Rd., Unit 6
Mississauga L4T 1R2
Tel: (416) 277-0484

†Hamilton/Avnet Electronics
190 Colonnade Road South
Nepean K2E 7L5
Tel: (613) 226-1700
TWX: 05-349-71

†Zentronics
8 Tilbury Court
Brampton L6T 3T4
Tel: (416) 451-9600
TWX: 06-976-78

†Zentronics
155 Colonnade Road
Unit 17
Nepean K2E 7K1
Tel: (613) 226-8840

Zentronics
60-1313 Border St.
Winnipeg R3H 0I4
Tel: (204) 694-7957

QUEBEC

†Arrow Electronics Inc.
4050 Jean Talon Quest
Montreal H4P 1W1
Tel: (514) 735-5511
TWX: 05-25590

Arrow Electronics, Inc.
500 Avenue St-Jean Baptiste
Suite 280
Quebec G2E 5R9
Tel: (418) 871-7500
FAX: 418-871-6816

Hamilton/Avnet Electronics
2795 Halpern
St. Laurent H2E 7K1
Tel: (514) 335-1000
TWX: 610-421-3731

Zentronics
817 McCaffrey
St. Laurent H4T 1M3
Tel: (514) 737-9700
TWX: 05-827-535

†Microcomputer System Technical Distributor Center

EUROPEAN SALES OFFICES

DENMARK

Intel Denmark A/S
Glentevej 61, 3rd Floor
2400 Copenhagen NV
Tel: (45) (31) 19 80 33
TLX: 19567

FINLAND

Intel Finland OY
Ruosilantie 2
00390 Helsinki
Tel: (358) 0 544 644
TLX: 123332

FRANCE

Intel Corporation S.A.R.L.
1, Rue Edison-BP 303
78054 St. Quentin-en-Yvelines
Cedex
Tel: (33) (1) 30 57 70 00
TLX: 699016

WEST GERMANY

Intel Semiconductor GmbH*
Dornacher Strasse 1
8016 Feldkirchen bei Muenchen
Tel: (49) 089/90992-0
TLX: 5-23177

Intel Semiconductor GmbH
Hohenzollern Strasse 5
3000 Hannover 1
Tel: (49) 0511/344081
TLX: 9-23625

Intel Semiconductor GmbH
Abraham Lincoln Strasse 16-18
6200 Wiesbaden
Tel: (49) 06121/7605-0
TLX: 4-186183

Intel Semiconductor GmbH
Zettachring 10A
7000 Stuttgart 80
Tel: (49) 0711/7287-280
TLX: 7-254826

ISRAEL

Intel Semiconductor Ltd.*
Atidim Industrial Park-Neve Sharet
P.O. Box 43202
Tel-Aviv 61430
Tel: (972) 03-498080
TLX: 371215

ITALY

Intel Corporation Italia S.p.A.*
Milanofiori Palazzo E
20090 Assago
Milano
Tel: (39) (02) 89200950
TLX: 341286

NETHERLANDS

Intel Semiconductor B.V.*
Postbus 84130
3099 CC Rotterdam
Tel: (31) 10.407.11.11
TLX: 22283

NORWAY

Intel Norway A/S
Hvamveien 4-PO Box 92
2013 Skjetten
Tel: (47) (6) 842 420
TLX: 78018

SPAIN

Intel Iberia S.A.
Zurbaran, 28
28010 Madrid
Tel: (34) (1) 308.25.52
TLX: 46880

SWEDEN

Intel Sweden A.B.*
Dalvagen 24
171 36 Solna
Tel: (46) 8 734 01 00
TLX: 12261

SWITZERLAND

Intel Semiconductor A.G.
Zuerichstrasse
8185 Winkel-Rueti bei Zuerich
Tel: (41) 01/860 62 62
TLX: 825977

UNITED KINGDOM

Intel Corporation (U.K.) Ltd.*
Pipers Way
Swindon, Wiltshire SN3 1RJ
Tel: (44) (0793) 696000
TLX: 444447/8

EUROPEAN DISTRIBUTORS/REPRESENTATIVES

AUSTRIA

Bacher Electronics G.m.b.H.
Rotenmuehlgasse 26
1120 Wien
Tel: (43) (0222) 83 56 46
TLX: 31532

BELGIUM

Inelco Belgium S.A.
Av. des Croix de Guerre 94
1120 Bruxelles
Oorlogskruisenlaan, 94
1120 Brussel
Tel: (32) (02) 216 01 60
TLX: 64475 or 22090

DENMARK

ITT-Multikomponent
Naverland 29
2600 Glostrup
Tel: (45) (0) 2 45 66 45
TLX: 33 355

FINLAND

OY Fintronic AB
Melkonkatu 24A
00210 Helsinki
Tel: (358) (0) 6926022
TLX: 124224

FRANCE

Almex
Zone industrielle d'Antony
48, rue de l'Aubepine
BP 102
92164 Antony cedex
Tel: (33) (1) 46 66 21 12
TLX: 250067

Jermyn-Generim
60, rue des Gemeaux
Silic 580
94653 Rungis cedex
Tel: (33) (1) 49 78 49 78
TLX: 261585

Metrologie
Tour d'Asnieres
4, av. Laurent-Cely
92606 Asnieres Cedex
Tel: (33) (1) 47 90 62 40
TLX: 611448

Tekelec-Airtronic
Cite des Bruyeres
Rue Carle Vernet - BP 2
92310 Sevres
Tel: (33) (1) 45 34 75 35
TLX: 204552

WEST GERMANY

Electronic 2000 AG
Stahlgruberring 12
8000 Muenchen 82
Tel: (49) 089/42001-0
TLX: 522561

ITT Multikomponent GmbH
Postfach 1265
Bahnhofstrasse 44
7141 Moeglingen
Tel: (49) 07141/4879
TLX: 7264472

Jermyn GmbH
Im Dachsstueck 9
6250 Limburg
Tel: (49) 06431/508-0
TLX: 415257-0

Metrologie GmbH
Meglingerstrasse 49
8000 Muenchen 71
Tel: (49) 089/78042-0
TLX: 5213189

Proelectron Vertriebs GmbH
Max Planck Strasse 1-3
6072 Dreieich
Tel: (49) 06103/30434-3
TLX: 417903

IRELAND

Micro Marketing Ltd.
Glenageary Office Park
Glenageary
Co. Dublin
Tel: (21) (353) (01) 85 63 25
TLX: 31584

ISRAEL

Eastronics Ltd.
11 Rozanis Street
P.O.B. 39300
Tel-Aviv 61392
Tel: (972) 03-475151
TLX: 33638

ITALY

Intesi
Divisione ITT Industries GmbH
Viale Milanofiori
Palazzo E/5
20090 Assago (MI)
Tel: (39) 02/824701
TLX: 311351

Lasi Elettronica S.p.A.
V. le Fulvio Testi, 126
20092 Cinisello Balsamo (MI)
Tel: (39) 02/2440012
TLX: 352040

Telcom S.r.l.
Via M. Civitali 75
20148 Milano
Tel: (39) 02/4049046
TLX: 335654

ITT Multicomponents
Viale Milanofiori E/5
20090 Assago (MI)
Tel: (39) 02/824701
TLX: 311351

Silverstar
Via Dei Gracchi 20
20146 Milano
Tel: (39) 02/49961
TLX: 332189

NETHERLANDS

Koning en Hartman Elektrotechniek
B.V.
Energieweg 1
2627 AP Delft
Tel: (31) (0) 15/609906
TLX: 38250

NORWAY

Nordisk Elektronikk (Norge) A/S
Postboks 123
Smedsvingen 4
1364 Hvalstad
Tel: (47) (02) 84 62 10
TLX: 77546

PORTUGAL

ATD Portugal LDA
Rua Dos Lusiados, 5 Sala B
1300 Lisboa
Tel: (35) (1) 64 80 91
TLX: 61562

Ditram
Avenida Miguel Bombarda, 133
1000 Lisboa
Tel: (35) (1) 54 53 13
TLX: 14182

SPAIN

ATD Electronica, S.A.
Plaza Ciudad de Viena, 6
28040 Madrid
Tel: (34) (1) 234 40 00
TLX: 42477

ITT-SESA
Calle Miguel Angel, 21-3
28010 Madrid
Tel: (34) (1) 419 09 57
TLX: 27461

Metrologia Iberica, S.A.
Ctra. de Fuencarral, n.80
28100 Alcobendas (Madrid)
Tel: (34) (1) 653 86 11

SWEDEN

Nordisk Elektronik AB
Torshamnsgatan 39
Box 36
164 93 Kista
Tel: (46) 08-03 46 30
TLX: 105 47

SWITZERLAND

Industrade A.G.
Hertistrasse 31
8304 Wallisellen
Tel: (41) (01) 8328111
TLX: 56788

TURKEY

EMPA Electronic
Lindwurmstrasse 95A
8000 Muenchen 2
Tel: (49) 089/53 80 570
TLX: 528573

UNITED KINGDOM

Accent Electronic Components Ltd.
Jubilee House, Jubilee Road
Letchworth, Herts SG6 1TL
Tel: (44) (0462) 686666
TLX: 826293

Bytech-Comway Systems
3 The Western Centre
Western Road
Bracknell RG12 1RW
Tel: (44) (0344) 55333
TLX: 847201

Jermyn
Vestry Estate
Otford Road
Sevenoaks
Kent TN14 5EU
Tel: (44) (0732) 450144
TLX: 95142

MMD
Unit 8 Southview Park
Caversham
Reading
Berkshire RG4 0AF
Tel: (44) (0734) 481666
TLX: 846669

Rapid Silicon
Rapid House
Denmark Street
High Wycombe
Buckinghamshire HP11 2ER
Tel: (44) (0494) 442266
TLX: 837931

Rapid Systems
Rapid House
Denmark Street
High Wycombe
Buckinghamshire HP11 2ER
Tel: (44) (0494) 450244
TLX: 837931

YUGOSLAVIA

H.R. Microelectronics Corp.
2005 de la Cruz Blvd., Ste. 223
Santa Clara, CA 95050
U.S.A.
Tel: (1) (408) 988-0286
TLX: 387452

Rapido Electronic Components
S.p.a.
Via C. Beccaria, 8
34133 Trieste
Italia
Tel: (39) 040/360555
TLX: 460461

*Field Application Location

16-Bit Embedded Controller

If you need the performance of a 16-bit controller, this book has the answers. Real-time event control....motion control.... motor control....all these and more can be handled by the Intel MCS®-96 family of 16-bit embedded controllers. The MCS-96 product line is designed to be used in applications which require high speed calculations and fast I/O operations.

The 80C196KB family is a CHMOS branch of the MCS-96 family. Included in this volume are comprehensive 80C196KB and 80C196KC User's Guides.

In this handbook are complete specifications including data sheets and architectural descriptions for these products. Also included are instruction sets and development support tools; comprehensive charts covering symbols and functions; block diagrams, electrical characteristics and functional descriptions.

Printed in U.S.A./0190/62K/RRD JB
Embedded Control

ISBN 1-55512-092-X