MC68302UM/AD
REV 1

MC68302

Integrated Multi-Protocol Processor User's Manual

MOTOROLA

MC68302

INTEGRATED MULTIPROTOCOL PROCESSOR USER'S MANUAL

MOTOROLA INC., 1990

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

LIST OF ILLUSTRATIONS (Concluded)

LIST OF TABLES

SECTION 1
GENERAL DESCRIPTION

The MC68302 integrated multiprotocol processor (IMP) is a very large-scale integration (VLSI) device incorporating the main building blocks needed for the design of a wide variety of controllers. The device is especially suitable to applications in the communications industry. The IMP is the first device to offer the benefits of a closely coupled, industry-standard M68000 microprocessor core and a flexible communications architecture. The IMP may be configured to support a number of popular industry interfaces, including those for the Integrated Services Digital Network (ISDN) basic rate and terminal adaptor applications. Concurrent operation of different protocols is easily achieved through a combination of architectural and programmable features. Data concentrators, line cards, bridges, and gateways are examples of suitable applications for this device.

The IMP is a high-density complementary metal-oxide semiconductor (HCMOS) device consisting of an M68000 microprocessor core, a system integration block (SIB), and a communications processor (CP).

1.1 BLOCK DIAGRAM

The block diagram is shown in Figure 1-1.

By integrating the microprocessor core with the serial ports (in the CP) and the system peripherals (in the SIB), the IMP is capable of handling complex tasks such as all ISDN basic rate (2B + D) access tasks. For example, the IMP architecture and the serial communications controller (SCC) ports can support the interface of an S/T transceiver chip and the lower part (bit handling) ISO/OSI layer-2 functions. Other layer-2 functions and the higher protocol layers would then be implemented by software executed by the M68000 core.

Using the flexible memory-based buffer structure of the IMP, terminal adaptor applications also can be supported by transforming and sharing data buffer information between the three SCC ports and the serial communications port (SCP). Each SCC channel is available for HDLC/SDLC™, UART, BISYNC, DDCMP™, V.110, or transparent operation. The IMP provides a number of choices for various rate adaption techniques and can be used for functions such as a terminal controller, multiplexer, or concentrator.

SDLC is a trademark of International Business Machines.
DDCMP is a trademark of Digital Equipment Corporation.

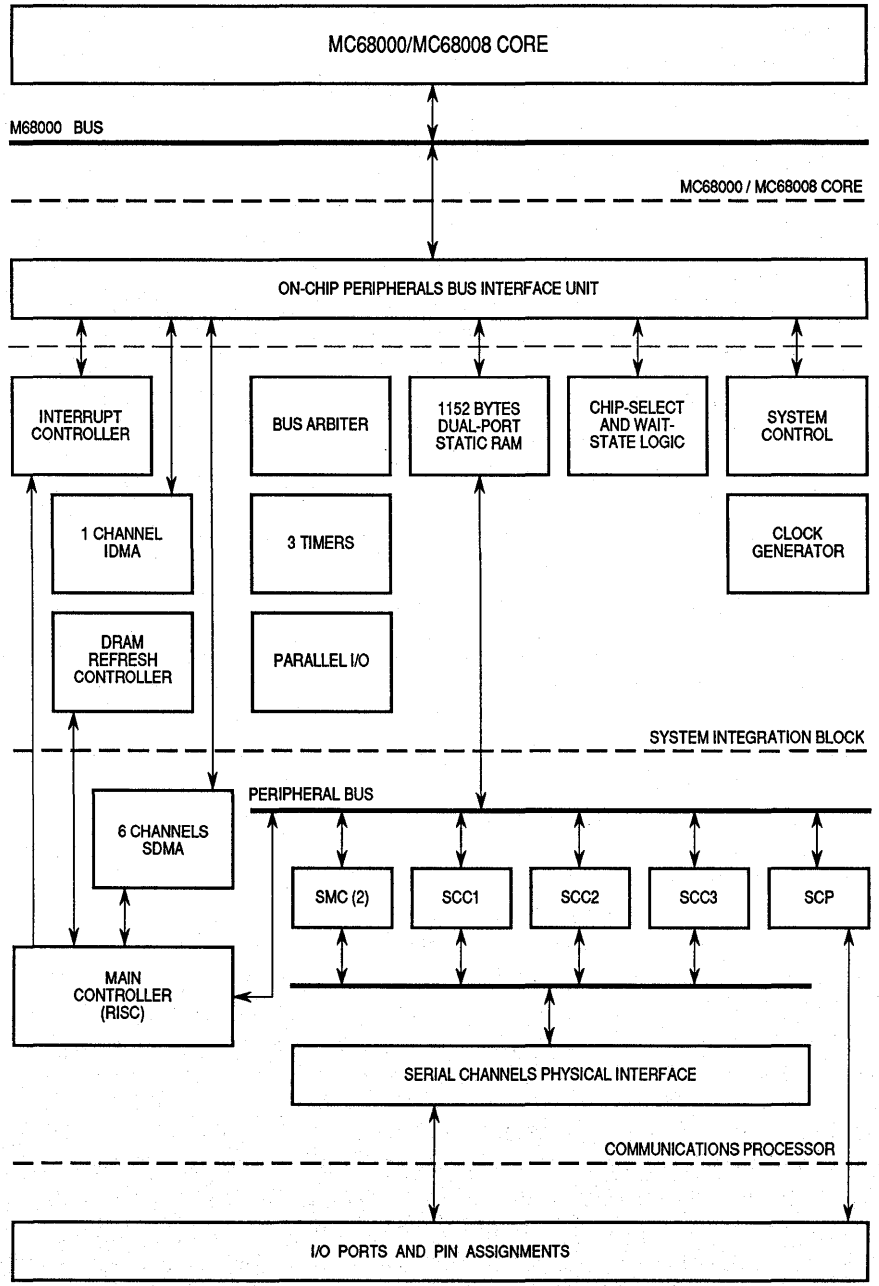


Figure 1-1. MC68302 Block Diagram

The MC68302 can also be used in applications such as board-level industrial controllers performing real-time control applications with a local control bus and an X.25 packet network connection. Such a system provides the real-time response to a demanding peripheral while permitting remote monitoring and communication through an X.25 packet network.

1.2 FEATURES

The features of the IMP are as follows:

- On-Chip HCMOS MC68000/MC68008 Core Supporting a 16- or 8-Bit M68000 Family System
- SIB Including:
  - Independent Direct Memory Access (IDMA) Controller with Three Handshake Signals: $\overline{\text{DREQ}}$, $\overline{\text{DACK}}$, and $\overline{\text{DONE}}$.
  - Interrupt Controller with Two Modes of Operation
  - Parallel Input/Output (I/O) Ports, Some with Interrupt Capability
  - On-Chip 1152-Byte Dual-Port RAM
  - Three Timers Including a Watchdog Timer
  - Four Programmable Chip-Select Lines with Wait-State Generator Logic
  - Programmable Address Mapping of the Dual-Port RAM and IMP Registers
  - On-Chip Clock Generator with Output Signal
  - System Control
    - Bus Arbitration Logic with Low-Interrupt Latency Support
    - System Status and Control Logic
    - Disable CPU Logic (M68000)
    - Hardware Watchdog
    - Low-Power (Standby) Modes
    - Freeze Control for Debugging
    - DRAM Refresh Controller
- CP Including:
  - Main Controller (RISC Processor)
  - Three Independent Full-Duplex Serial Communications Controllers (SCCs) Supporting Various Protocols:
    - High-Level/Synchronous Data Link Control (HDLC/SDLC)
    - Universal Asynchronous Receiver Transmitter (UART)
    - Binary Synchronous Communication (BISYNC)
    - Synchronous/Asynchronous Digital Data Communications Message Protocol (DDCMP)
    - Transparent Modes
    - V.110 Rate Adaption
  - Six Serial DMA Channels for the Three SCCs

Flexible Physical Interface Accessible by SCCs Including:
- Motorola Interchip Digital Link (IDL)
- General Circuit Interface (GCI also known as IOM™-2)
- Pulse Code Modulation (PCM) Highway Interface
- Nonmultiplexed Serial Interface (NMSI) Implementing Standard Modem Signals

SCP for Synchronous Communication

Two Serial Management Controllers (SMCs) To Support IDL and GCI Auxiliary Channels

1

1.3 MC68302 SYSTEM ARCHITECTURE

Most general-purpose microprocessor-based systems use an architecture that interfaces all peripheral devices directly onto a single microprocessor bus (see Figure 1-2).

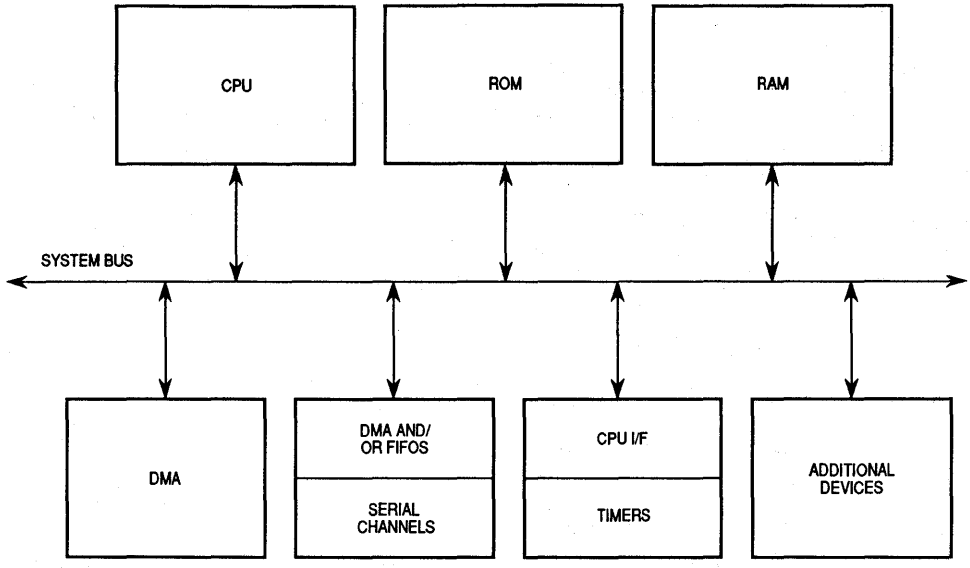


Figure 1-2. General-Purpose Microprocessor System Design

The MC68302 microprocessor architecture is shown in Figure 1-3. In this architecture, the peripheral devices are isolated from the system bus through a dual-port memory. Various parameters and counters and all memory buffer descriptor tables reside in the dual-port RAM. The receive and transmit data

IOM is a trademark of Siemens AG.

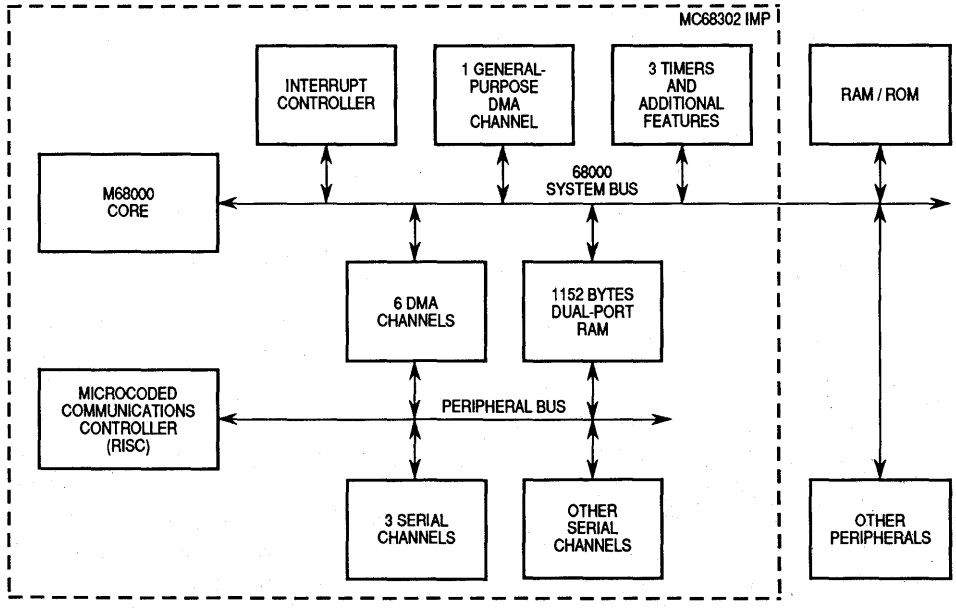


Figure 1-3. MC68302 System Design

buffers may be located in the on-chip RAM or in the off-chip system RAM. Six DMA channels are dedicated to the six serial ports (receive and transmit for each of the three SCC channels). If data for an SCC channel is programmed to be located in the external RAM, the CP will program the corresponding DMA channel for the required accesses, bypassing the dual-port RAM. If data resides in the dual-port RAM, then the CP accesses the RAM with one clock cycle and no arbitration delays.

The use of a unique arbitration scheme and synchronous transfers between the microprocessor and dual-port RAM gives the appearance of zero wait-state operation to the M68000 microprocessor core. The dual-port RAM can be accessed by the CP main controller (RISC) once every clock cycle for either read or write operations. When the M68000 core accesses the dual-port RAM, each access is pipelined along with the CP accesses so that data is read or written without conflict. The net effect is the loss of a single memory access by the CP main controller per M68000 core access.

The buffer memory structure of the MC68302 can be configured to closely match I/O channel requirements by careful selection of buffer size and buffer

linking. The interrupt structure is also programmable so that the on-chip M68000 processor can be off-loaded from the peripheral bit-handling functions to perform higher layer application software or protocol processing.

1.4 NMSI COMMUNICATIONS-ORIENTED ENVIRONMENT

When the interface to equipment or proprietary networks requires the use of standard control and data signals, the MC68302 can be programmed into the nonmultiplexed serial interface (NMSI) mode. This mode, which is available for one, two, or all three SCC ports, can be selected while the other ports use one of the multiplexed interface modes (IDL, GCI, or PCM highway).

In the example shown in Figure 1-4, one SCC channel connects through the NMSI mode to a commercial packet data network. This connection might be used for remote status monitoring or for maintenance functions for a system. The other SCC channel is used as a local synchronous channel, which could connect to another computer or subsystem. The SCP channel could then be used for local interconnection of interface chips or peripherals to the MC68302-based system.

1.5 BASIC RATE ISDN OR DIGITAL VOICE/DATA TERMINAL

A basic rate ISDN or digital voice/data terminal can be made from a chip set based on the MC68302. Refer to Figure 1-5 for an example of a basic rate ISDN voice/data terminal. In this terminal, the CP can directly support the 2B + D channels and perform either V.110 or V.120 rate adaption. The physical layer serial interface is connected to the local interconnection bus (IDL in Figure 1-5, but the GCI and PCM buses can also be supported). The system then supports one of the B channels for voice (connected directly to the physical bus). The D channel consists of one SCC port; the other B channel is used for data transfer through a second SCC port. The data can be routed to a terminal (RS-232 type) via the third SCC port in the NMSI mode. The SCP can function as a peripheral chip control channel for the IDL bus or as a general-purpose communications channel for devices using SCP-channel format.

Some physical layer devices support the signaling and framing functions of the D channel. In these cases, the D channel connects through the microprocessor interface to the physical layer device, and the remaining SCC port can be used for a second B channel to transfer data.

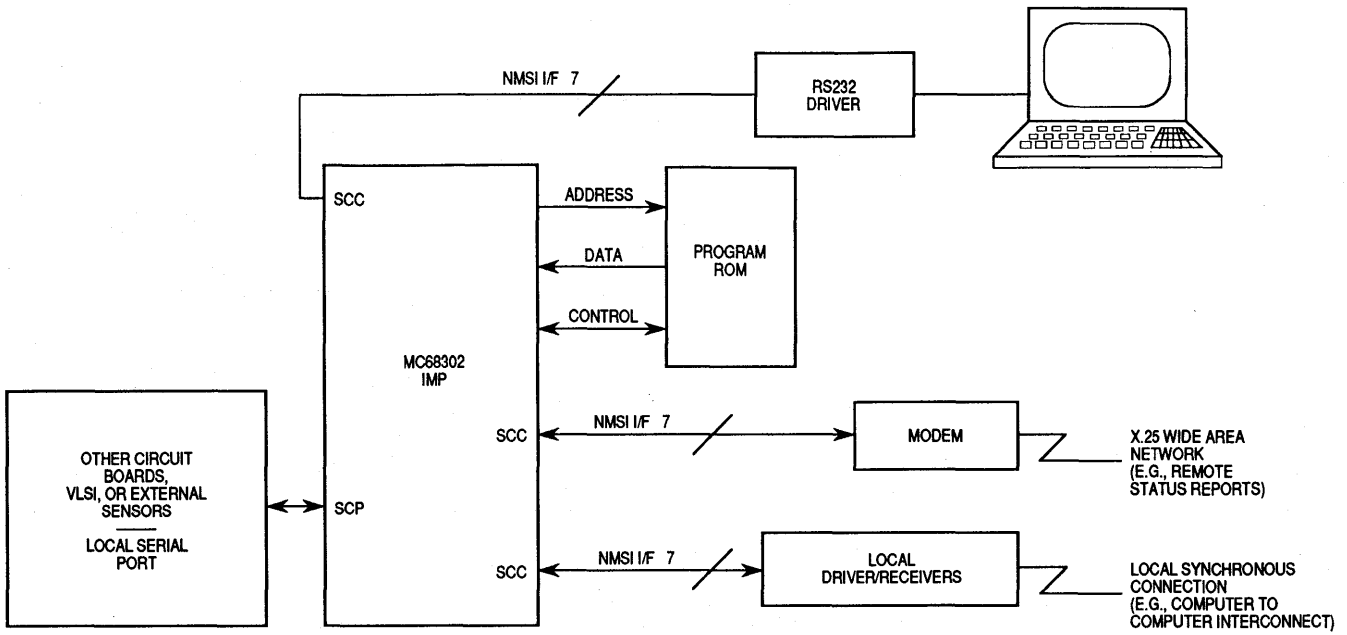


Figure 1-4. NMSI Communications-Oriented Board Design

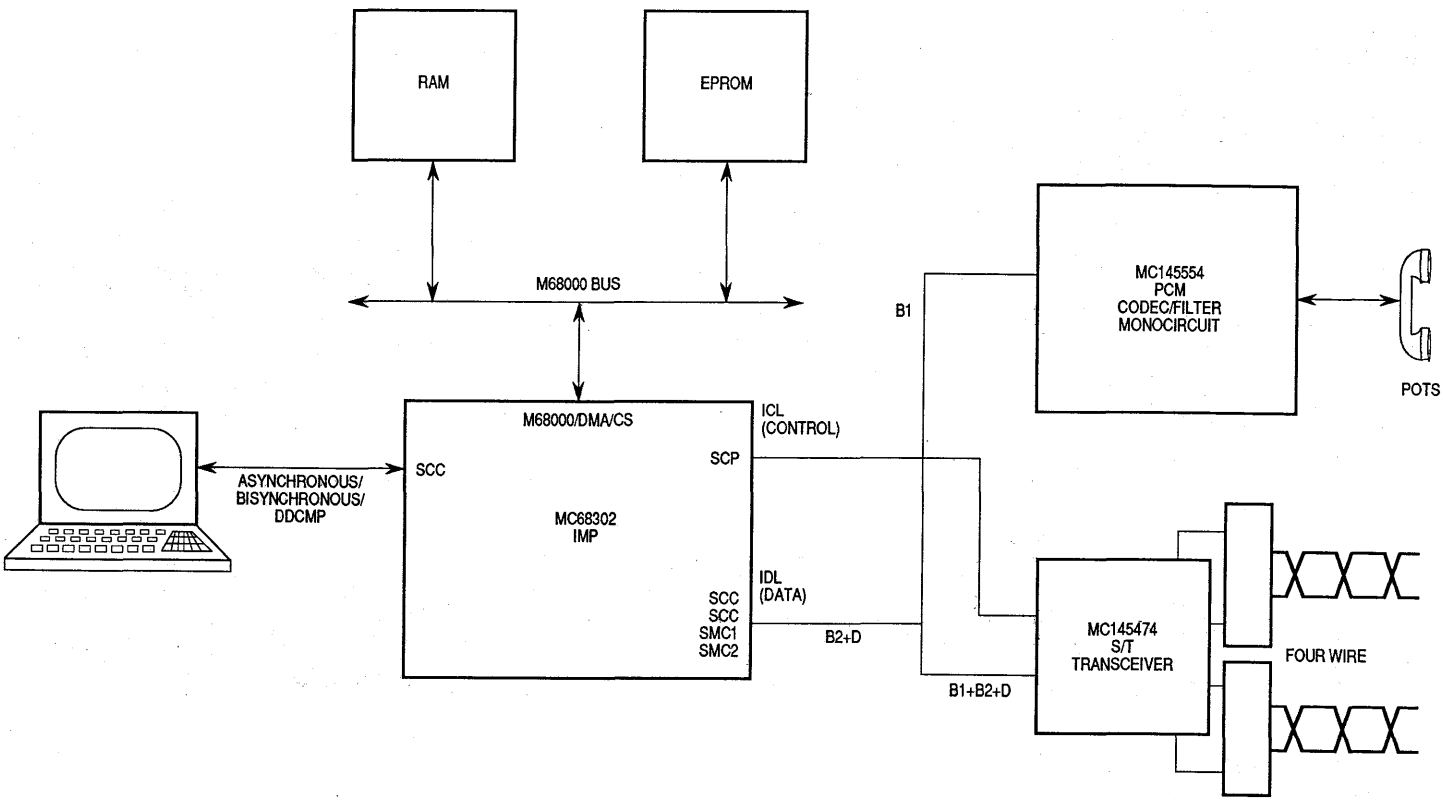


Figure 1-5. Basic Rate ISDN Voice/Data Terminal

The benefit of a local interconnection bus (see Figure 1-5) versus a microprocessor bus is a lower pin count. It is also easy to maintain this low pin-count interface between several different interface chips, such as the MC145554 PCM codec/filter monocircuit and the MC145474 S/T transceiver.

The MC68302 combines the M68000 architecture with a number of peripherals for integrated applications in communications control. The M68000 core manages the CP through the on-chip, dual-port RAM and internal registers. The base address of the dual-port RAM and internal registers is selected through the base address register. Other peripherals are also accessed and controlled through internal registers: the IDMA controller, the three timers, I/O ports, and the interrupt controller.

SECTION 2
MC68000/MC68008 CORE

The MC68302 integrates a high-speed M68000 processor with multiple communications peripherals. The provision of direct memory access (DMA) control and link layer management with the serial ports allows high throughput of data for communications-intensive applications, such as basic rate Integrated Services Digital Network (ISDN).

The MC68302 can operate either in the full MC68000 mode with a 16-bit data bus or in the MC68008 mode with an 8-bit data bus by tying the bus width (BUSW) pin low. $\overline{UDS}$/A0 functions as A0 and $\overline{LDS}$/$\overline{DS}$ functions as $\overline{DS}$ in the MC68008 mode.

NOTE

The BUSW pin is static and is not intended to be used for dynamic bus sizing. If the state of BUSW is changed during operation of the MC68302, erratic operation may occur.

Refer to the MC68000UM/AD, *M68000 8-/16-/32-Bit Microprocessors User's Manual*, for complete details of the on-chip microprocessor. Throughout this manual, references may use the notation M68000, meaning all devices belonging to this family of microprocessors, or the notation MC68000, MC68008, meaning the specific microprocessor products.

2.1 PROGRAMMING MODEL

The M68000 microprocessor executes instructions in one of two modes: user or supervisor. The user mode provides the execution environment for most of the application programs. The supervisor mode, which allows some additional instructions and privileges, is intended for use by the operating system and other system software.

Shown in Figure 2-1, the M68000 core programming model offers 16 32-bit, general-purpose registers (D7–D0, A7–A0), a 32-bit program counter (PC), and an 8-bit condition code register (CCR) when running in user space. The first eight registers (D7–D0) are used as data registers for byte (8-bit), word (16-bit), and long-word (32-bit) operations. The second set of seven registers

(A6–A0) and the stack pointer (USP in user space) may be used as software stack pointers and base address registers. In addition, the address registers may be used for word and long-word operations. All 16 registers may be used as index registers.

The supervisor's programming model includes supplementary registers, including the supervisor stack pointer (SSP) and the status register (SR) as shown in Figure 2-2. The SR contains the interrupt mask (eight levels avail-

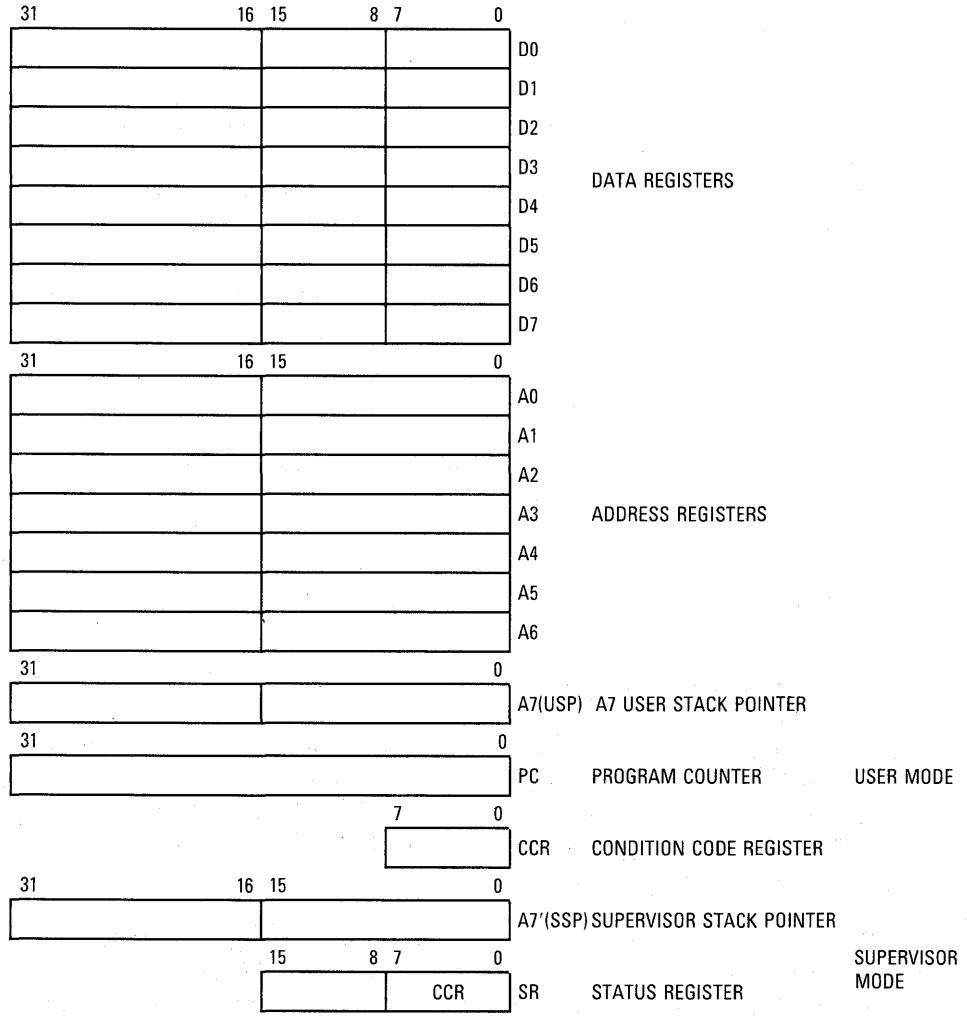


Figure 2-1. M68000 Programming Model

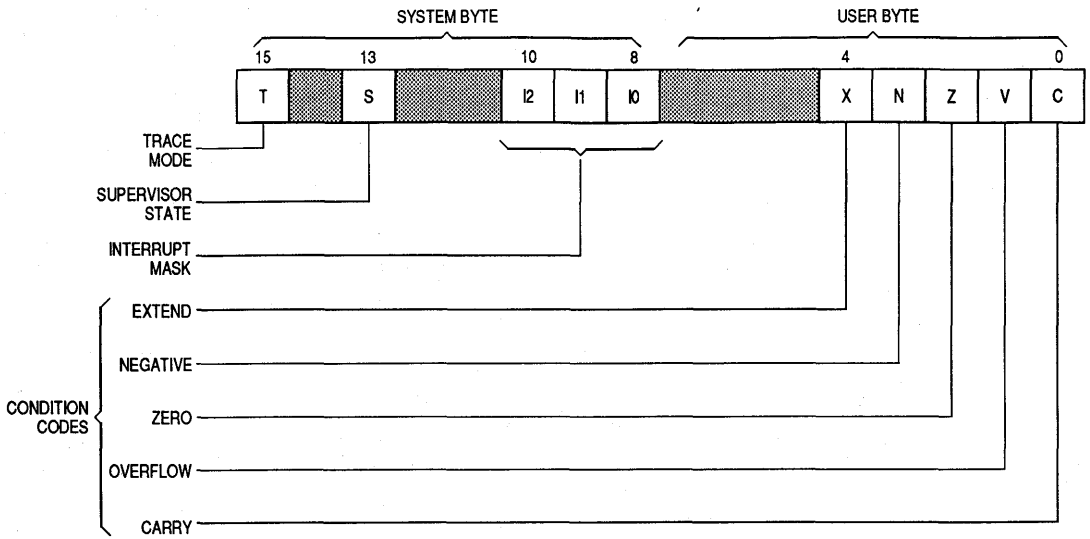


Figure 2-2. M68000 Status Register

able) as well as the following condition codes: overflow (V), zero (Z), negative (N), carry (C), and extend (X). Additional status bits indicate that the processor is in trace (T) mode and/or in a supervisor (S) state.

2.2 INSTRUCTION SET SUMMARY

The five data types supported are bits, binary-coded decimal (BCD) digits (4 bits), bytes (8 bits), words (16 bits), and long words (32 bits).

In addition, operations on other data types, such as memory addresses, status word data, etc., are provided for in the instruction set. Shown in Table 2-1, the 14 flexible addressing modes include six basic types:

- Register Direct
- Register Indirect
- Absolute
- Immediate
- Program Counter Relative
- Implied

The capability to perform postincrementing, predecrementing, offsetting, and indexing is included in the register indirect addressing modes. Program counter relative modes can also be modified via indexing and offsetting.

The M68000 instruction set is shown in Table 2-2.

Some basic instructions also have variations as shown in Table 2-3.

Special emphasis has been placed on the instruction set to simplify programming and to support structured high-level languages. With a few exceptions, each instruction operates on bytes, words, or long words, and most instructions can use any of the 14 addressing modes.

Combining instruction types, data types, and addressing modes provides over 1000 useful instructions. These instructions include signed and unsigned multiply and divide, quick arithmetic operations, BCD arithmetic, and expanded operations (through traps).

Table 2-1. M68000 Data Addressing Modes

Mode	Generation
Register Direct Addressing	
Data Register Direct	EA = Dn
Address Register Direct	EA = Dn
Absolute Data Addressing	
Absolute Short	EA = (Next Word)
Absolute Long	EA = (Next Two Words)
Program Counter Relative Addressing	
Relative with Offset	EA = (PC) + d16
Relative with Index and Offset	EA = (PC) + Xn + d8
Register Indirect Addressing	
Register Indirect	EA = (An)
Postincrement Register Indirect	EA = (An), An ← An + N
Predecrement Register Indirect	EA = ← An − N, EA = (An)
Register Indirect with Offset	EA = (An) + d16
Indexed Register Indirect with Offset	EA = (An) + (Xn) + d8
Immediate Data Addressing	
Immediate	DATA = Next Word(s)
Quick Immediate	Inherent Data
Implied Addressing	
Implied Register	EA = SR, USP, SSP, PC

NOTES:
EA = Effective Address
An = Address Register
Dn = Data Register
Xn = Address or Data Register used as an Index Register
SR = Status Register
PC = Program Counter
( ) = Contents of
d8 = 8-Bit Offset (Displacement)
d16 = 16-Bit Offset (Displacement)
N = 1 for byte, 2 for word, and 4 for long word. If An is the stack pointer and the operand size is byte, N = 2 to keep the stack pointer on a word boundary.
← = Replaces

Table 2-2. M68000 Instruction Set Summary

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
AND	Logical AND
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
Bcc	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR	Clear Operand
CMP	Compare
DBcc	Decrement and Branch Conditionally
DIVS/	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right

Mnemonic	Description
MOVE	Move Source to Destination
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NOT	One's Complement
OR	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditional
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Table 2-3. M68000 Instruction Type Variations

Instruction Type	Variation	Description
ADD	ADD	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	AND	Logical AND
	ANDI	And Immediate
	ANDI to CCR	And Immediate to Condition Codes
	ANDI to SR	And Immediate to Status Registers
CMP	CMP	Compare
	CMPA	Compare Addresses
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	EOR	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive OR Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register
MOVE	MOVE	Move Source to Destination
	MOVEA	Move Address
	MOVEC	Move Control Register
	MOVEM	Move Multiple Register
	MOVEP	Move Peripheral Data
	MOVEQ	Move Quick
	MOVE from SR	Move from Status Register
	MOVE to SR	Move to Status Register
	MOVE to CCR	Move to Condition Codes
	MOVE USP	Move User Stack Pointer
NEG	NEG	Negate
	NEGX	Negate with Extend
OR	OR	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX	Subtract with Extend

2.3 ADDRESS SPACES

The M68000 microprocessor operates in one of two privilege states: user or supervisor. The privilege state determines which operations are legal, which operations are used by the external memory management device to control and translate accesses, and which operations are used to choose between the SSP and the USP in instruction references. The M68000 address spaces are shown in Table 2-4.

Table 2-4. M68000 Address Spaces

Function Code Output			Reference Class
FC2	FC1	FC0	
0	0	0	(Unassigned)
0	0	1	User Data
0	1	0	User Program
0	1	1	(Unassigned)
1	0	0	(Unassigned)
1	0	1	Supervisor Data
1	1	0	Supervisor Program
1	1	1	Interrupt Acknowledge

The supervisor state is the highest state of privilege. For instruction execution, the supervisor state is determined by the S bit of the SR; if the S bit is asserted (high), the processor is in the supervisor state. The bus cycles generated by instructions executed in the supervisor state are classified as supervisor references. While the processor is in the supervisor privilege state, those instructions using either the system stack pointer implicitly or address register seven explicitly access the SSP.

All exception processing occurs in the supervisor state, regardless of the state of the S bit when the exception occurs. The bus cycles generated during exception processing are classified as supervisor references. All stacking operations during exception processing use the SSP.

The user state is the lower state of privilege. For instruction execution, the user state is determined by the S bit of the SR; if the S bit is negated (low), the processor is executing instructions in the user state. Most instructions execute identically in either user state or supervisor state. However, instructions having important system effects are privileged. User programs are not

permitted to execute the STOP instruction or the RESET instruction. To ensure that a user program cannot enter the supervisor state except in a controlled manner, the instructions which modify the entire SR are privileged. To aid in debugging programs to be used in operating systems, the move-to-user-stack-pointer (MOVE to USP) and move-from-user-stack-pointer (MOVE from USP) instructions are also privileged.

Once the processor is in the user state and executing instructions, only exception processing can change the privilege state. During exception processing, the current state of the S bit in the SR is saved and the S bit is asserted, putting the processor in the supervisor state. Therefore, when instruction execution resumes at the address specified to process the exception, the processor is in the supervisor privilege state. The transition from the supervisor to user state can be accomplished by any of four instructions: return from exception (RTE), move to status register (MOVE to SR), AND immediate to status register (ANDI to SR), and exclusive OR immediate to status register (EORI to SR).

2.4 EXCEPTION PROCESSING

The processing of an exception occurs in four steps, with variations for different exception causes. During the first step, a temporary copy of the SR is made, and the SR is set for exception processing. During the second step, the exception vector is determined; during the third step, the current processor context is saved. During the fourth step, a new context is obtained, and the processor switches to instruction processing.

2.4.1 Exception Vectors

Exception vectors are memory locations from which the processor fetches the address of a routine to handle that exception. All exception vectors are two words long except for the reset vector, which is four words. All exception vectors lie in the supervisor data space except for the reset vector, which is in the supervisor program space. A vector number is an 8-bit number which, when multiplied by four, gives the offset of the exception vector. Vector numbers are generated internally or externally, depending on the cause of the exception. In the case of interrupts, during the interrupt acknowledge bus cycle, a peripheral may provide an 8-bit vector number to the processor on data bus lines D7–D0. Alternatively, the peripheral may assert autovector ($\overline{\text{AVEC}}$) instead of data transfer acknowledge ($\overline{\text{DTACK}}$) to request an autovector for that priority level of interrupt. The exception vector assignments for the M68000 processor are shown in Table 2-5.

Table 2-5. M68000 Exception Vector Assignment (Sheet 1 of 2)

Vector Number	Decimal	Address Hex	Space	Assignment
0	0	000	SP	Reset: Initial SSP[2]
1	4	004	SP	Reset: Initial PC[2]
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12[1]	48	030	SD	(Unassigned, Reserved)
13[1]	52	034	SD	(Unassigned, Reserved)
14	56	038	SD	(Unassigned, Reserved)
15	60	03C	SD	Uninitialized Interrupt Vector
16–23[1]	64	040	SD	(Unassigned, Reserved)
	92	05C	SD	
24	96	060	SD	Spurious Interrupt[3]
25	100	064	SD	Level 1 Interrupt Autovector
26	104	068	SD	Level 2 Interrupt Autovector
27	108	06C	SD	Level 3 Interrupt Autovector
28	112	070	SD	Level 4 Interrupt Autovector
29	116	074	SD	Level 5 Interrupt Autovector
30	120	078	SD	Level 6 Interrupt Autovector
31	124	07C	SD	Level 7 Interrupt Autovector
32–47	128	080	SD	TRAP Instruction Vectors[4]
	188	0BC	SD	
48–63[1]	192	0C0	SD	(Unassigned, Reserved)
	255	0FF	SD	

Table 2-5. M68000 Exception Vector Assignment (Sheet 2 of 2)

Vector Number	Decimal	Address Hex	Space	Assignment
64–255	256	100	SD	User Interrupt Vectors
	1020	3FC	SD	

NOTES:
1. Vector numbers 12, 13, 16–23, and 48–63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.
2. Unlike the other vectors which only require two words, reset vector (0) requires four words and is located in the supervisor program space.
3. The spurious interrupt vector is taken when there is a bus error indication during interrupt processing.
4. TRAP #n uses vector number 32+n.

2.4.2 Exception Stacking Order

Exception processing saves the most volatile portion of the current processor context on top of the supervisor stack. This context is organized in a format called the exception stack frame. The amount and type of information saved on the stack is determined by the type of exception. The reset exception causes the M68000 to halt current execution and to read a new SSP and PC as shown in Table 2-5. A bus error or address error causes the M68000 to store the information shown in Figure 2-3. The interrupts, traps, illegal instructions, and trace stack frames are shown in Figure 2-4.

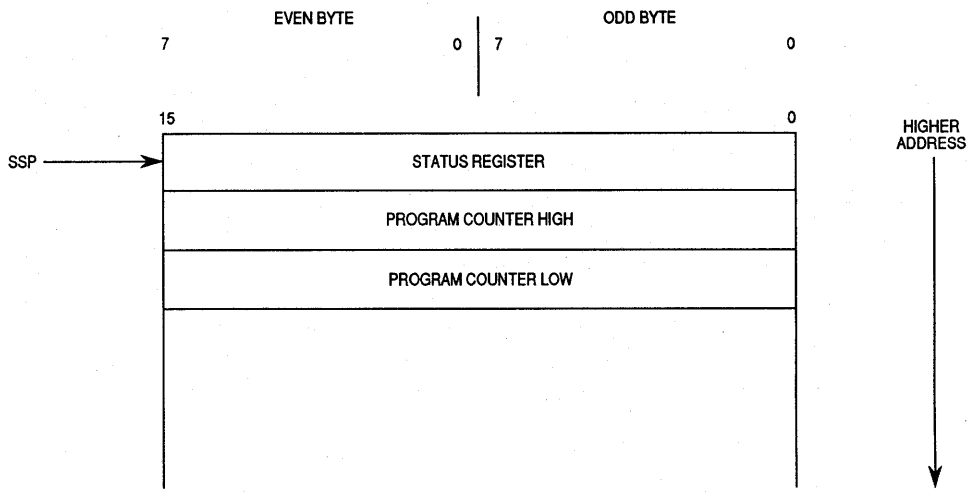


Figure 2-3. M68000 Bus/Address-Error Exception Stack Frame

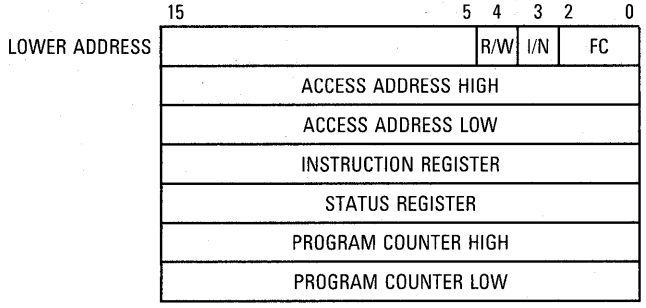


Figure 2-4. M68000 Short-Form Exception Stack Frame

2.5 INTERRUPT PROCESSING

Seven interrupt levels are provided by the M68000 core. If the IMP's interrupt controller is placed in the normal mode, six levels are available to the user. If the interrupt controller is in the dedicated mode, three levels are available to the user. In either mode, level 4 is reserved for the on-chip peripherals. Devices may be chained externally within one of the available priority levels, allowing an unlimited number of external peripheral devices to interrupt the processor. The SR contains a 3-bit mask indicating the current processor priority level. Interrupts are inhibited for all priority levels less than or equal to the current processor priority (see Figure 2-2).

An interrupt request is made to the processor by encoding the request on the interrupt request lines (normal mode) or by asserting the appropriate request line (dedicated mode). Rather than forcing immediate exception processing, interrupt requests arriving at the processor are made pending to be detected between instruction executions.

If the priority of the pending interrupt is lower than or equal to the current processor priority, execution continues with the next instruction, and the interrupt exception processing is postponed.

If the priority of the pending interrupt is greater than the current processor priority, the exception processing sequence is started. A copy of the SR is saved, the privilege state is set to supervisor state, tracing is suppressed, and the processor priority level is set to the level of the interrupt being

acknowledged. The processor fetches the vector number from the interrupting device, classifying the reference as an interrupt acknowledge on the address bus. If external logic requests automatic vectoring (via the $\overline{\text{AVEC}}$ pin), the processor internally generates a vector number determined by the interrupt level number. If external logic indicates a bus error, the interrupt is considered spurious, and the generated vector number references the spurious interrupt vector number.

2.6 M68000 SIGNAL DIFFERENCES

The MC68302 core supports two additional signals not visible on the standard M68000: $\overline{\text{IPEND}}$ and $\overline{\text{RMC}}$. The interrupt pending ($\overline{\text{IPEND}}$) signal provides information to the on-chip bus arbiter to assert the internal and external bus-clear signals, thus supporting a low-latency interrupt mechanism. $\overline{\text{IPEND}}$ is not visible externally. Asserted externally on read-modify-write cycles, the $\overline{\text{RMC}}$ signal is typically used as a bus lock to insure integrity of instructions using the read-modify-write operation. The $\overline{\text{RMC}}$ signal from the M68000 core is applied to the arbiter and can be programmed to prevent the arbiter from issuing bus grants until the completion of an MC68000-core-initiated read-modify-write cycle.

The MC68302 can be programmed to use the $\overline{\text{RMC}}$ signal to negate address strobe ($\overline{\text{AS}}$) at the end of the read portion of the cycle and assert $\overline{\text{AS}}$ at the beginning of the write portion of the cycle.

Two M6800 signals are omitted from the MC68302: valid memory address ($\overline{\text{VMA}}$) and enable (E). The valid peripheral address ($\overline{\text{VPA}}$) signal is retained, but is only used on the MC68302 as $\overline{\text{AVEC}}$ to direct the core to use an autovector during interrupt acknowledge cycles.

2.7 MC68302 IMP CONFIGURATION CONTROL

Four entries in the external M68000 exception vector table are used as addresses for internal system configuration registers. The first entry is the on-chip peripheral base address register (BAR) entry; the second is the on-chip system control register (SCR); the third and fourth entries are reserved for future use. Refer to **3.8 SYSTEM CONTROL** for the detailed description of the SCR.

NOTE

These registers are internally reset only when a total system reset occurs by the simultaneous assertion of $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$. The $\overline{\text{CS}}$ lines are not asserted on accesses to these locations. Thus, it is very

convenient to use $\overline{CS}$ lines to select external ROM/RAM that overlaps the BAR and SCR register locations, since this prevents potential bus contention. Otherwise, the IAC signal may be used.

Figure 2-5 shows all the MC68302 IMP on-chip addressable locations (except the M68000 core registers).

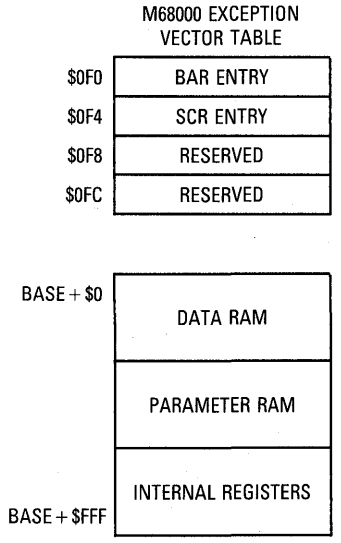


Figure 2-5. MC68302 IMP Configuration Control

The on-chip peripherals, including those peripherals in both the CP and SIB, require a 4K-byte block of address space. This 4K-byte block location is determined by writing the intended base address to the BAR in supervisor space (FC = 5). The address of the BAR entry is $0F0; however, the actual BAR is a 16-bit value within the BAR entry and is located at $0F2. The $\overline{CS}$ lines are not asserted on accesses to any on-chip peripheral locations.

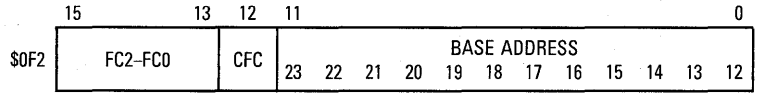
After a total system reset, the on-chip peripheral base address is undefined, and it is not possible to access the on-chip peripherals at any address until BAR is written. The BAR and the SCR can always be accessed at their fixed addresses.

NOTE

In 8-bit system bus operation, IMP accesses are not possible until the low byte of the BAR is written. Since the MOVE.W instruction writes the high byte followed by the low byte, this instruction guarantees the entire word is written.

Do not assign other devices on the system bus an address that falls within the address range of the peripherals defined by the BAR. If this happens, $\overline{\text{BERR}}$ is generated (if the address decode conflict enable (ADCE) bit is set) and the address decode conflict (ADC) bit in the SCR is set.

The BAR is a 16-bit, memory-mapped, read-write register consisting of the high address bits, the compare function code bit, and the function code bits. Upon a total system reset, its value may be read as $BFFF. The address of this register is fixed at $0F2 in supervisor space.



Bits 15–13 — FC2–FC0

The FC2–FC0 field is contained in bits 15–13 of the BAR. These bits are used to set the MC68302 address space. The address compare logic uses these bits, dependent upon the CFC bit, to cause an address match within its address space.

CFC — Compare Function Code

0 = The FC bits in the BAR are ignored. Accesses to the IMP occur without comparing the FC bits.

1 = The FC bits in the BAR are compared. The address space compare logic uses the FC bits to detect address matches.

Bits 11–0 — Base Address

The high address field is contained in bits 11–0 of the BAR. These bits are used to set the starting address of the dual-port RAM. The address compare logic uses only the most significant bits to cause an address match within its block size.

NOTE

Do not assign the MC68302 internal address space into the M68000 core interrupt acknowledge space (FC2–FC0 = 7).

2.8 MC68302 MEMORY MAP

The following tables show the additional registers added to the M68000 to make up the MC68302. All of the registers are memory-mapped. Four entries in the M68000 exception vectors table (located in low RAM) are reserved for addresses of system configuration registers (see Table 2-6) that reside on-chip. These registers have fixed addresses of $0F0–$0FF. All other on-chip peripherals occupy a 4K-byte relocatable address space. When an on-chip register or peripheral is accessed, the internal access (IAC) pin is asserted.

Table 2-6. System Configuration Registers

Address	Name	Width	Description	Reset Value
$0F0	RES	16	Reserved	
$0F2*	BAR	16	Base Address Register	BFFF
$0F4*	SCR	32	System Control Register	0000 0F00
$0F8	RES	32	Reserved	
$0FC	RES	32	Reserved	

*Reset only upon a total system reset.

The internal 1176-byte dual-port RAM has 576 bytes of system RAM (see Table 2-7) and 576 bytes of parameter RAM (see Table 2-8).

Table 2-7. System RAM

Address	Width	Block	Description
Base + 000 • • • Base + 23F	576 Bytes	RAM	User Data Memory
Base + 240 • • • Base + 3FF			Reserved

The parameter RAM contains the buffer descriptors for each of the three SCC channels, the SCP, and the two SMC channels. The memory structures of the three SCC channels are identical. When any SCC, SCP, or SMC channel buffer descriptors or parameters are not used, their parameter RAM area can be used for additional memory. For detailed information about the use of the buffer descriptors and protocol parameters in a specific protocol, see **4.5 SERIAL COMMUNICATION CONTROLLERS (SCCs)**. Base + 67E contains the

MC68302 revision number. Revision A parts (mask 1B14M) correspond to the value $0001. Revision B parts (mask 2B14M described in this manual) correspond to the value $0002.

Table 2-8. Parameter RAM (Sheet 1 of 2)

Address	Width	Block	Description
Base + 400	4 Word	SCC1	Rx BD 0
Base + 408	4 Word	SCC1	Rx BD 1
Base + 410	4 Word	SCC1	Rx BD 2
Base + 418	4 Word	SCC1	Rx BD 3
Base + 420	4 Word	SCC1	Rx BD 4
Base + 428	4 Word	SCC1	Rx BD 5
Base + 430	4 Word	SCC1	Rx BD 6
Base + 438	4 Word	SCC1	Rx BD 7
Base + 440	4 Word	SCC1	Tx BD 0
Base + 448	4 Word	SCC1	Tx BD 1
Base + 450	4 Word	SCC1	Tx BD 2
Base + 458	4 Word	SCC1	Tx BD 3
Base + 460	4 Word	SCC1	Tx BD 4
Base + 468	4 Word	SCC1	Tx BD 5
Base + 470	4 Word	SCC1	Tx BD 6
Base + 478	4 Word	SCC1	Tx BD 7
Base + 480		SCC1	Specific Protocol Parameters
•			
•			
•			
Base + 4BF		SCC1	
Base + 4C0			Reserved
•			
•			
•			
Base + 4FF			
Base + 500	4 Word	SCC2	Rx BD 0
Base + 508	4 Word	SCC2	Rx BD 1
Base + 510	4 Word	SCC2	Rx BD 2
Base + 518	4 Word	SCC2	Rx BD 3
Base + 520	4 Word	SCC2	Rx BD 4
Base + 528	4 Word	SCC2	Rx BD 5
Base + 530	4 Word	SCC2	Rx BD 6
Base + 538	4 Word	SCC2	Rx BD 7
Base + 540	4 Word	SCC2	Tx BD 0
Base + 548	4 Word	SCC2	Tx BD 1
Base + 550	4 Word	SCC2	Tx BD 2
Base + 558	4 Word	SCC2	Tx BD 3
Base + 560	4 Word	SCC2	Tx BD 4
Base + 568	4 Word	SCC2	Tx BD 5
Base + 570	4 Word	SCC2	Tx BD 6
Base + 578	4 Word	SCC2	Tx BD 7
Base + 580		SCC2	Specific Protocol Parameters
•			
•			
•			
Base + 5BF		SCC2	

Table 2-8. Parameter RAM Sheet (2 of 2)

Address	Width	Block	Description
Base + 5C0 • • • Base + 5FF			Reserved
Base + 600	4 Word	SCC3	Rx BD 0
Base + 608	4 Word	SCC3	Rx BD 1
Base + 610	4 Word	SCC3	Rx BD 2
Base + 618	4 Word	SCC3	Rx BD 3
Base + 620	4 Word	SCC3	Rx BD 4
Base + 628	4 Word	SCC3	Rx BD 5
Base + 630	4 Word	SCC3	Rx BD 6
Base + 638	4 Word	SCC3	Rx BD 7
Base + 640	4 Word	SCC3	Tx BD 0
Base + 648	4 Word	SCC3	Tx BD 1
Base + 650	4 Word	SCC3	Tx BD 2
Base + 658	4 Word	SCC3	Tx BD 3
Base + 660	3 Word	SMC	Reserved
Base + 666	Word	SMC1	Rx BD
Base + 668	Word	SMC1	Tx BD
Base + 66A	Word	SMC2	Rx BD
Base + 66C	Word	SMC2	Tx BD
Base + 66E*	6 Word	SMC1–SMC2	Internal Use
Base + 67A	Word	SCP	Rx/Tx BD
Base + 67C	Word	SCC1–SCC3	BERR Channel Number
Base + 67E*	Word	CP	MC68302 Revision Number
Base + 680 • • • Base + 6BF		SCC3 SCC3	Specific Protocol Parameters
Base + 6C0 • • • Base + 7FF			Reserved

NOTE: Tx BD 4, 5, 6, and 7 are not initially available to SCC3. (See **4.5.5 Buffer Descriptors Table.**)
*Modified by the CP after a CP or system reset.

In addition to the internal dual-port RAM, there are a number of internal registers to support the functions of the various M68000 core peripherals. The internal registers (see Table 2-9) are memory-mapped registers offset from the BAR pointer and are located on the internal M68000 bus.

NOTE

All undefined and reserved bits within registers and parameter RAM values written by the user in a given application should be written with zero to allow for future enhancements to the device.

Table 2-9. Internal Registers (Sheet 1 of 2)

Address	Name	Width	Block	Description	Reset Value
Base+800	RES	16	IDMA	Reserved	
Base+802	CMR	16	IDMA	Channel Mode Register	0000
Base+804	SAPR	32	IDMA	Source Address Pointer	0000 0000
Base+808	DAPR	32	IDMA	Destination Address Pointer	0000 0000
Base+80C	BCR	16	IDMA	Byte Count Register	0000
Base+80E	CSR	8	IDMA	Channel Status Register	00
Base+80F	RES	8	IDMA	Reserved	
Base+810	FCR	8	IDMA	Function Code Register	00
Base+811	RES	8	IDMA	Reserved	
Base+812*	GIMR	16	Int Cont	Global Interrupt Mode Register	0000
Base+814	IPR	16	Int Cont	Interrupt Pending Register	0000
Base+816	IMR	16	Int Cont	Interrupt Mask Register	0000
Base+818	ISR	16	Int Cont	In-Service Register	0000
Base+81A	RES	16	Int Cont	Reserved	
Base+81C	RES	16	Int Cont	Reserved	
Base+81E*	PACNT	16	PIO	Port A Control Register	0000
Base+820*	PADDR	16	PIO	Port A Data Direction Register	0000
Base+822*	PADAT	16	PIO	Port A Data Register	† XXXX
Base+824*	PBCNT	16	PIO	Port B Control Register	0080
Base+826*	PBDDR	16	PIO	Port B Data Direction Register	0000
Base+828*	PBDAT	16	PIO	Port B Data Register	† XXXX
Base+82A	RES	16	PIO	Reserved	
Base+82C	RES	16	CS	Reserved	
Base+82E	RES	16	CS	Reserved	
Base+830*	BR0	16	CS0	Base Register 0	C001
Base+832*	OR0	16	CS0	Option Register 0	DFFD
Base+834*	BR1	16	CS1	Base Register 1	C000
Base+836*	OR1	16	CS1	Option Register 1	DFFD
Base+838*	BR2	16	CS2	Base Register 2	C000
Base+83A*	OR2	16	CS2	Option Register2	DFFD
Base+83C*	BR3	16	CS3	Base Register 3	C000
Base+83E*	OR3	16	CS3	Option Register 3	DFFD
Base+840	TMR1	16	Timer	Timer Unit 1 Mode Register	0000
Base+842	TRR1	16	Timer	Timer Unit 1 Reference Reg.	FFFF
Base+844	TCR1	16	Timer	Timer Unit 1 Capture Register	0000
Base+846	TCN1	16	Timer	Timer Unit 1 Counter	0000
Base+848	RES	8	Timer	Reserved	
Base+849	TER1	8	Timer	Timer Unit 1 Event Register	00
Base+84A	WRR	16	WD	Watchdog Reference Register	FFFF
Base+84C	WCN	16	WD	Watchdog Counter	0000
Base+84E	RES	16	Timer	Reserved	
Base+850	TMR2	16	Timer	Timer Unit 2 Mode Register	0000
Base+852	TRR2	16	Timer	Timer Unit 2 Reference Reg.	FFFF
Base+854	TCR2	16	Timer	Timer Unit 2 Capture Register	0000
Base+856	TCN2	16	Timer	Timer Unit 2 Counter	0000
Base+858	RES	8	Timer	Reserved	
Base+859	TER2	8	Timer	Timer Unit 2 Event Register	00
Base+85A	RES	16	Timer	Reserved	
Base+85C	RES	16	Timer	Reserved	
Base+85E	RES	16	Timer	Reserved	
Base+860	CR	8	CP	Command Register	0000

Table 2-9. Internal Registers (Sheet 2 of 2)

Address	Name	Width	Block	Description	Reset Value
Base + 861 • • • Base + 87F				Reserved	
Base + 880	RES	16	SCC1	Reserved	
Base + 882	SCON1	16	SCC1	SCC1 Configuration Register	0004
Base + 884	SCM1	16	SCC1	SCC1 Mode Register	0000
Base + 886	DSR1	16	SCC1	SCC1 Data Sync. Reg.	7E7E
Base + 888	SCCE1	8	SCC1	SCC1 Event Register	00
Base + 889	RES	8	SCC1	Reserved	
Base + 88A	SCCM1	8	SCC1	SCC1 Mask Register	00
Base + 88B	RES	8	SCC1	Reserved	
Base + 88C	SCCS1	8	SCC1	SCC1 Status Register	00
Base + 88D	RES	8	SCC1	Reserved	
Base + 88E	RES	16	SCC1	Reserved	
Base + 890	RES	16	SCC2	Reserved	
Base + 892	SCON2	16	SCC2	SCC2 Configuration Register	0004
Base + 894	SCM2	16	SCC2	SCC2 Mode Register	0000
Base + 896	DSR2	16	SCC2	SCC2 Data Sync. Reg.	7E7E
Base + 898	SCCE2	8	SCC2	SCC2 Event Register	00
Base + 899	RES	8	SCC2	Reserved	
Base + 89A	SCCM2	8	SCC2	SCC2 Mask Register	00
Base + 89B	RES	8	SCC2	Reserved	
Base + 89C	SCCS2	8	SCC2	SCC2 Status Register	00
Base + 89D	RES	8	SCC2	Reserved	
Base + 89E	RES	16	SCC2	Reserved	
Base + 8A0	RES	16	SCC3	Reserved	
Base + 8A2	SCON3	16	SCC3	SCC3 Configuration Register	0004
Base + 8A4	SCM3	16	SCC3	SCC3 Mode Register	0000
Base + 8A6	DSR3	16	SCC3	SCC3 Data Sync. Reg.	7E7E
Base + 8A8	SCCE3	8	SCC3	SCC3 Event Register	00
Base + 899	RES	8	SCC3	Reserved	
Base + 8AA	SCCM3	8	SCC3	SCC3 Mask Register	00
Base + 8AB	RES	8	SCC3	Reserved	
Base + 8AC	SCCS3	8	SCC3	SCC3 Status Register	00
Base + 8AD	RES	8	SCC3	Reserved	
Base + 8AE	RES	16	SCC3	Reserved	
Base + 8B0	SPMODE	16	SCM	SCP, SMC Mode and Clock Control Register	0000
Base + 8B2*	SIMASK	16	SI	Serial Interface Mask Register	FFFF
Base + 8B4*	SIMODE	16	SI	Serial Interface Mode Register	0000
Base + 8B6 • • • Base + FFF				Reserved	

*Reset only upon a total system reset.
†The output latches are undefined at total system reset.

SECTION 3
SYSTEM INTEGRATION BLOCK (SIB)

The MC68302 contains an extensive SIB that simplifies the job of both the hardware and software designer. The independent direct memory access (DMA) controller relieves the hardware designer of the extra effort and board logic needed to connect an external DMA controller. The interrupt controller can be used in a dedicated mode to generate interrupt acknowledge signals without external logic. Also, the chip-select signals and wait-state logic eliminate the need to generate chip-select signals externally. The three timers simplify control and improve reliability. These and other features in the SIB conserve board space and cost, decreasing the amount of time needed to develop hardware.

The SIB includes the following functions:

- IDMA Controller with Three Handshake Signals: $\overline{\text{DREQ}}$, $\overline{\text{DACK}}$, and $\overline{\text{DONE}}$
- Interrupt Controller with Two Modes of Operation
- Parallel Input/Output (I/O) Ports, Some with Interrupt Capability
- On-Chip 1152-Byte Dual-Port RAM
- Three Timers Including a Watchdog Timer
- Four Programmable Chip-Select Lines with Wait-State Generator Logic
- On-Chip Clock Generator with Output Signal
- System Control
  - System Status and Control Logic
  - Disable CPU Logic (M68000)
  - Bus Arbitration Logic with Low-Interrupt Latency Support
  - Hardware Watchdog for Monitoring Bus Activity
  - Low-Power (Standby) Modes
  - Freeze Control for Debugging
- DRAM Refresh Controller

3.1 DMA CONTROL

The IMP includes seven on-chip DMA channels, six serial DMA (SDMA) channels for the three serial communications controllers (SCCs), and one IDMA. The SDMA channels are discussed in **4.2 SDMA CHANNELS**. The IDMA is discussed in the following paragraphs.

The one general-purpose IDMA controller can operate in different modes of data transfer as programmed by the user. The IDMA is capable of transferring data between any combination of memory and I/O. In addition, data may be transferred in either byte or word quantities, and the source and destination addresses may be either odd or even. Note that the chip select and wait state generation logic on the MC68302 may be used with the IDMA, if desired.

Every IDMA cycle requires between two and four bus cycles, depending on the address boundary and transfer size. Each bus cycle is a standard M68000-type read or write cycle. If both the source and destination addresses are even, the IDMA fetches one word of data and immediately deposits it. If either the source or destination address begins on an odd boundary, the transfer is handled differently. For example, if the source address starts on an odd boundary and the destination address is even, the IDMA reads one byte from the source, then reads the second byte from the source, and finally stores the word in a single access. If the source is even and the destination odd, then the IDMA will read one word from the source and store it in two consecutive cycles. If both the source and destination are odd, the IDMA performs two read byte cycles followed by two write byte cycles until the transfer is complete.

The IDMA controller block diagram is shown in Figure 3-1.

3.1.1 Key Features

The IDMA has the following key features:

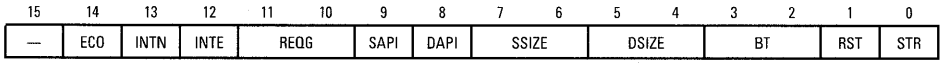
- Two Address Pointers and One Counter
- Support of Memory-to-Memory, Peripheral-to-Memory, and Memory-to-Peripheral Data Transfers
- Three I/O Lines, $\overline{\text{DREQ}}$, $\overline{\text{DACK}}$ and $\overline{\text{DONE}}$, for Externally Requested Data Transfers
- Asynchronous M68000 Bus Structure with 24-Bit Address and 8-Bit or 16-Bit Data Bus

- Support for Data Blocks Located at Even or Odd Addresses
- Packing and Unpacking of Operands
- Fast Transfer Rates: Up to 4M bytes/second at 16 MHz with No Wait States
- Full Support of All M68000 Bus Exceptions: Halt, Bus Error, Reset, and Retry
- Flexible Request Generation:
  Internal, Maximum Rate (One Burst)
  Internal, Limited Rate (Limited Burst Bandwidth)
  External, Burst ($\overline{\text{DREQ}}$ Level Sensitive)
  External, Cycle Steal ($\overline{\text{DREQ}}$ Edge Sensitive)

3.1.2 IDMA Registers

The IDMA has six registers that define its specific operation. These registers include a 32-bit source address pointer register (SAPR), a 32-bit destination address pointer register (DAPR), an 8-bit function code register (FCR), a 16-bit byte count register (BCR), a 16-bit channel mode register (CMR), and an 8-bit channel status register (CSR). These registers provide the addresses, transfer count, and configuration information necessary to set up a transfer. They also provide a means of controlling the IDMA and monitoring its status. All registers can be modified by the M68000 core. The IDMA also includes another 16-bit register, the data holding register (DHR), which is not accessible to the M68000 core and is used by the IDMA for temporary data storage.

3.1.2.1 CHANNEL MODE REGISTER (CMR). The CMR, a 16-bit register, is reset to $0000.



Bit 15 — Reserved for future use.

ECO — External Control Option
- 0 = If the request generation is programmed to be external, the control signals ($\overline{\text{DREQ}}$, $\overline{\text{DACK}}$, and $\overline{\text{DONE}}$) are used in the source (read) portion of the transfer since the peripheral is the source.
- 1 = If the request generation is programmed to be external, the control signals ($\overline{\text{DREQ}}$, $\overline{\text{DACK}}$, and $\overline{\text{DONE}}$) are used in the destination (write) portion of the transfer since the peripheral is the destination.

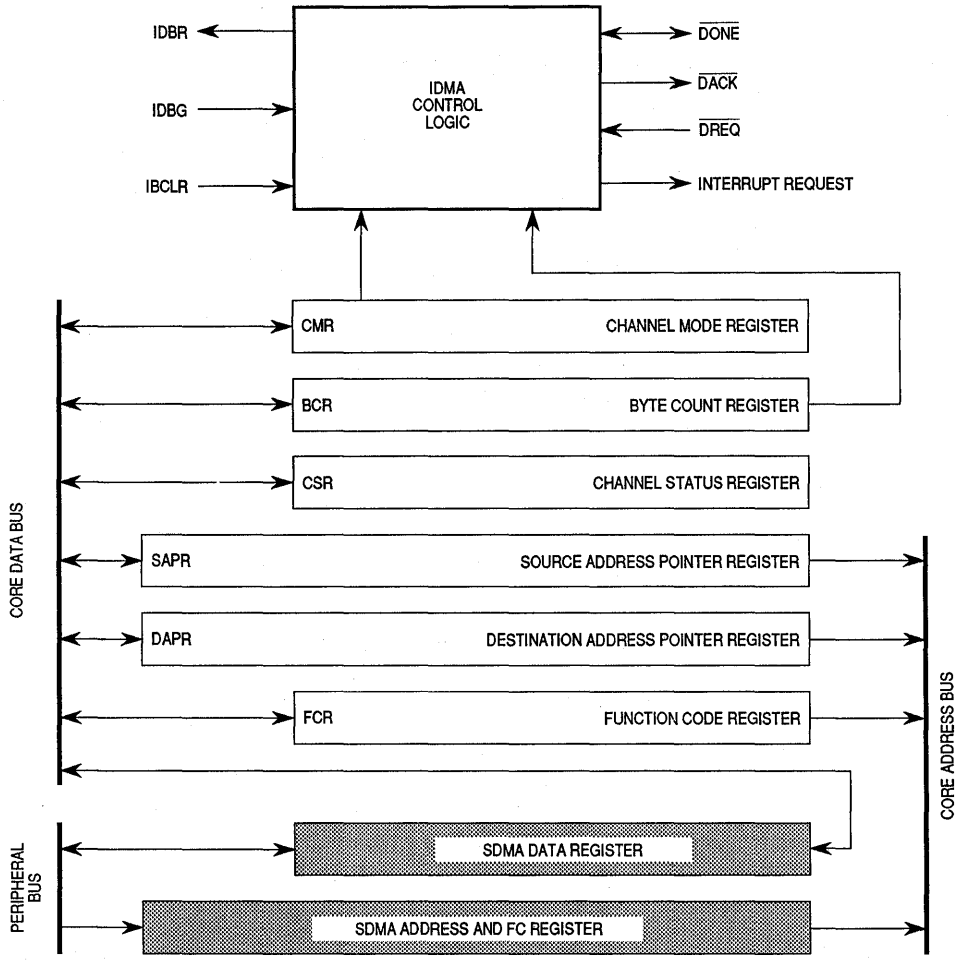


Figure 3-1. IDMA Controller Block Diagram

INTN — Interrupt Normal

0 = When the channel has completed an operand transfer without error conditions as indicated by $\overline{\text{DONE}}$, the channel does not generate an interrupt request to the IMP interrupt controller. The DONE bit remains set in the CSR.

1 = When the channel has completed an operand transfer without error conditions as indicated by $\overline{\text{DONE}}$, the channel generates an interrupt request to the IMP interrupt controller and sets DONE in the CSR.

NOTE

An interrupt will only be generated if the IDMA bit is set in the interrupt mask register (IMR).

INTE — Interrupt Error

0 = If a bus error occurs during an operand transfer either on the source read (BES) or the destination write (BED), the channel does not generate an interrupt to the IMP interrupt controller. The appropriate bit remains set in the CSR.

1 = If a bus error occurs during an operand transfer either on BES or BED, the channel generates an interrupt to the IMP interrupt controller and sets the appropriate bit (BES or BED) in the CSR.

NOTE

An interrupt will only be generated if the IDMA bit is set in the IMR.

REQG — Request Generation

The following decode shows the definitions for the REQG bits:

00 = Internal request at limited rate (limited burst bandwidth) set by burst transfer (BT) bits

01 = Internal request at maximum rate (one burst)

10 = External request burst transfer mode ($\overline{\text{DREQ}}$ level sensitive)

11 = External request cycle steal ($\overline{\text{DREQ}}$ edge sensitive)

SAPI — Source Address Pointer (SAP) Increment

0 = SAP is not incremented after each transfer.

1 = SAP is incremented by one or two after each transfer, according to the source size (SSIZE) bits and the starting address.

DAPI — Destination Address Pointer (DAP) Increment

0 = DAP is not incremented after each transfer.

1 = DAP is incremented by 1 or 2 after each transfer, according to the destination size (DSIZE) bits and the starting address.

SSIZE — Source Size

The following decode shows the definitions for the SSIZE bits.

00 = Reserved

01 = Byte

10 = Word

11 = Reserved

DSIZE — Destination Size

The following decode shows the definitions for the DSIZE bits.

00 = Reserved
01 = Byte
10 = Word
11 = Reserved

BT — Burst Transfer

The BT bits control the maximum percentage of the M68000 bus that the IDMA can use during each 1024 clock cycle period following the enabling of the IDMA. The IDMA runs for a consecutive number of cycles up to its burst transfer percentage if bus clear ($\overline{BCLR}$) is not asserted and the BCR is greater than zero. The following decode shows these percentages.

00 = IDMA gets up to 75% of the bus bandwidth.
01 = IDMA gets up to 50% of the bus bandwidth.
10 = IDMA gets up to 25% of the bus bandwidth.
11 = IDMA gets up to 12.5% of the bus bandwidth.

NOTE

These percentages are valid only when using internal limited request generation (REQG = 00).

RST — Software Reset

This bit will reset the IDMA to the same state as an external reset. The IDMA clears RST when the reset is complete.

0 = Normal operation
1 = The channel aborts any external pending or running bus cycles and terminates channel operation. Setting RST clears all bits in the CSR and CMR.

STR — Start Operation

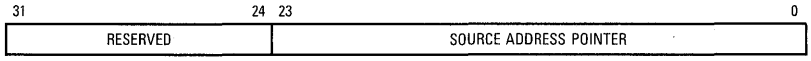
This bit starts the IDMA transfer if the REQG bits are programmed for an internal request. If the REQG bits are programmed for an external request, this bit must be set before the IDMA will recognize the first request on the $\overline{DREQ}$ input.

0 = Stop channel; clearing this bit will cause the IDMA to stop transferring data at the end of the current operand transfer. The IDMA internal state is not altered.
1 = Start channel; setting this bit will allow the IDMA to start (or continue if previously stopped) transferring data.

NOTE

STR is cleared automatically when the transfer is complete.

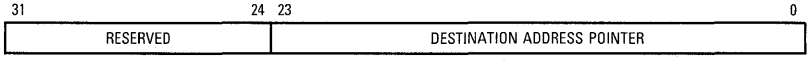
3.1.2.2 SOURCE ADDRESS POINTER REGISTER (SAPR). The SAPR is a 32-bit register.



The SAPR contains 24 (A23–A0) address bits of the source operand used by the IDMA to access memory or memory-mapped peripheral controller registers. During the IDMA read cycle, the address on the master address bus is driven from this register. The SAPR may be programmed by the SAPI bit to be incremented or remain constant after each operand transfer.

The register is incremented using unsigned arithmetic and will roll over if an overflow occurs. For example, if a register contains $00FFFFFF and is incremented by one, it will roll over to $00000000. This register can be incremented by one or two, depending on the SSIZE bit and the starting address in this register.

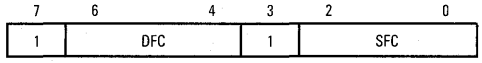
3.1.2.3 DESTINATION ADDRESS POINTER REGISTER (DAPR). The DAPR is a 32-bit register.



The DAPR contains 24 (A23–A0) address bits of the destination operand used by the IDMA to access memory or memory-mapped peripheral controller registers. During the IDMA write cycle, the address on the master address bus is driven from this register. The DAPR may be programmed by the DAPI bit to be incremented or remain constant after each operand transfer.

The register is incremented using unsigned arithmetic and will roll over if overflow occurs. For example, if a register contains $00FFFFFF and is incremented by one, it will roll over to $00000000. This register can be incremented by one or two depending on the DSIZE bit and the starting address.

3.1.2.4 FUNCTION CODE REGISTER (FCR). The FCR is an 8-bit register.

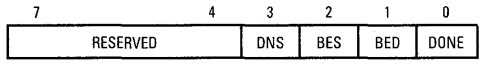


The SFC and the DFC bits define the source and destination function code values that are output by the IDMA and the appropriate address registers during an IDMA bus cycle. The address space on the function code lines may

be used by an external memory management unit (MMU) or other memory-protection device to translate the IDMA logical addresses to proper physical addresses. The function code value programmed into the FCR is placed on pins FC2–FC0 during a bus cycle to further qualify the address bus value.

3.1.2.5 BYTE COUNT REGISTER (BCR). This 16-bit register specifies the amount of data to be transferred by the IDMA; up to 64K bytes (BCR = 0) is permitted. This register is decremented once for each byte transferred successfully. BCR may be even or odd as desired. DMA activity will terminate as soon as this register reaches zero. Thus, an odd number of bytes may be transferred in a 16-bit operand scenario.

3.1.2.6 CHANNEL STATUS REGISTER (CSR). The CSR is an 8-bit register used to report events recognized by the IDMA controller. On recognition of an event, the IDMA sets its corresponding bit in the CSR (regardless of the INTE and INTN bits in the CMR). The CSR is a memory-mapped register which may be read at any time. A bit is reset by writing a one and is left unchanged by writing a zero. More than one bit may be reset at a time, and the register is cleared by reset.



Bits 7–4 — These bits are reserved for future use.

DNS — Done Not Synchronized

This bit is set if operand packing is performed between 16-bit memory and an 8-bit peripheral and the $\overline{\text{DONE}}$ signal is asserted as an input to the IDMA (i.e., by the peripheral) during the first access of the 8-bit peripheral. In such a case, the IDMA will still attempt to finish the second access of the 8-bit peripheral even though $\overline{\text{DONE}}$ has been asserted (the access could be blocked with external logic); however, the DNS bit will be set to signify this condition. DNS will not be set if the transfer is terminated by an odd byte count; since in this case, the exact number of requested bytes will be transferred by the IDMA.

BES — Bus Error Source

This bit indicates that the IDMA channel terminated with an error returned during the read cycle. The channel terminates the IDMA operation without setting DONE. BES is cleared by writing a one or by setting RST in the CMR. Writing a zero has no affect on BES.

BED — Bus Error Destination

This bit indicates that the IDMA channel terminated with an error during the write cycle. The channel terminates the IDMA operation without setting DONE. BED is cleared by writing a one or by setting RST in the CMR. Writing a zero has no affect on BED.

DONE — Normal Channel Transfer Done

This bit indicates that the IDMA channel has terminated normally. Normal channel termination is defined as 1) having decremented the BCR to zero with no errors occurring during any IDMA transfer bus cycle or 2) by the external peripheral asserting $\overline{\text{DONE}}$ with no errors occurring during any IDMA transfer bus cycle. DONE will not be set if the channel terminates due to an error. DONE is cleared by writing a one or by a software RST in the CMR. Writing a zero has no affect on this bit.

3.1.3 Interface Signals

The IDMA channel has three dedicated control signals: DMA request ($\overline{\text{DREQ}}$), DMA acknowledge ($\overline{\text{DACK}}$), and end of IDMA transfer ($\overline{\text{DONE}}$). The IDMA's use of the bus arbitration signals is described in **3.1.6 DMA Bus Arbitration.** The peripheral used with these signals may be either a source or a destination of the transfers.

3.1.3.1 $\overline{\text{DREQ}}$ and $\overline{\text{DACK}}$. These are handshake signals between the peripheral requiring service and the IMP. When the peripheral requires IDMA service, it asserts $\overline{\text{DREQ}}$, and the IMP begins the IDMA process. When the IDMA service is in progress, $\overline{\text{DACK}}$ is asserted during accesses to the device. These signals are not used when the IDMA is programmed to internal request modes.

3.1.3.2 $\overline{\text{DONE}}$. This bidirectional signal is used to indicate the last IDMA transfer. With internal request modes, the IDMA activates $\overline{\text{DONE}}$ as an output during the last IDMA bus cycle. If $\overline{\text{DONE}}$ is externally asserted during internal request modes, the IDMA transfer is terminated. With external request modes, $\overline{\text{DONE}}$ may be used as an input to the IDMA controller indicating that the device being serviced requires no more transfers and that the transmission is to be terminated. $\overline{\text{DONE}}$ is an output if the transfer count is exhausted.

3.1.4 IDMA Operational Description

Every IDMA operation involves the following steps: IDMA channel initialization, data transfer, and block termination. In the initialization phase, the M68000 core (or external processor) loads the registers with control information, address pointers and transfer count, and then starts the channel. In the transfer phase, the IDMA accepts requests for operand transfers and provides addressing and bus control for the transfers. The termination phase occurs when the operation is complete and the IDMA interrupts the M68000 core, if interrupts are enabled.

3.1.4.1 CHANNEL INITIALIZATION. To start a block transfer operation, the M68000 core must initialize IDMA registers with information describing the data block, device type, request generation method, and other special control options. See **3.1.2 IDMA Registers** and **3.1.5 IDMA Programming** for further details.

3.1.4.2 DATA TRANSFER. The IDMA supports dual address transfers only. Thus, each operand transfer consists of a source operand read and a destination operand write. The source operand is read from the address contained in the SAPR into the DHR. When the source and destination operand sizes differ, the operand read may take up to two bus cycles to complete. The operand is then written to the address contained in the DAPR. Again, this transfer may be up to two bus cycles long. In this manner, various combinations of peripheral, memory, and operand sizes may be used.

NOTE

When the SAPR and DAPR are programmed not to increment and the bus width is 16 bits, the SAPR and DAPR addresses must be even.

Source Operand Read

During this cycle, the SAPR drives the address bus, the FCR drives the source function codes, and the CMR drives the size control. The data is read from memory or the peripheral and placed temporarily into the data holding register (DHR) when the bus cycle is terminated with $\overline{\text{DTACK}}$. When the complete operand has been read, the SAPR is incremented by one or two, depending on the address and size information. See **3.1.2.2 SOURCE ADDRESS POINTER REGISTER (SAPR)** for more details.

Destination Operand Write

During this cycle, the data in DHR is written to the device or memory selected by the address from the DAPR, using the destination function codes from the FCR and the size from the CMR. The same options exist for operand size and alignment as for the source operand read. When the complete operand is written, the DAPR is incremented by one or two, and the BCR is decremented by the number of bytes transferred. See **3.1.2.3 DESTINATION ADDRESS POINTER REGISTER (DAPR)** and **3.1.2.5 BYTE COUNT REGISTER (BCR)** for more details.

3.1.4.3 ADDRESS SEQUENCING. The manner in which the DAPR and SAPR are incremented during a transfer depends on the programming of the SAPI and DAPI bits, the source and destination sizes (DSIZE and SSIZE), and the system data bus width.

The IDMA will run at least two, and up to four, bus cycles to transfer each operand. With an 8-bit bus width, SSIZE and DSIZE are ignored, and each operand transfer requires two cycles. With a 16-bit bus width, the number of bus cycles required to transfer each operand is determined by DSIZE and SSIZE, whether the source and destination addresses are odd or even, and whether the BCR equals one. When SSIZE and DSIZE both select either a byte or word, there will be no operand packing, and the operand transfer will take two bus cycles. One exception occurs when DSIZE and SSIZE are words and the address is odd. In this case, there will be two (one byte each) memory cycles for each read or write at an odd address. When both the source and destination addresses are odd, four bus cycles are required to transfer each operand. When SSIZE and DSIZE are not equal, the IDMA will perform operand packing. If SSIZE is one byte, two read cycles are required to fetch the operand. If DSIZE is one byte, two write cycles are required to store the operand.

When SAPI and/or DAPI are programmed to increment either SAPR or DAPR, the amount (one or two) by which the address pointer increments depends upon DSIZE, SSIZE, and the bus width.

When operating in a 16-bit bus environment with an 8-bit peripheral, the peripheral may be placed on one-half of the bus (consecutive even or odd addresses only). In this case, SSIZE (or DSIZE) must be set to 16 bit, and the IDMA will perform data packing. If the 8-bit peripheral is to be arranged with consecutive addresses, both SSIZE and DSIZE must be 8 bit. As a result, the peripheral's addresses must be incremented twice after each peripheral bus cycle, which results in adding four to the address for each data transfer (two cycles per transfer). This is consistent with the M68000 MOVEP instruction.

Refer to Table 3-1 to see how the SAPR and DAPR will be incremented in all combinations.

Table 3-1. SAPR and DAPR Incrementing Rules

Bus Width	Source Size	Destination Size	SAPR Increment	DAPR Increment	Transfer Description
8 Bit	X	X	+1	+1	Read Byte — Write Byte Packing is Not Possible
16 Bit	Byte	Byte	+1	+1	Read Byte — Write Byte Packing is Not Desired
16 Bit	Byte	Word	+4	+2	Read Byte, Read Byte — Write Word Operand Packing
16 Bit	Word	Byte	+2	+4	Read Word — Write Byte, Write Byte Operand Unpacking
16 Bit	Word	Word	+2	+2	Read Word — Write Word

3.1.4.4 TRANSFER REQUEST GENERATION. IDMA transfers may be initiated by either internally or externally generated requests. Internally generated requests can be initiated by setting STR in the CMR. Externally generated transfers are those requested by an external device using $\overline{\text{DREQ}}$ in conjunction with the activation of STR.

Internal Maximum Rate

The first method of internal request generation is a nonstop transfer until the transfer count is exhausted. If this method is chosen, the IDMA will arbitrate for the bus and begin transferring data after STR is set and the IDMA becomes the bus master. If no exception occurs, all operands in the data block will be transferred in one burst with the IDMA using 100 percent of the available bus bandwidth (unless an external bus master requests the bus or the M68000 core has an unmasked pending interrupt request and BCLM = 1). See **3.1.6 DMA Bus Arbitration** for more details.

Internal Limited Rate

To guarantee that the IDMA will not use all the available system bus bandwidth during a transfer, internal requests can be limited to the amount of bus bandwidth allocated to the IDMA. Programming the REQG bits to "internal limited rate" and the BT bits to limit the percentage of bandwidth achieves this result. As soon as STR is set, the IDMA module arbitrates for the bus and begins to transfer data when it becomes bus master. If no exception occurs, transfers will continue normally, but the IDMA will not

exceed the percentage of bus bandwidth programmed into the control register (12%, 25%, 50%, or 75%). This percentage is calculated over each ensuing 1024 internal clock cycle period.

External Burst Mode

For external devices requiring very high data transfer rates, the external burst mode allows the IDMA to use all the bus bandwidth to service the device. In the burst mode, the $\overline{\text{DREQ}}$ input to the IDMA is level-sensitive and is sampled at certain points to determine when a valid request is asserted by the device. The device requests service by asserting $\overline{\text{DREQ}}$ and leaving it asserted. In response, the IDMA arbitrates for the system bus and begins to perform an operand transfer. During each access to the device, the IDMA will assert $\overline{\text{DACK}}$ to indicate to the device that a request is being serviced. If $\overline{\text{DREQ}}$ is asserted when the IDMA completes the peripheral cycle (the cycle during which $\overline{\text{DACK}}$ is asserted by the IDMA) one setup time before $\overline{\text{DTACK}}$, then a valid request for another operand transfer is recognized, and the IDMA will service the next request immediately. If $\overline{\text{DREQ}}$ is negated before $\overline{\text{DTACK}}$, a new request will not be recognized, and the IDMA will relinquish the bus.

External Cycle Steal

For external devices that generate a pulsed signal for each operand to be transferred, the external cycle steal mode uses $\overline{\text{DREQ}}$ as a falling edge-sensitive input. The IDMA will respond to cycle-steal requests in the same manner as for all other requests. However, if subsequent $\overline{\text{DREQ}}$ pulses are generated before $\overline{\text{DACK}}$ is asserted in response to each request, they will be ignored. If $\overline{\text{DREQ}}$ is asserted after the IDMA asserts $\overline{\text{DACK}}$ for the previous request but before $\overline{\text{DTACK}}$ is asserted, then the new request will be serviced before the bus is relinquished. If a new request has not been generated by the time $\overline{\text{DTACK}}$ has been asserted, the bus will be released to the next bus master.

3.1.4.5 BLOCK TRANSFER TERMINATION. The user may stop the channel by clearing STR. Additionally, the channel operation can be terminated for any of the following reasons: transfer count exhausted, external device termination, or error termination. This is independent of how requests are generated to the IDMA.

Transfer Count Exhausted

When the channel begins an operand transfer, if the current value of the BCR is one or two (according to the operand size in the CMR), $\overline{\text{DONE}}$ is asserted during the bus cycle to the device to indicate that the channel

operation will be terminated when the current operand transfer has successfully completed. When the operand transfer has completed and the BCR has been decremented to zero, the channel operation is terminated, STR is cleared, and an interrupt is generated if INTN is set. The SAPR and/or DAPR are also incremented in the normal fashion.

NOTE

If the channel is started with BCR value set to zero, the channel will transfer 64K bytes.

External Device Termination

If desired, a transfer may be terminated by the device even before the BCR is decremented to zero. If $\overline{\text{DONE}}$ is asserted while $\overline{\text{DTACK}}$ is asserted during a device access, then the channel operation will be terminated following the operand transfer (see the DNS bit in the CSR). STR is cleared, and an interrupt is generated if INTN is set. The BCR is also decremented, and the SAPR and/or DAPR are incremented in the normal fashion. The use of $\overline{\text{DONE}}$ is not limited to external request generation only; it may also be used to externally terminate an internally generated IDMA transfer sequence.

Error Termination

When a fatal error occurs during an IDMA bus cycle, a bus error is used to abort the cycle and terminate the channel operation. STR is cleared, either BED or BES is set, and an error interrupt is generated if INTE is set.

3.1.5 IDMA Programming

Once the channel has been initialized with all parameters required for a transfer operation, it is started by setting the start operation (STR) bit in the CMR. After the channel has been started, any register that describes the current operation may be read but not modified (SAPR/DAPR, FCR, or BCR).

Once STR has been set, the channel is active and either accepts operand transfer requests in external mode or generates requests automatically in internal mode. When the first valid external request is recognized, the IDMA arbitrates for the bus. The $\overline{\text{DREQ}}$ input is ignored until STR is set.

STR is cleared automatically when the BCR reaches zero and the channel transfer is either terminated by $\overline{\text{DONE}}$ or the IDMA cycle is terminated by a bus error.

Channel transfer operation may be suspended at any time by clearing STR. In response, any operand transfer in progress will be completed, and the bus will be released. No further bus cycles will be started while STR remains negated. During this time, the M68000 core may access IDMA internal registers to determine channel status or to alter operation. When STR is set again, if a transfer request is pending, the IDMA will arbitrate for the bus and continue normal operation.

Interrupt handling for the IDMA is configured globally through the interrupt pending register (IPR), the IMR, and the interrupt in-service register (ISR). Within each of these registers, two bits are used to either mask, enable, or report the presence of an interrupt in the IDMA. One bit is used for normal termination; the other bit is used for error termination. When the interrupt enable bits in the CMR (INTN and INTE) are cleared and the IDMA status changes, status bits are set in the CSR but not in the IPR. When either INTN or INTE is set and the corresponding event occurs, the appropriate bit is set in the IPR, and, if this bit is not masked, the interrupt controller will interrupt the M68000 core.

3.1.6 DMA Bus Arbitration

The IDMA controller uses the M68000 bus arbitration protocol to request bus mastership before entering the DMA mode of operation. The six SDMA channels have priority over the IDMA and can transfer data between two IDMA cycles with $\overline{\text{BGACK}}$ remaining continuously low. Once the processor has initialized and started a DMA channel, an operand transfer request is made pending by either an external device or by using an internal request.

When the IDMA channel has an operand transfer request pending and $\overline{\text{BCLR}}$ is not asserted, the IDMA will request bus mastership from the internal bus arbiter using the internal signal IDBR (see Figure 3-9). The arbiter will assert the internal M68000 core bus request ($\overline{\text{CBR}}$) signal and will monitor the core bus grant ($\overline{\text{CBG}}$) and external $\overline{\text{BR}}$ to determine when it may grant the IDMA mastership. The IDMA will monitor the address strobe ($\overline{\text{AS}}$) and bus grant acknowledge ($\overline{\text{BGACK}}$) signals. These signals must be negated to indicate that the previous bus cycle has completed and the previous bus master has released the bus. When these conditions are met, the arbiter only asserts $\overline{\text{BGACK}}$ to indicate that the IDMA has taken control of the bus. When all operand transfers have occurred, the arbiter will release control of the bus by negating $\overline{\text{BGACK}}$.

Internally generated IDMA requests are affected by a mechanism supported to reduce the M68000 core interrupt latency and external bus master arbitration latency (see **3.8.5 Bus Arbitration Logic**). The IDMA is forced to relinquish the bus when an external bus master requests the bus ($\overline{BR}$ is asserted) or when the M68000 core has an unmasked pending interrupt request. In these cases, the on-chip arbiter sends an internal bus-clear signal to the IDMA. In response, any operand transfer in progress will be fully completed (up to four bus cycles depending on the configration), and bus ownership will be released.

If the core caused the bus to be relinquished, no further IDMA bus cycles will be started until IPA in the SCR is cleared. If the cause was an external request, no further IDMA bus cycles will be started while $\overline{BR}$ remains asserted. When $\overline{BR}$ is externally negated, if a transfer request is pending, the IDMA will arbitrate for the bus and continue normal operation.

3.1.7 Bus Exceptions

In any computer system, the possibility always exists that an error will occur during a bus cycle due to a hardware failure, random noise, or an improper access. When an asynchronous bus structure, such as that supported by the M68000 is used, it is easy to make provisions allowing a bus master to detect and respond to errors during a bus cycle. The IDMA recognizes the same bus exceptions as the M68000 core: reset, bus error, halt, and retry.

NOTE

These exceptions also apply to the SDMA channels except that the reporting method is different. See **4.5 SERIAL COMMUNICATION CONTROLLERS (SCCs)** for further details.

3.1.7.1 RESET. Upon an external reset, the IDMA channel immediately aborts the channel operation, returns to the idle state, and clears CSR and CMR (including the STR bit). If a bus cycle is in progress when reset is detected, the cycle is terminated, the control and address/data pins are three-stated, and bus ownership is released. The IDMA can also be reset by RST in the CMR.

3.1.7.2 BUS ERROR. When a fatal error occurs during a bus cycle, a bus error exception is used to abort the cycle and systematically terminate that channel's operation. The IDMA terminates the current bus cycle, signals an error in the CSR, and generates a maskable interrupt. The IDMA clears STR and

waits for a restart of the channel and the negation of $\overline{\text{BERR}}$ before starting any new bus cycles.

NOTE

Any data that was previously read from the source into the DHR will be lost.

3.1.7.3 HALT. IDMA transfer operation may be suspended at any time by asserting $\overline{\text{HALT}}$ to the IDMA. In response, any bus cycle in progress is completed (after $\overline{\text{DTACK}}$ is asserted), and bus ownership is released. No further bus cycles will be started while $\overline{\text{HALT}}$ remains asserted. When the IDMA is in the middle of an operand transfer when halted and $\overline{\text{HALT}}$ is subsequently negated, and if a new transfer request is pending, then IDMA will arbitrate for the bus and continue normal operation.

3.1.7.4 RETRY. When $\overline{\text{HALT}}$ and $\overline{\text{BERR}}$ are asserted during a bus cycle, the IDMA terminates the bus cycle, releases the bus, and suspends any further operation until these signals are negated. When $\overline{\text{HALT}}$ and $\overline{\text{BERR}}$ are negated, the IDMA will arbitrate for the bus, re-execute the previous bus cycle, and continue normal operation.

3.2 INTERRUPT CONTROLLER

The IMP interrupt controller accepts and prioritizes both internal and external interrupt requests and generates a vector number during the CPU interrupt acknowledge cycle. Interrupt nesting is also provided so that an interrupt service routine of a lower priority interrupt may be suspended by a higher priority interrupt request. The interrupt controller block diagram is shown in Figure 3-2.

The on-chip interrupt controller has the following features:

- Two Operational Modes: Normal and Dedicated
- Eighteen Prioritized Interrupt Sources (Internal and External)
- A Fully Nested Interrupt Environment
- Unique Vector Number for Each Internal/External Source Generated
- Three Interrupt Request and Interrupt Acknowledge Pairs
- $\overline{\text{DTACK}}$ Generation When Vectors Supplied Internally

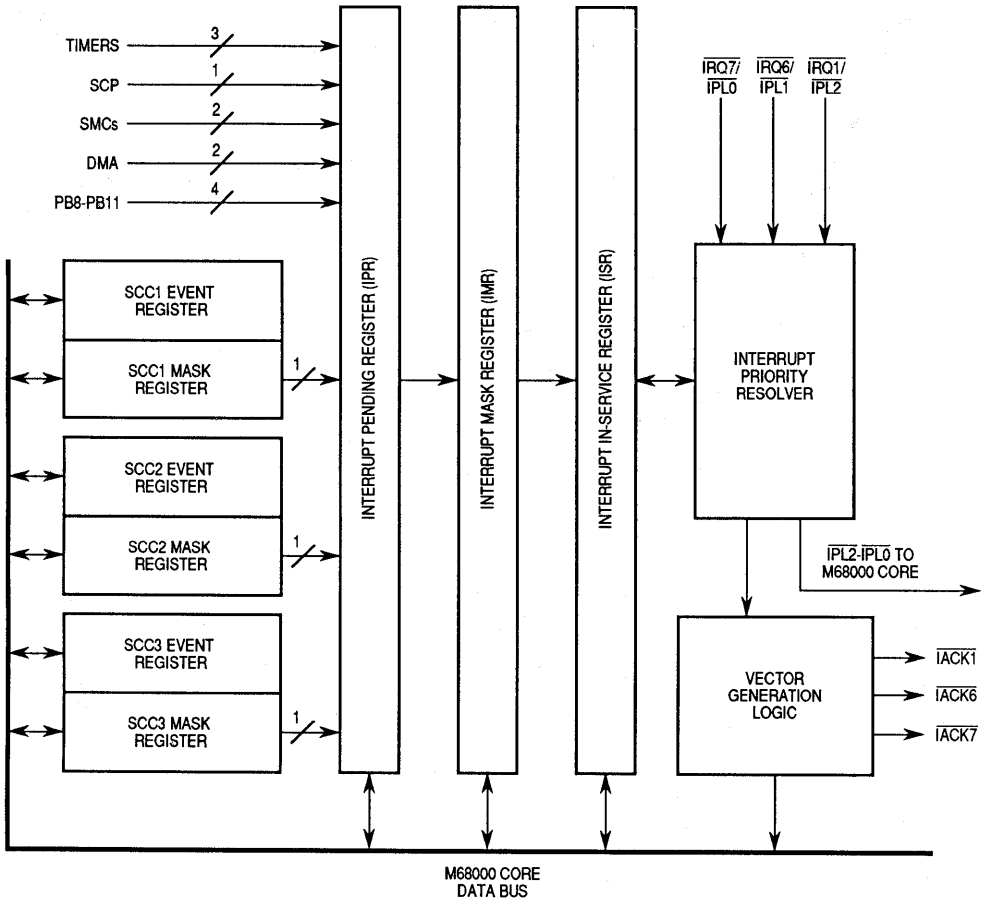


Figure 3-2. Interrupt Controller Block Diagram

3.2.1 Operation

The interrupt controller receives interrupts from internal sources such as the timers, the IDMA controller, the serial communication controllers, and the parallel I/O pins (port B pins 11–8). These interrupts are called internal requests (INRQ). The interrupt controller allows for masking each INRQ interrupt source. When multiple events within a peripheral can cause the INRQ interrupt, each event is also maskable.

In addition to the INRQ interrupts, the interrupt controller can also receive external requests (EXRQ). EXRQ interrupts are input to the IMP according to the operational mode selected by the user. In the normal mode, EXRQ in-

terrupts are encoded on the $\overline{IPL2}$–$\overline{IPL0}$ lines. In the dedicated mode, EXRQ interrupts are presented directly as $\overline{IRQ7}$, $\overline{IRQ6}$, and $\overline{IRQ1}$.

Normal Mode

In this mode, the three external interrupt request pins are configured as $\overline{IPL2}$–$\overline{IPL0}$. Up to seven levels of interrupt priority may be implemented in the IMP system. Level 4 is reserved for IMP INRQ interrupts and may not be generated by an external device.

Dedicated Mode

In this mode, the three interrupt request pins are configured as $\overline{IRQ7}$, $\overline{IRQ6}$, and $\overline{IRQ1}$ to provide dedicated request lines for three external sources. Each of these lines is programmed to be edge-triggered or level-sensitive. In addition to level 4, which is reserved for INRQ interrupts, interrupt priority levels 2, 3, and 5 must not be assigned to external devices in this mode.

In systems that use the dedicated mode, the user may program the port B control register (PBCNT) to select the dedicated on-chip peripheral functions for $\overline{IACK7}$, $\overline{IACK6}$, $\overline{IACK1}$ (normally PB2–PB0). Dedicated interrupt acknowledge signals eliminate the need for external logic to perform the decoding of the A19–A16, A3–A1, and FC2–FC0 pins and allow an external device to detect an interrupt acknowledge cycle. By selecting the dedicated mode for the interrupt controller and correctly programming PBCNT, the user can create three interrupt request/interrupt acknowledge pairs.

The interrupt controller also prioritizes both INRQ and EXRQ interrupts for handling by the M68000 core. All INRQ interrupt sources are assigned a fixed priority level (level 4). The system designer assigns each EXRQ interrupt to an appropriate priority level so that pending interrupts are serviced according to their relative importance. If more than one interrupt request is pending, the interrupt controller presents the highest priority interrupt. A priority encoder combines EXRQ requests and INRQ requests to deliver $\overline{IPL2}$–$\overline{IPL0}$ to the M68000 core.

In response to an interrupt request, the M68000 core executes an interrupt acknowledge cycle. Interrupt latency can be improved by programming the internal interrupt pending ($\overline{IPEND}$) signal to assert the bus clear ($\overline{BCLR}$) signal and by using this signal to cause the current bus master to relinquish the M68000 bus. This involves using the IPA and BCLM bits in the system control register (SCR), and is discussed more fully in **3.8 System Control**. During an interrupt acknowledge cycle, FC2–FC0 are encoded as 111, A3–A1 encode the priority level, and A19–A16 are driven high. The timings are those of the read-cycle timing diagram.

The interrupt controller recognizes an interrupt acknowledge cycle by decoding these signals. If the interrupt acknowledge cycle is in response to an EXRQ interrupt, the external device may recognize the cycle either by decoding FC2–FC0, A3–A1, and A19–A16 or by detecting the assertion of $\overline{IACK7}$, $\overline{IACK6}$, or $\overline{IACK1}$.

As part of the interrupt acknowledge cycle, an interrupt vector number is passed to the M68000 core on the lower half of the data bus. For INRQ interrupts, the vector number is supplied by the interrupt controller. The user can also program the interrupt controller to provide the vector number for EXRQ interrupts through the global interrupt mode register (GIMR). Bits 7-5 of the vector number are programmed by the user. Bits 4-0 are encoded by the interrupt controller to specify the interrupt source. $\overline{DTACK}$ is provided by the interrupt controller if it provides the vector.

If the vector is supplied by the external device, then the device must either place its interrupt vector on the lower byte of the data bus or assert the $\overline{AVEC}$ pin to use an autovector. (The autovector method maps each interrupt level to a fixed location in the exception vector table regardless of how many interrupt sources exist at that level.)

3.2.2 Interrupt Priorities

INRQ and EXRQ interrupts are assigned to an interrupt priority level. INRQ interrupts are also assigned relative priorities within their given interrupt priority level. A fully nested interrupt environment is provided so that a higher priority interrupt is serviced before a lower priority interrupt.

3.2.2.1 INRQ AND EXRQ PRIORITY LEVELS. Seven levels of interrupt priority may be implemented in IMP system designs, with level 7 having the highest priority. INRQ interrupts are assigned to level 4 (fixed). EXRQ interrupts are assigned by the user to any of the remaining six priority levels in normal mode. In dedicated mode, EXRQ interrupts may be assigned to priority levels 7, 6, and 1.

Table 3-2 indicates the interrupt levels available in both normal and dedicated modes. This table also shows the $\overline{IPL2}$–$\overline{IPL0}$ encoding that should be provided by external logic for each EXRQ interrupt level in normal mode. For the dedicated mode, this table shows the IMP input pins ($\overline{IRQ7}$, $\overline{IRQ6}$, and $\overline{IRQ1}$) that should be asserted by an external device according to the desired interrupt priority level.

Table 3-2. EXRQ and INRQ Prioritization

Priority Level	Normal Mode $\overline{IPL2}$–$\overline{IPL0}$	Dedicated Mode $\overline{IRQ7}$, $\overline{IRQ6}$, $\overline{IRQ1}$	Interrupt Source
7 (Highest)	000	$\overline{IRQ7}$	EXRQ
6	001	$\overline{IRQ6}$	EXRQ
5	010	†	EXRQ
4	†	†	INRQ
3	100	†	EXRQ
2	101	†	EXRQ
1 (Lowest)	110	$\overline{IRQ1}$	EXRQ

†Priority level not available to an external device in this mode.

3.2.2.2 INRQ INTERRUPT SOURCE PRIORITIES. Although all INRQ interrupts are presented at level 4, the interrupt controller further organizes interrupt servicing of the 15 INRQ interrupts according to the priorities illustrated in Table 3-3. The interrupt from the port B pin 11 (PB11) has the highest priority, and the interrupt from the port B pin 8 (PB8) has the lowest priority. A single interrupt priority within level 4 is associated with each table entry. The IDMA entry is associated with the general-purpose DMA channel only, and not with the SDMA channels that service the SCCs. Those interrupts are reported through each individual SCC channel or, in the case of a bus error, through the SDMA channels bus error entry.

Table 3-3. INRQ Prioritization within Interrupt Level 4

Priority Level	Interrupt Source Description	Multiple Interrupt Events
Highest	General-Purpose Interrupt 3 (PB11)	No
	General-Purpose Interrupt 2 (PB10)	No
	SCC1	Yes
	SDMA Channels Bus Error	No
	IDMA Channel	Yes
	SCC2	Yes
	Timer 1	Yes
	SCC3	Yes
	General-Purpose Interrupt 1 (PB9)	No
	Timer 2	Yes
	SCP	No
	Timer 3	No
	SMC1	No
	SMC2	No
	General-Purpose Interrupt 0 (PB8)	No
Lowest	Error	—

3.2.2.3 NESTED INTERRUPTS. The following rules apply to nested interrupts:

1. The interrupt controller responds to all EXRQ and INRQ interrupts based upon their assigned priority level. The highest priority interrupt request is presented to the M68000 core for servicing. After the vector number corresponding to this interrupt is passed to the core during an interrupt acknowledge cycle, an INRQ interrupt request is cleared in IPR. (EXRQ requests must be cleared externally.) The remaining interrupt requests, if any, are then assessed by priority so that another interrupt request may be presented to the core.
2. The 3-bit mask in the M68000 core status register ensures that a subsequent interrupt request at a higher interrupt priority level will suspend handling of a lower priority interrupt. The 3-bit mask indicates the current M68000 priority. Interrupts are inhibited for all priority levels less than or equal to the current M68000 priority. Priority level 7 cannot be inhibited by the mask; it is a nonmaskable interrupt level.
3. The interrupt controller allows a higher priority INRQ interrupt to be presented to the M68000 core before the servicing of a lower priority INRQ interrupt is completed. This is achieved using the interrupt in-service register (ISR). Each bit in the ISR corresponds to an INRQ interrupt source.

   During an interrupt acknowledge cycle for an INRQ interrupt, the in-service bit is set by the interrupt controller for that interrupt source. When this bit is set, any subsequent INRQ interrupt requests at this priority level or lower are disabled until servicing of the current interrupt is completed and the in-service bit is cleared by the user. Pending interrupts for these sources are still set by the corresponding interrupt pending bit.

   Thus, in the interrupt service routine for the INRQ interrupt, the user can lower the M68000 core mask to level 3 in the status register to allow higher priority level 4 (INRQ) interrupts to generate an interrupt request. This capability provides nesting of INRQ interrupt requests for sources within level 4. This capability is similar to the way the M68000 core interrupt mask provides nesting of interrupt requests for the seven interrupt priority levels.

3.2.3 Masking Interrupt Sources and Events

The user may mask EXRQ and INRQ interrupts to prevent an interrupt request to the M68000 core. EXRQ interrupt masking is handled external to the IMP — e.g., by programming a mask register within an external device. INRQ

interrupt masking is accomplished by programming the IMR. Each bit in the IMR corresponds to one of 15 INRQ interrupt sources.

When a masked INRQ interrupt source has a pending interrupt request, the corresponding bit is set in the IPR, even though the interrupt is not generated to the core. By masking all interrupt sources using the IMR, the user may implement a polling interrupt servicing scheme for INRQ interrupts.

When an INRQ interrupt source from an on-chip peripheral has multiple interrupt events, the user can individually mask these events by programming that peripheral mask register. Table 3-3 indicates the interrupt sources that have multiple interrupt events. In this case, when a masked event occurs, an interrupt request is not generated for the associated interrupt source, and the corresponding bit in the IPR is not set. If the corresponding bit in the IPR is already set, then masking the event in the peripheral mask register causes the IPR bit to be cleared. To determine the cause of a pending interrupt when an interrupt source has multiple interrupt events, the user interrupt service routine must read the event register within that on-chip peripheral.

3.2.4 Interrupt Vector Generation

Pending EXRQ interrupts and unmasked INRQ interrupts are presented to the M68000 core in order of priority. The M68000 core responds to an interrupt request by initiating an interrupt acknowledge cycle to receive a vector number, which allows the core to locate the interrupt's service routine.

For INRQ interrupts, the interrupt controller passes an interrupt vector corresponding to the highest priority, unmasked, pending interrupt. By programming the GIMR, the user can also enable the interrupt controller to provide the vector for EXRQ interrupts at levels 1, 6, and 7 (regardless of whether normal or dedicated mode is selected). Whenever a vector is provided by the interrupt controller, $\overline{\text{DTACK}}$ is also provided during the cycle. If the vector is not provided by the IMP, the external device should provide the 8-bit vector or assert the $\overline{\text{AVEC}}$ pin for an autovector.

The three most significant bits of the interrupt vector number are programmed by the user in the GIMR. These three bits are concatenated with five bits generated by the interrupt controller to provide an 8-bit vector number to the core. The interrupt controller's encoding of the five low-order bits of the interrupt vector is shown in Table 3-4. When the core initiates an interrupt acknowledge cycle for level 4 and there is no internal interrupt pending, the interrupt controller encodes the error code 00000 onto the five low-order bits of the interrupt vector.

3.2.5 Interrupt Controller Programming Model

The user communicates with the interrupt controller using four registers. The global interrupt mode register (GIMR) defines the interrupt controller's operational mode. The interrupt pending register (IPR) indicates which INRQ interrupt sources require interrupt service. The interrupt mask register (IMR) allows the user to prevent any of the INRQ interrupt sources from generating an interrupt request. The interrupt in-service register (ISR) provides a capability for nesting INRQ interrupt requests.

Table 3-4. Encoding the Interrupt Vector

Priority Level	5-Bit Vector	Interrupt Source
7 (Highest)	10111	External Device
6	10110	External Device
5	None	External Device
4	01111	General-Purpose Interrupt 3 (PB11)
4	01110	General-Purpose Interrupt 2 (PB10)
4	01101	SCC1
4	01100	SDMA Channels Bus Error
4	01011	IDMA Channel
4	01010	SCC2
4	01001	Timer 1
4	01000	SCC3
4	00111	General-Purpose Interrupt 1 (PB9)
4	00110	Timer 2
4	00101	SCP
4	00100	Timer 3
4	00011	SMC1
4	00010	SMC2
4	00001	General-Purpose Interrupt 0 (PB8)
4	00000	Error
3	None	External Device
2	None	External Device
1 (Lowest)	10001	External Device

3.2.5.1 GLOBAL INTERRUPT MODE REGISTER (GIMR). The user normally writes the GIMR soon after a total system reset. The GIMR is initially $0000 and is reset only upon a total system reset. If bits V7–V5 of the GIMR are not written to specify an interrupt vector prior to the first interrupt condition, the interrupt controller will pass the vector $0F (the uninitialized interrupt vector), regardless of the interrupt source.

15	14	13	12	11	10	9	8	7 5	4 0
MOD	IV7	IV6	IV1	—	ET7	ET6	ET1	V7–V5	RESERVED

MOD — Mode
- 0 = Normal operational mode. Interrupt request lines are configured as $\overline{IPL2}$–$\overline{IPL0}$.
- 1 = Dedicated operational mode. Interrupt request lines are configured as $\overline{IRQ7}$, $\overline{IRQ6}$, and $\overline{IRQ1}$.

IV7 — Level 7 Interrupt Vector

This bit is valid in both normal and dedicated modes.
- 0 = Internal vector. The interrupt controller will provide the vector number for a level 7 interrupt during interrupt acknowledge cycle.
- 1 = External vector. The interrupt controller will not provide the vector number for a level 7 interrupt.

IV6 — Level 6 Interrupt Vector

This bit is valid in both normal and dedicated modes.
- 0 = Internal vector. The interrupt controller will provide the vector number for a level 6 interrupt during the interrupt acknowledge cycle.
- 1 = External vector. The interrupt controller will not provide the vector number for a level 6 interrupt.

IV1 — Level 1 Interrupt Vector

This bit is valid in both normal and dedicated modes.
- 0 = Internal vector. The interrupt controller will provide the vector number for a level 1 interrupt acknowledge cycle.
- 1 = External vector. The interrupt controller will not provide the vector number for a level 1 interrupt.

ET7 — $\overline{IRQ7}$ Edge-/Level-Triggered

This bit is valid only in the dedicated mode.
- 0 = Level-triggered. An interrupt is generated when $\overline{IRQ7}$ is low.

NOTE

During the M68000 core interrupt acknowledge cycle for $\overline{IRQ7}$, if $\overline{IRQ7}$ is not continuously asserted, the interrupt controller will still provide the vector number (and $\overline{DTACK}$) regardless of the IV7 bit.

1 = Edge-triggered. An interrupt is generated when $\overline{IRQ7}$ changes from one to zero (falling edge).

ET6 — $\overline{IRQ6}$ Edge-/Level-Triggered
This bit is valid only in the dedicated mode.
0 = Level-triggered. An interrupt is generated when $\overline{IRQ6}$ is low.

NOTE

During the M68000 core interrupt acknowledge cycle for $\overline{IRQ6}$, if $\overline{IRQ6}$ is not continuously asserted, the interrupt controller will still provide the vector number (and $\overline{DTACK}$) regardless of the IV6 bit.

1 = Edge-triggered. An interrupt is generated when $\overline{IRQ6}$ changes from one to zero (falling edge).

ET1 — $\overline{IRQ1}$ Edge-/Level-Triggered
This bit is valid only in the dedicated mode.
0 = Level-triggered. An interrupt is generated when $\overline{IRQ1}$ is low.

NOTE

During the M68000 core interrupt acknowledge cycle for $\overline{IRQ1}$, if $\overline{IRQ1}$ is not continuously asserted, the interrupt controller will still provide the vector number (and $\overline{DTACK}$) regardless of the IV1 bit.

1 = Edge-triggered. An interrupt is generated when $\overline{IRQ1}$ changes from one to zero (falling edge).

V7–V5 — Interrupt Vector Bits 7–5
These three bits are concatenated with five bits provided by the interrupt controller, which indicate the specific interrupt source, to form an 8-bit interrupt vector number. If these bits are not written, the vector $0F is provided.

Bits 11 and 4–0 — Reserved for future use.

3.2.5.2 INTERRUPT PENDING REGISTER (IPR). Each bit in the 16-bit IPR corresponds to an INRQ interrupt source. When an INRQ interrupt is received, the interrupt controller sets the corresponding bit in the IPR. In a vectored interrupt environment, the interrupt controller clears the IPR bit when the vector number corresponding to the INRQ interrupt source is passed to the M68000 core during an interrupt acknowledge cycle. In a polled interrupt scheme, the user must periodically read the IPR. When a pending interrupt is handled, the user should clear the corresponding bit in the IPR by writing a one to that bit. Since the user can only clear bits in this register, the bits that are written as zeros will not be affected. The IPR is cleared by reset.

NOTE

The ERR bit is set if the user drives the $\overline{IPL2}$–$\overline{IPL0}$ lines to interrupt level 4 and no INRQ interrupt is pending.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB11	PB10	SCC1	SDMA	IDMA	SCC2	TIMER1	SCC3	PB9	TIMER2	SCP	TIMER3	SMC1	SMC2	PB8	ERR

3.2.5.3 INTERRUPT MASK REGISTER (IMR). Each bit in the 16-bit IMR corresponds to an INRQ interrupt source. The user masks an interrupt source by clearing the corresponding bit in the IMR. When a masked INRQ interrupt occurs, the corresponding bit in the IPR is set, but the IMR bit prevents the interrupt request from reaching the M68000 core. If an INRQ source is requesting interrupt service when the user clears the IMR bit, the request to the core will cease, but the IPR bit remains set. If the IMR bit is then set later by the user, the pending interrupt request will once again request interrupt service and will be processed by the core according to its assigned priority. The IMR, which can be read by the user at any time, is cleared by reset.

If a bit in the IMR is masked during the time that the interrupt at level 4 is presented to the M68000 core to begin the interrupt acknowledge cycle, then a special case occurs. If other interrupts are pending at level 4, then the interrupt controller will acknowledge the interrupt with a vector from one of those interrupt sources. If no other interrupts are pending at level 4, then the interrupt controller will acknowledge the interrupt with the error vector (00000b). If desired, this case can be prevented by raising the interrupt mask in the M68000 core status register to 4 before masking the interrupt source, and subsequently lowering it. Also, if the interrupt source has multiple events (e.g., SCC1), then the interrupts for that peripheral can be masked within the peripheral mask register.

NOTE

To clear bits that were set by multiple interrupt events, the user should clear all the unmasked events in the corresponding on-chip peripheral's event register.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB11	PB10	SCC1	SDMA	IDMA	SCC2	TIMER1	SCC3	PB9	TIMER2	SCP	TIMER3	SMC1	SMC2	PB8	ERR

3.2.5.4 INTERRUPT IN-SERVICE REGISTER (ISR). Each bit in the 16-bit ISR corresponds to an INRQ interrupt source. In a vectored interrupt environment, the interrupt controller sets the ISR bit when the vector number corresponding to the INRQ interrupt source is passed to the core during an interrupt acknowledge cycle. The user's interrupt service routine should clear this bit during the servicing of the interrupt. (If an event register exists for this peripheral, its bits would normally be cleared as well). To clear a bit in the ISR, the user writes a one to that bit. Since the user can only clear bits in this register, the bits that are written as zeros will not be affected. The ISR is cleared by reset.

This register may be read by the user to determine which INRQ interrupts are currently being processed. More than one bit in the ISR may be a one if the capability is used to allow higher priority level 4 interrupts to interrupt lower priority level 4 interrupts. See **3.2.2.3 NESTED INTERRUPTS** for more details.

The user can control the extent to which level 4 interrupts may interrupt other level 4 interrupts by selectively clearing the ISR. A new INRQ interrupt will be processed if it has a higher priority than the highest priority INRQ interrupt having its ISR bit set. Thus, if an INRQ interrupt routine lowers the 3-bit mask in the M68000 core to level 3 and also clears its ISR bit at the beginning of the interrupt routine, then a lower priority INRQ interrupt can interrupt it as long as the lower priority is higher than any other ISR bits that are set.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB11	PB10	SCC1	SDMA	IDMA	SCC2	TIMER1	SCC3	PB9	TIMER2	SCP	TIMER3	SMC1	SMC2	PB8	—

3.3 PARALLEL I/O PORTS

The IMP supports two general-purpose I/O ports, port A and port B, whose pins can be general-purpose I/O pins or dedicated peripheral interface pins. Some port B pins are always maintained as four general-purpose I/O pins, each with interrupt capability.

3.3.1 Port A

Each of the 16 port A pins are independently configured as a general-purpose I/O pin if the corresponding port A control register (PACNT) bit is cleared. Port A pins are configured as dedicated on-chip peripheral pins if the corresponding PACNT bit is set.

When acting as a general-purpose I/O pin, the signal direction for that pin is determined by the corresponding control bit in the port A data direction register (PADDR). The port I/O pin is configured as an input if the corresponding PADDR bit is cleared; it is configured as an output if the corresponding PADDR bit is set. All PACNT bits and PADDR bits are cleared on total system reset, configuring all port A pins as general-purpose input pins. (Note that the port pins do not have internal pullup resistors).

If a port A pin is selected as a general-purpose I/O pin, it may be accessed through the port A data register (PADAT). Data written to the PADAT is stored in an output latch. If a port A pin is configured as an output, the output latch data is gated onto the port pin. In this case, when the PADAT is read, the contents of the output latch associated with the output port pin are read. If a port A pin is configured as an input, data written to PADAT is still stored in the output latch but is prevented from reaching the port pin. In this case, when PADAT is read, the state of the port pin is read.

If a port A pin is selected as a dedicated on-chip peripheral pin, the corresponding bit in the PADDR is ignored, and the direction of the pin is determined by the operating mode of the on-chip peripheral. In this case, the PADAT contains the current state of the input pin or output driver.

Certain pins may be selected as general-purpose I/O pins, even when other pins related to the same on-chip peripheral are used as dedicated pins. For example, a system that configures SCC2 to operate in a nonmultiplexed mode without the modem control lines and external clocks (RCLK2, TCLK2, $\overline{CD2}$, $\overline{CTS2}$, and $\overline{RTS2}$) may dedicate the data lines (RXD2 and TXD2) to SCC2 and configure the others as general-purpose I/O pins. What the peripheral now receives as its input, given that some of its pins have been reassigned, is shown in Table 3-5. If an input pin to a channel (for example $\overline{CD2}$ or $\overline{CTS2}$) is used as a general-purpose I/O pin, then the input to the peripheral is automatically connected internally to V_{DD} or GND, based on the pin's function. This does not affect the operation of the port pins in their general-purpose I/O function.

NOTE

If the $\overline{\text{DREQ}}$/PA13 pin is selected to be PA13, then $\overline{\text{DREQ}}$ is tied low. If the IDMA is programmed for external requests, then it always recognizes an external request, and the entire block will be transferred in one burst.

Table 3-5. Port A Pin Functions

PACNT Bit=1 Pin Function	PACNT Bit=0 Pin Function	Input to SCC2/SCC3/IDMA
RXD2	PA0	GND
TXD2	PA1	—
RCLK2	PA2	GND
TCLK2	PA3	RCLK2
$\overline{\text{CTS2}}$	PA4	GND
$\overline{\text{RTS2}}$	PA5	—
$\overline{\text{CD2}}$	PA6	GND
SDS2/BRG2	PA7	—
RXD3	PA8	GND
TXD3	PA9	—
RCLK3	PA10	GND
TCLK3	PA11	RCLK3
BRG3	PA12	—
$\overline{\text{DREQ}}$	PA13	GND
$\overline{\text{DACK}}$	PA14	—
$\overline{\text{DONE}}$	PA15	V_{DD}

3.3.2 Port B

Port B has 12 pins. PB8–PB0 may be configured as general-purpose I/O pins or as dedicated peripheral interface pins; whereas, PB11–PB9 are always maintained as four general-purpose pins, each with interrupt capability.

3.3.2.1 PB7–PB0. Each port B pin may be configured as a general-purpose I/O pin or as a dedicated peripheral interface pin. PB7–PB0 functions exactly like PA15–PA0, except that PB7–PB0 is controlled by the port B control register (PBCNT), the port B data direction register (PBDDR), and the port B data register (PBDAT), and PB7 is configured as an open-drain output ($\overline{\text{WDOG}}$) upon total system reset.

Table 3-6 shows the dedicated function of each pin. The third column shows the input to the peripheral when the pin is used as a general-purpose I/O pin.

Table 3-6. Port B Pin Functions

PBCNT Bit = 1 Pin Function	PBCNT Bit = 0 Pin Function	Input to Interrupt Control and Timers
$\overline{\text{IACK7}}$	PB0	—
$\overline{\text{IACK6}}$	PB1	—
$\overline{\text{IACK1}}$	PB2	—
TIN1	PB3	GND
$\overline{\text{TOUT1}}$	PB4	—
TIN2	PB5	GND
$\overline{\text{TOUT2}}$	PB6	—
$\overline{\text{WDOG}}$	PB7	—

3.3.2.2 PB11–PB8. PB11–PB8 are four general-purpose I/O pins continuously available as general-purpose I/O pins and, therefore, are not referenced in the PBCNT. PB8 operates like PB11–PB9 unless it is used as the DRAM refresh controller request pin, as selected in the system control register (SCR).

The direction of each pin is determined by the corresponding bit in the PBDDR. The port pin is configured as an input if the corresponding PBDDR bit is cleared; it is configured as an output if the corresponding PBDDR bit is set. PBDDR11–PBDDR8 are cleared on total system reset, configuring all PB11–PB8 pins as general-purpose input pins. (Note that the port pins do not have internal pullup resistors). The GIMR is also cleared on total system reset so that if any PB11–PB8 pin is left floating it will not cause a spurious interrupt.

The PB11–PB8 pins are accessed through the PBDAT. Data written to PBDAT11–PBDAT8 is stored in an output latch. If the port pin is configured as an output, the output latch data is gated onto the port pin. In this case, when PBDAT11–PBDAT8 is read, the contents of the output latch associated with the output port pin are read.

When a PB11–PB8 pin is configured as an input, a high-to-low change will cause an interrupt request signal to be sent to the IMP interrupt controller. Each of the four interrupt requests is associated with a fixed internal interrupt

priority level within level 4. (The priority at which each bit requests an interrupt is detailed in Table 3-3.) Each request can be masked independently in the IMP interrupt controller by clearing the appropriate bit in the IMR (PB11–PB8). In this case, when PBDAT11–PBDAT8 is read, the state of the port pin is read.

3.3.3 I/O Port Registers

The I/O port consists of three memory-mapped read-write 16-bit registers for port A and three memory-mapped read-write 16-bit registers for port B. Refer to Figure 3-3 for the I/O port registers. The reserved bits are read as zeros.

PORT A CONTROL REGISTER (PACNT)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CA	CA	CA	CA	CA	CA	CA	CA	CA	CA	CA	CA	CA	CA	CA	CA

PORT A DATA DIRECTION REGISTER (PADDR)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA

PORT A DATA REGISTER (PADAT)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA	PA	PA	PA	PA	PA	PA	PA	PA	PA	PA	PA	PA	PA	PA	PA

PORT B CONTROL REGISTER (PBCNT)

15	8	7	6	5	4	3	2	1	0
RESERVED		CB	CB	CB	CB	CB	CB	CB	CB

PORT B DATA DIRECTION REGISTER (PBDDR)

15	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		DB	DB	DB	DB	DB	DB	DB	DB	DB	DB	DB	DB

PORT B DATA REGISTER (PBDAT)

15	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		PB	PB	PB	PB	PB	PB	PB	PB	PB	PB	PB	PB

Figure 3-3. Parallel I/O Port Registers

3.4 DUAL-PORT RAM

The CP has 1152 bytes of static RAM configured as a dual-port memory. The dual-port RAM can be accessed by the CP main controller or by one of three bus masters: the M68000 core, the IDMA, or an external master. The M68000 core and the IDMA access the RAM synchronously with no wait states. The external master requests the M68000 bus using the $\overline{BR}$ pin and is granted bus ownership. The external master must then access the RAM synchronously with respect to the IMP system clock with zero or one wait state, or asynchronously as determined by the EMWS and SAM bits in the system control register. Except for several locations initialized by the CP, the dual-port RAM is undefined at power-on reset but is not modified by successive resets. The RAM is divided into two parts: parameter RAM and system RAM.

The 576-byte parameter RAM area includes pointers, counters, and registers used with the serial ports. This area is accessed by the CP during communications processing. Any individual locations not required in a given application may be used as general-purpose RAM.

The 576-byte system RAM is a general-purpose RAM, which may be used as M68000 data and/or program RAM or CP microcode RAM. As data RAM, it can include serial port data buffers or can be used for other purposes such as a no-wait-state cache for the M68000 core. As CP microcode RAM, it is used exclusively to store microcode for the CP main controller, allowing the development of special protocols or protocol enhancements, under special arrangement with Motorola. Appendix C discusses available offerings.

The RAM block diagram is shown in Figure 3-4. The M68000 core, the IDMA, and the external master access the RAM through the IMP bus interface unit (BIU) using the M68000 bus. When an access is made, the BIU generates a wait signal to the CP main controller to prevent simultaneous access of the RAM. The CP main controller waits for one cycle to allow the RAM to service the M68000 bus cycle and then regenerates its RAM cycle. This mechanism allows the RAM to be accessed synchronously by the M68000 core, IDMA, or external master without wait states. Thus, during the four-clock M68000 memory cycle, three internal accesses by the CP main controller may occur. The BIU also provides the $\overline{DTACK}$ signal output when the RAM and on-chip registers are accessed by any M68000 bus master.

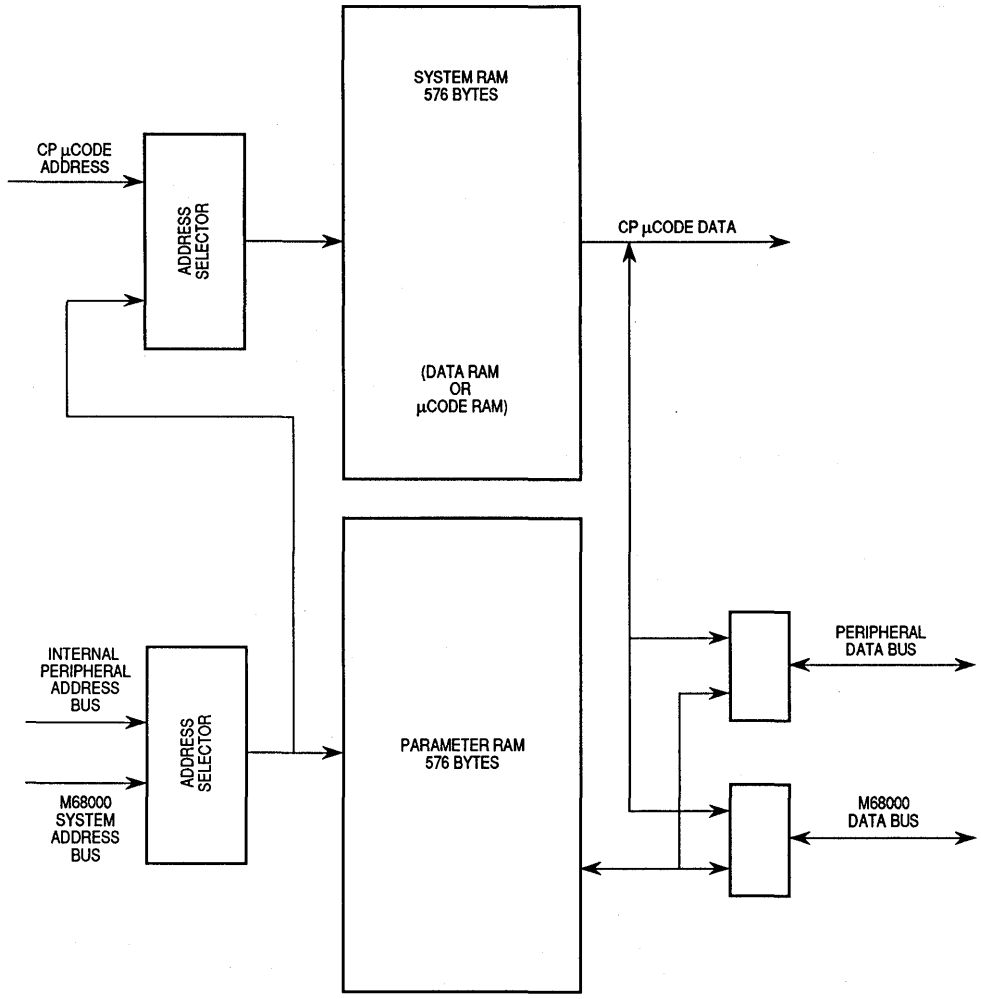


Figure 3-4. RAM Block Diagram

3.5 TIMERS

The MC68302 includes three timer units: two identical general-purpose timers and a watchdog timer.

Each general-purpose timer consists of a timer mode register (TMR), a timer capture register (TCR), a timer counter (TCN), a timer reference register (TRR), and a timer event register (TER). The TMR contains the prescaler value programmed by the user. The watchdog timer, which has a watchdog reference

register (WRR) and a watchdog counter (WCN), uses a fixed prescaler value. The timer block diagram is shown in Figure 3-5.

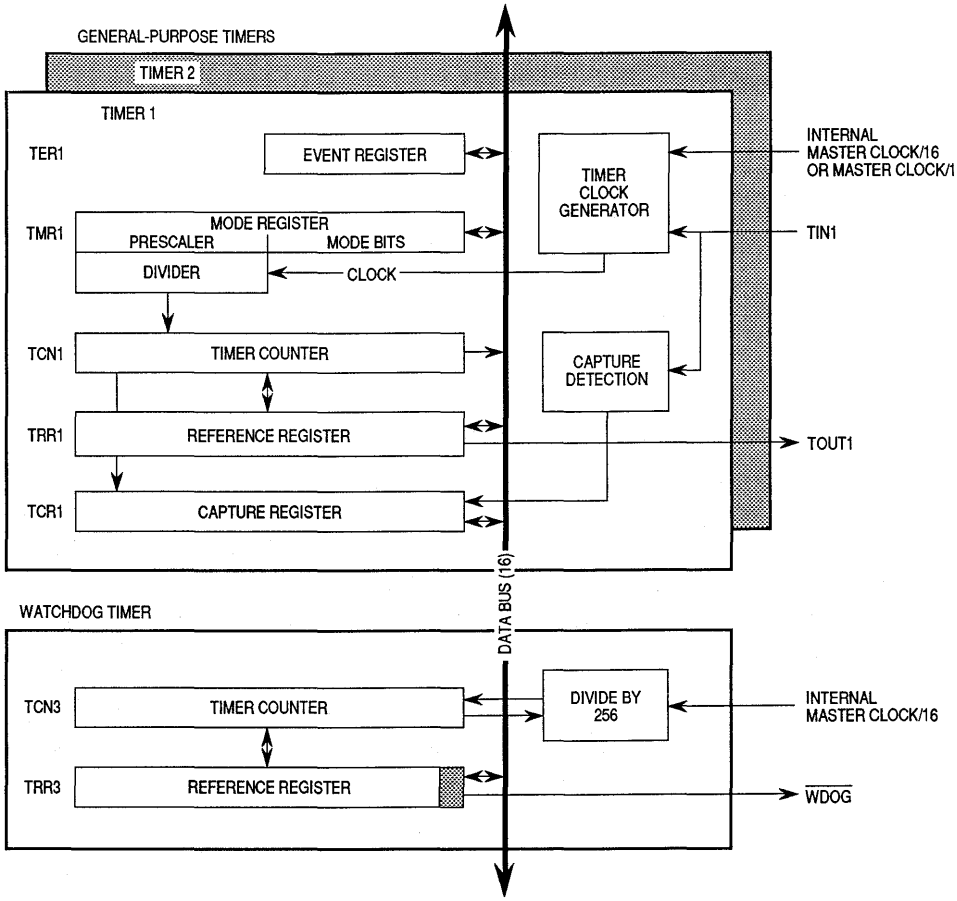


Figure 3-5. Timer Block Diagram

3.5.1 Timer Key Features

The two identical general-purpose timer units have the following features:

- Maximum Period of 16 Seconds (at 16.67 MHz)
- 60-ns Resolution (at 16.67 MHz)
- Programmable Sources for the Clock Input

- Input Capture Capability
- Output Compare with Programmable Mode for the Output Pin
- Two Timers Cascadable to Form a 32-Bit Timer
- Free Run and Restart Modes

The watchdog timer has the following features:

- A 16-bit Counter and Reference Register
- Maximum Period of 16.67 Seconds (at 16 MHz)
- 0.5-ms Resolution (at 16 MHz)
- Output Signal ($\overline{\text{WDOG}}$)
- Interrupt Capability

3.5.2 General-Purpose Timer Units

The clock input to the prescaler may be selected from the main clock (divided by 1 or by 16) or from the corresponding timer input (TIN) pin. TIN is internally synchronized to the internal clock. The clock input source is selected by the ICLK bits of the corresponding TMR. The prescaler is programmed to divide the clock input by values from 1 to 256. The output of the prescaler is used as an input to the 16-bit counter.

The resolution of the timer is one clock cycle (60 ns at 16.67 MHz). The maximum period (when the reference value is all ones) is 268,435,456 cycles (16.67 seconds at 16.00 MHz).

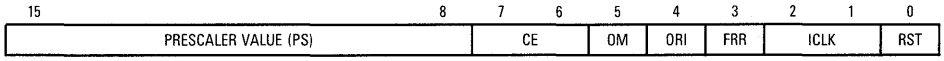
Each timer may be configured to count until a reference is reached and then either starts a new time count immediately or continues to run. The free run/restart (FRR) bit of the corresponding TMR selects each mode. Upon reaching the reference value, the corresponding TER bit is set, and an interrupt is issued if the output reference interrupt enable (ORI) bit in TMR is set.

Each timer may output a signal on the timer output ($\overline{\text{TOUT1}}$ or $\overline{\text{TOUT2}}$) pin when the reference value is reached, as selected by the output mode (OM) bit of the corresponding TMR. This signal can be an active-low pulse or a toggle of the current output. The output can also be used as an input to the other timer, resulting in a 32-bit timer.

Each timer has a 16-bit TCR, which is used to latch the value of the counter when a defined transition (of TIN1 or TIN2) is sensed by the corresponding

input capture edge detector. The type of transition triggering the capture is selected by the capture edge and enable interrupt (CE) bits in the corresponding TMR. Upon a capture or reference event, the corresponding TER bit is set, and a maskable interrupt is issued.

3.5.2.1 TIMER MODE REGISTER (TMR1, TMR2). TMR1 and TMR2 are identical 16-bit registers. TMR1 and TMR2, which are memory-mapped read-write registers to the user, are cleared by reset.



RST — Reset Timer

This bit performs a software reset of the timer identical to that of an external reset.

0 = Reset timer (software reset)
1 = Enable timer

ICLK — Input Clock Source for the Timer

00 = Stop count
01 = Master clock
10 = Master clock divided by 16
11 = Corresponding TIN pin, TIN1 or TIN2 (falling edge)

FRR — Free Run/Restart

0 = Free run — timer count continues to increment after the reference value is reached.
1 = Restart — timer count is reset immediately after the reference value is reached.

ORI — Output Reference Interrupt Enable

0 = Disable interrupt for reference reached (does not affect interrupt on capture function)
1 = Enable interrupt upon reaching the reference value

OM — Output Mode

0 = Active-low pulse for one CLKO clock cycle (60 ns at 16.67 MHz)
1 = Toggle output

NOTE

After reset, the $\overline{\text{TOUT}}$ signal begins in a high state, but is not available externally until the PBCNT register is configured for this function.

CE — Capture Edge and Enable Interrupt
00 = Disable interrupt on capture event
01 = Capture on rising edge only and enable interrupt on capture event
10 = Capture on falling edge only and enable interrupt on capture event
11 = Capture on any edge and enable interrupt on capture event

PS — Prescaler Value
The prescaler is programmed to divide the clock input by values from 1 to 256. The value 00000000 divides the clock by 1; the value 11111111 divides the clock by 256.

3.5.2.2 TIMER REFERENCE REGISTERS (TRR1, TRR2). Each TRR is a 16-bit register containing the reference value for the timeout. TRR1 and TRR2 are memory-mapped read-write registers.

When working in the MC68008 mode (BUSW is low), writing the high byte of TRR1 and TRR2 will disable the timer's compare logic until the low byte is written.

TRR1 and TRR2 are set to all ones by reset. The reference value is not "reached" until TCN increments to equal TRR.

3.5.2.3 TIMER CAPTURE REGISTERS (TCR1, TCR2). Each TCR is a 16-bit register used to latch the value of the counter. TCR1 and TCR2 appear as memory-mapped read-only registers to the user.

When working in the MC68008 mode (BUSW is low), reading the high byte of TCR1 and TCR2 will disable the timer's capture logic until the low byte is read.

TCR1 and TCR2 are set to all ones by reset.

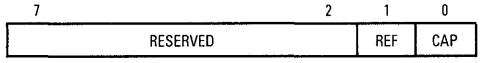
3.5.2.4 TIMER COUNTER (TCN1, TCN2). TCN1 and TCN2 are 16-bit up-counters. Each is memory-mapped and can be read by the user. A read cycle to TCN1 and TCN2 yields the current value of the timer and does not affect the counting operation.

When working in the MC68008 mode (BUSW is low), reading the high byte of TCN1 and TCN2 will latch the low byte into a temporary register; a subsequent read cycle on the low byte yields the value of the temporary register.

A write cycle to TCN1 and TCN2 causes it and the corresponding prescaler to be reset. In MC68008 mode (BUSW is low), a write cycle to either the high or low byte will reset the respective counter and the prescaler.

3.5.2.5 TIMER EVENT REGISTERS (TER1, TER2). Each TER is an 8-bit register used to report events recognized by any of the timers. On recognition of an event, the timer will set the appropriate bit in the TER, regardless of the corresponding interrupt enable bits (ORI and CE) in the TMR. TER1 and TER2, which appear to the user as memory-mapped registers, may be read at any time.

A bit is reset by writing a one to that bit (writing a zero does not affect a bit's value). More than one bit may be reset at a time. Both bits must be reset before the timer will negate the INRQ to the interrupt controller. This register is cleared by reset.



CAP — Capture Event
The counter value has been latched into the TCR. The CE bits in the TMR are used to enable the interrupt request caused by this event.

REF — Output Reference Event
The counter has reached the TRR value. The ORI bit in the TMR is used to enable the interrupt request caused by this event.

Bits 7–2 — Reserved for future use.

3.5.3 Watchdog Timer

A watchdog timer is used to protect against system failures by providing a means to escape from unexpected input conditions, external events, or programming errors. Timer 3 may be used for this purpose. Once started, the watchdog timer must be cleared by software on a regular basis so that it never reaches its timeout value. Upon reaching the timeout value, the assumption may be made that a system failure has occurred, and steps can be taken to recover or reset the system.

3.5.3.1 WATCHDOG TIMER OPERATION. The watchdog timer counts from zero to a maximum of 32768 (16.67 seconds at 16.00 MHz) with a resolution or step size of 8192 clock periods (0.5 ms at 16.00 MHz). This timer uses a 16-bit counter with an 8-bit prescaler value.

The watchdog timer uses the main clock divided by 16 as the input to the prescaler. The prescaler circuitry divides the clock input by a fixed value of 256. The output of this prescaler circuitry is connected to the input of the 16-bit counter. Since the least significant bit of the WCN is not used in the comparison with the WRR reference value, the effective value of the prescaler is 512.

The timer counts until the reference value is reached and then starts a new time count immediately. Upon reaching the reference value, the counter asserts the $\overline{\text{WDOG}}$ output for a period of 16 master clock (CLKO) cycles, and issues an interrupt to the interrupt controller. The value of the timer can be read any time.

To use the watchdog function directly with the M68000 core, the timer 3 open-drain output pin ($\overline{\text{WDOG}}$) can be connected externally to the $\overline{\text{IPL2}}$–$\overline{\text{IPL0}}$ pins to generate a level 7 interrupt (normal mode), to IRQ7 (dedicated mode), or to the $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ pin. After a total system reset, the $\overline{\text{WDOG}}$ pin function is active on pin PB7.

The watchdog timer has an 8-bit prescaler that is not accessible to the user, a read-only 16-bit counter, and a reference register (WRR).

3.5.3.2 WATCHDOG REFERENCE REGISTER (WRR). WRR is a 16-bit register containing the reference value for the timeout. The EN bit of the register enables the timer. WRR appears as a memory-mapped read-write register to the user.

When operating in the MC68008 mode (BUSW is low), writing to the high byte of WRR will disable the timer compare logic until the low byte is written.

Reset initializes the register to $FFFF, enabling the watchdog timer and setting it to the maximum timeout period. This causes a timeout to occur if there is an error in the boot program.

3.5.3.3 WATCHDOG COUNTER (WCN). WCN is a 16-bit up-counter, appears as a memory-mapped register, and may be read at any time. Clearing EN causes the counter to be reset and disables the count operation.

A read cycle to WCN causes the current value of the timer to be read. When working in MC68008 mode (BUSW is low), reading the high byte of WCN will latch the low byte into a temporary register. When reading the low byte, the temporary register value is read. Reading the timer does not affect the counting operation.

A write cycle to WCN causes the counter and prescaler to be reset. In the MC68008 mode (BUSW is low), a write cycle to either the high or low byte resets the counter and the prescaler. A write cycle should be executed on a regular basis so that the watchdog timer is never allowed to reach the reference value during normal program operation.

3.6 EXTERNAL CHIP-SELECT SIGNALS AND WAIT-STATE LOGIC

The MC68302 provides a set of four programmable chip-select signals. Each chip-select signal has an identical internal structure. For each memory area, the user may also define an internally generated cycle termination signal ($\overline{DTACK}$). This feature eliminates board space that would be necessary for cycle termination logic. The four chip-select signals allow four different classes of memory to be used: e.g., high-speed static RAM, slower dynamic RAM, EPROM, and nonvolatile RAM.

The chip-select block diagram is shown in Figure 3-6.

The chip-select logic is active for memory cycles generated by internal bus masters (M68000 core, IDMA, SDMA), or external bus masters. These signals are driven externally on the falling edge of $\overline{AS}$ and are valid shortly after $\overline{AS}$ goes low.

The user should not normally program more than one chip-select line to the same area. When this occurs, the address compare logic will set address decode conflict (ADC) in the system control register (SCR) and generate $\overline{BERR}$ if address decode conflict enable (ADCE) is set. Only one chip-select line will be driven because of internal line priorities. $\overline{CS0}$ has the highest priority, and $\overline{CS3}$ the lowest. $\overline{BERR}$ will not be asserted on write accesses to the chip-select registers.

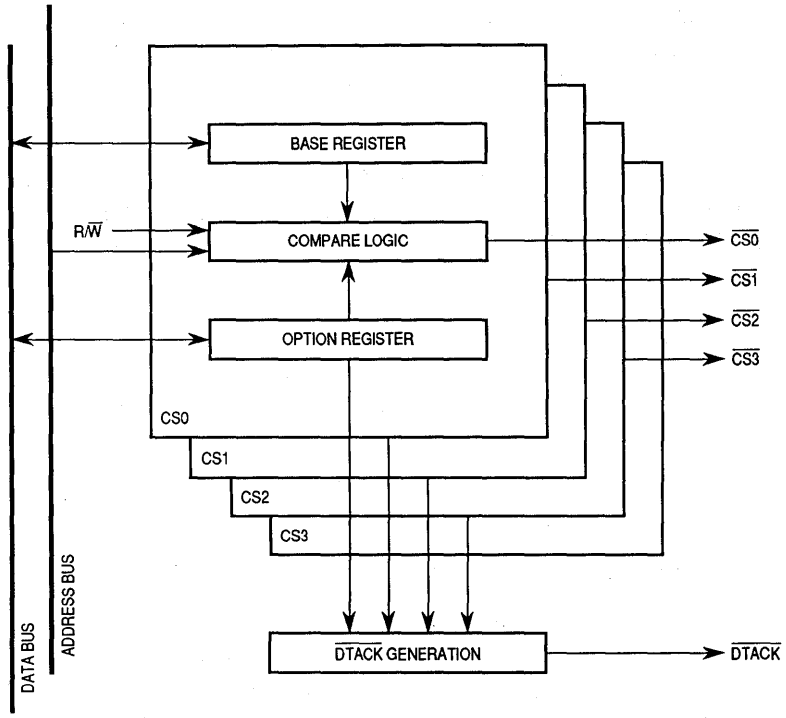


Figure 3-6. Chip-Select Block Diagram

When a bus master attempts to write to a read-only location, the chip-select logic will set write protect violation (WPV) in the SCR and generate $\overline{BERR}$ if write protect violation enable (WPVE) is set. The $\overline{CS}$ line will not be asserted.

NOTE

The chip-select logic is reset only on total system reset (assertion of $\overline{RESET}$ and $\overline{HALT}$). Accesses to the internal RAM and registers, including the system configuration registers (BAR and SCR), will not activate the chip-select lines. This is very convenient for BAR and SCR. See **2.7 MC68302 IMP CONFIGURATION CONTROL** for details. The chip-select logic does not allow an address match during interrupt acknowledge cycles.

3.6.1 Chip-Select Logic Key Features

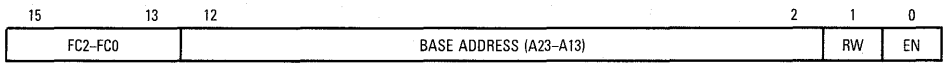
Key features of the chip-select logic are as follows:

- Four Programmable Chip-Select Lines
- Various Block Sizes: 8K, 16K, 32K, 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M, and 16M Bytes
- Read-Only, Write-Only or Read-Write Select
- Internal $\overline{\text{DTACK}}$ Generation with Wait-State Options
- Default Line ($\overline{\text{CS0}}$) to Select an 8K-Boot ROM Containing the Reset Vector and Initial Program

3.6.2 Chip-Select Registers

Each of the four chip-select units has two registers that define its specific operation. These registers include a 16-bit base register (BR) and a 16-bit option register (OR). These registers may be modified by the M68000 core.

3.6.2.1 BASE REGISTER (BR3–BR0). These 16-bit registers consist of a base address field, a read-write bit, and a function code field.



FC2–FC0 — Function Code Field

This field is contained in bits 15–13 of each BR. These bits are used to set the address space function code. The address compare logic uses these bits to determine whether an address match exists within its address space and, therefore, whether to assert the chip-select line.

111 = Not supported; reserved
110 = Value may be used
•
•
•
000 = Value may be used

After system reset, the FC field in BR3–BR0 defaults to supervisor program space (FC = 110) to select a ROM device containing the reset vector. Because of the priority mechanism and the EN bit, only the $\overline{\text{CS0}}$ line is active after a system reset.

NOTE

The FC bits can be masked and ignored by the chip-select logic using CFC in the OR.

Bits 12–2 — Base Address

These bits are used to set the starting address of a particular address space. The address compare logic uses only A23–A13 to cause an address match within its block size.

After system reset, the base address defaults to zero to select a ROM device on which the reset vector resides. All base address values default to zero on system reset, but, because of the priority mechanism, only CS0 will be active.

NOTE

All address bits can be masked and ignored by the chip-select logic through the base address mask in the OR.

RW — Read/Write

0 = The chip-select line is asserted for read operations only.
1 = The chip-select line is asserted for write operations only.

After system reset, this bit defaults to zero (read-only operation).

NOTE

This bit can be masked and ignored by the read-write compare logic, as determined by MRW in the OR. The line is then asserted for both read and write cycles.

On write protect violation cycles (RW = 0 and MRW = 1), $\overline{BERR}$ will be generated if WPVE is set, and WPV will be set.

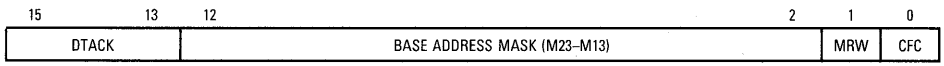
If the write protect mechanism is used by an external master, the R/$\overline{W}$ low to $\overline{AS}$ asserted timing should be 16-ns minimum.

EN — Enable

0 = The chip-select line is disabled.
1 = The chip-select line is enabled.

After system reset, only $\overline{CS0}$ is enabled; $\overline{CS3}$–$\overline{CS1}$ are disabled. In disable CPU mode, $\overline{CS3}$–$\overline{CS0}$ are disabled at system reset.

3.6.2.2 OPTION REGISTERS (OR3–OR0). These four 16-bit registers consist of a base address mask field, a read/write mask bit, a compare function code bit, and a $\overline{\text{DTACK}}$ generation field.



Bits 15–12 — DTACK Field

These bits are used to determine whether $\overline{\text{DTACK}}$ is generated internally with a programmable number of wait states or externally by the peripheral. With internal $\overline{\text{DTACK}}$ generation, zero to six wait states can be automatically inserted before the $\overline{\text{DTACK}}$ pin is asserted as an output (see Table 3-7).

Table 3-7. DTACK Field Encoding

Bits			Description
15	14	13	
0	0	0	No Wait State
0	0	1	1 Wait State
0	1	0	2 Wait States
0	1	1	3 Wait States
1	0	0	4 Wait States
1	0	1	5 Wait States
1	1	0	6 Wait States
1	1	1	External DTACK

When all the bits in this field are set to one, $\overline{\text{DTACK}}$ must be generated externally, and the IMP or external bus master waits for $\overline{\text{DTACK}}$ (input) to terminate its bus cycle. After system reset, the bits of the DTACK field default to six wait states.

The DTACK generator uses the IMP internal clock to generate the programmable number of wait states. For asynchronous external bus masters, the programmable number of wait states is counted directly from the internal clock. When no wait state is programmed (DTACK = 000), the DTACK generator will generate $\overline{\text{DTACK}}$ asynchronously.

The $\overline{\text{CS}}$ lines are asserted slightly earlier for internal IMP master memory cycles than for an external master using the $\overline{\text{CS}}$ lines. Set external master wait state (EMWS) in the SCR whenever these timing differences require an extra memory wait state for external masters.

NOTE

Do not assert $\overline{\text{DTACK}}$ externally when it is programmed to be generated internally.

Bits 12–2 — Base Address Mask

These bits are used to set the block size of a particular chip-select line. The address compare logic uses only the address bits that are not masked (i.e., mask bit set to one) to detect an address match within its block size.

0 = The address bit in the corresponding BR is masked; the address compare logic does not use this address bit.

1 = The address bit in the corresponding BR is not masked; the address compare logic uses this address bit.

For example, for a 64K-byte block, this field should be M13,M14,M15 = 0 with the rest of the base address mask bits equal to one.

After system reset, the bits of the base address mask field default to ones (selecting the smallest block size of 8K) to allow $\overline{\text{CS0}}$ to select the ROM device containing the reset vector.

MRW — Mask Read/Write

0 = The RW bit in the BR is masked. The chip select is asserted for both read and write operations.

1 = The RW bit in the BR is not masked. The chip select is asserted for read-only or write-only operations as programmed by the corresponding RW bit in BR3–BR0.

After system reset, this bit defaults to zero.

CFC — Compare Function Code

0 = The FC bits in the BR are ignored. The chip select is asserted without comparing the FC bits. If the application requires the user to recognize several address spaces (e.g., user space without distinguishing between data and program space), FC bits must be decoded externally.

1 = The FC bits on the BR are compared. The address space compare logic uses the FC bits to assert the $\overline{\text{CS}}$ line.

After system reset, this bit defaults to one.

3.7 ON-CHIP CLOCK GENERATOR

The IMP has an on-chip clock generator that supplies clocks to both the internal M68000 core and peripherals and to an external pin. The clock circuitry uses three dedicated pins: EXTAL, XTAL, and CLKO.

The external clock/crystal (EXTAL) input provides two clock generation options. EXTAL may be used to interface the internal generator to an external crystal (see Figure 3-7). Typical circuit parameters are C1 = C2 = 25 pF and R = 700 kΩ using a parallel resonant crystal. Typical crystal parameters are Co < 10 pF and Rx = 50 Ω.

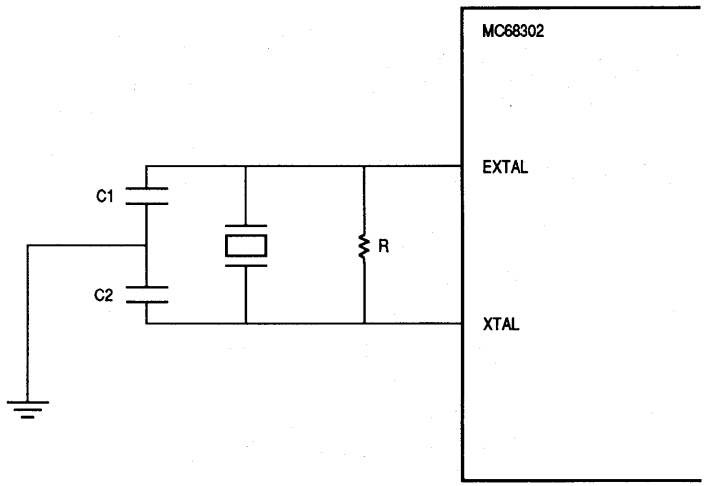


Figure 3-7. Using an External Crystal

EXTAL can also accept a CMOS-level clock input. The crystal output (XTAL) connects the internal crystal generator output to an external crystal. If an external clock is used, XTAL should be left unconnected. The CLKO pin, which drives the high-speed system clock, may be used to synchronize other peripherals to the IMP system clock.

3.8 SYSTEM CONTROL

The IMP system control consists of an SCR that configures the following functions:

- System Status and Control Logic

- $\overline{AS}$ Control During Read-Modify-Write Cycles
- Disable CPU (M68000) Logic
- Bus Arbitration Logic with Low-Interrupt Latency Support
- Hardware Watchdog
- Low-Power (Standby) Modes
- Freeze Control

3.8.1 System Control Register (SCR)

The SCR is a 32-bit register that consists of system status and control bits, a bus arbiter control bit, hardware watchdog control bits, low-power control bits, and freeze select bits. Refer to Figure 3-8 and to the following paragraphs for a description of each bit in this register. The SCR is a memory-mapped read-write register. The address of this register is fixed at $0F4 in supervisor space (FC = 5).

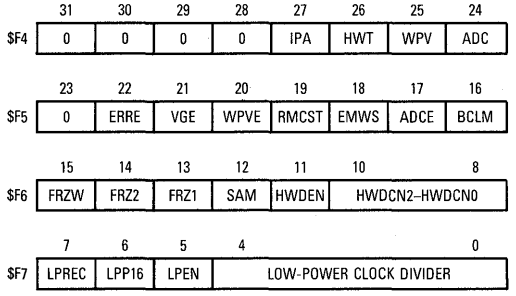


Figure 3-8. System Control Register

3.8.2 System Status Bits

The eight most significant bits of the SCR are used to report events recognized by the system control logic. On recognition of an event, this logic sets the corresponding bit in the SCR. The bits may be read at any time. A bit is reset by one and is left unchanged by zero. More than one bit may be reset at a time.

After system reset (simultaneous assertion of $\overline{RESET}$ and $\overline{HALT}$), these bits are cleared.

IPA — Interrupt Priority Active

This bit is set when the M68000 core has an unmasked interrupt request. When bus clear mask (BCLM) is set, $\overline{\text{BCLR}}$ and the internal bus clear to the IDMA are asserted.

NOTE

If BCLM is set, an interrupt handler will normally clear IPA at the end of the interrupt routine to allow an alternate bus master to regain the bus; however, if BCLM is cleared, no additional action need be taken in the interrupt handler.

In the case of nested interrupts, the user may wish to clear the IPA bit only at the end of the original lower priority interrupt routine to keep $\overline{\text{BCLR}}$ asserted until it completes. To guarantee that this happens and that other pending interrupts at the same original priority level also execute with $\overline{\text{BCLR}}$ continuously asserted, the following technique may be used. Using a parallel I/O line connected to the IRQ1 line, the original priority level interrupt toggles this I/O line just before it executes the RTE instruction, causing a request for a level 1 interrupt. Since this is the lowest interrupt level, this routine will not be executed until all other pending interrupt routines have executed. Then in the level 1 interrupt routine, the IPA bit in the SCR is cleared.

HWT — Hardware Watchdog Timeout

This bit is set when the hardware watchdog (see **3.8.6 Hardware Watchdog**) reaches the end of its time interval; $\overline{\text{BERR}}$ is generated following the watchdog timeout, even if this bit is already set.

WPV — Write Protect Violation

This bit is set when a bus master attempts to write to a location that has RW set to zero (read only) in its associated base register (BR3–BR0). Provided WPVE (bit 20) is set, $\overline{\text{BERR}}$ will be asserted on the bus cycle that sets this bit. If WPV and WPVE are both set when a write protect violation occurs, $\overline{\text{BERR}}$ will still be generated.

ADC — Address Decode Conflict

This bit is set when a conflict has occurred in the chip-select logic because two or more chip-select lines attempt assertion in the same bus cycle. This conflict may be caused by a programming error in which the user-allocated memory areas for each chip select overlap each other. Provided ADCE (bit 17) is set, the occurrence of ADC will cause $\overline{\text{BERR}}$ to be asserted. If this bit is already set when another address decode conflict occurs, $\overline{\text{BERR}}$ will still

be generated. The chip-select logic will protect the IMP from issuing two simultaneous chip selects by employing a priority system.

NOTE

Regardless of the state of the chip-select programming, this bit will not be set and $\overline{\text{BERR}}$ will not be asserted for an address decode conflict occurring during access to a system configuration register. This is provided to guarantee access to the system configuration registers (BAR and SCR) during initialization.

3.8.3 System Control Bits

The system control logic uses six control bits in the SCR.

WPVE — Write Protect Violation Enable
0 = $\overline{\text{BERR}}$ is not asserted when a write protect violation occurs.
1 = $\overline{\text{BERR}}$ is asserted when a write protect violation occurs.

After system reset, this bit defaults to zero.

NOTE

WPV will be set, regardless of the value of WPVE.

RMCST — RMC Cycle Special Treatment
0 = The read-modify-write cycles will be identical to the M68000 RMC cycles ($\overline{\text{AS}}$ will be asserted during the entire cycle). The arbiter will issue $\overline{\text{BG}}$, regardless of the M68000 core $\overline{\text{RMC}}$.
1 = The MC68302 uses $\overline{\text{RMC}}$ to negate $\overline{\text{AS}}$ at the end of the read portion of the RMC cycle and reassert $\overline{\text{AS}}$ at the beginning of the write portion. $\overline{\text{BG}}$ will not be asserted until the end of the write portion.

The assertion of the $\overline{\text{RMC}}$ by the M68000 core is seen by the arbiter and will prevent the arbiter from issuing bus grants until the completion of M68000-initiated RMC activity. After system reset, this bit defaults to zero.

EMWS — External Master Wait State
When EMWS is set and an external master is using the chip-select logic for $\overline{\text{DTACK}}$ generation or is synchronously reading from the internal peripherals (SAM = 1), one additional wait state will be inserted in that memory cycle. Synchronous writes to internal peripherals are always with zero wait states. When EMWS is cleared, all synchronous internal accesses will

be with zero wait states, and the chip-select logic will generate $\overline{\text{DTACK}}$ after the exact programmed number of wait states. The chip-select lines are asserted slightly earlier for internal master memory cycles than for an external master. EMWS should be set whenever these timing differences will necessitate an additional wait state for external masters. After system reset, this bit defaults to zero.

ADCE — Address Decode Conflict Enable

0 = $\overline{\text{BERR}}$ is not asserted by a conflict in the chip-select logic when two or more chip-select lines are programmed to overlap the same area.

1 = $\overline{\text{BERR}}$ is asserted by a conflict in the chip-select logic when two or more chip-select lines are programmed to overlap the same area.

After system reset, this bit defaults to zero.

NOTE

ADC will be set, regardles of the value of ADCE.

BCLM — Bus Clear Mask

0 = The arbiter does not use the M68000 core internal IPEND signal to assert the internal and external bus clear signals.

1 = The arbiter uses the M68000 core internal IPEND signal to assert the internal and external bus clear signals.

After system reset, this bit defaults to zero. If BCLM is set, then the typical maximum interrupt latency is about 78 clocks in a zero-wait-state system. This assumes a standard instruction mix, that the IDMA is just beginning a four-bus-cycle transfer when the interrupt becomes pending, and that an SDMA has an access pending (one bus cycle). Interrupt execution time is 44 clocks and includes the time to execute the interrupt acknowledge cycle, save the status register and PC value on the stack, and then vector to the first location of the interrupt service routine. Thus, the calculation is 78 = 14 (instruction completion) + 20 (DMAs) + 44 (interrupt execution).

SDMA operation is not affected by the BCLM bit. Note that the SDMA accesses only one byte/word of external memory at a time before giving up the bus and that accesses are relatively infrequent. External bus master operation may or may not be affected by the BCLM bit, depending on whether the $\overline{\text{BCLR}}$ signal is used to clear the external master off the bus.

Without using the BCLM bit, the maximum interrupt latency includes the maximum time that the IDMA or external bus master could use the bus in

the worst case. Note that the IDMA can limit its bus usage if its requests are generated internally.

NOTE

The IPA status bit will be set regardless of the BCLM value.

SAM — Synchronous Access Mode

This bit controls how external masters may access the MC68302 peripheral area. This bit is not relevant for applications that do not have external bus masters that access the MC68302. In applications such as disable CPU mode, in which the M68000 core is not operating, the user should note that SAM may be changed by an external master on the first access of the MC68302, but that first write access must be asynchronous with three wait states. (If $\overline{\text{DTACK}}$ is used to terminate bus cycles, this change need not influence hardware.)

0 = Asynchronous accesses. All accesses to the MC68302 internal RAM and registers (including BAR and SCR) by an external master are asynchronous to the MC68302 clock. Read and write accesses are with three wait states, and $\overline{\text{DTACK}}$ is asserted by the MC68302 assuming three-wait-state accesses. This is the default value.

1 = Synchronous accesses. All accesses to the MC68302 internal RAM and registers (including BAR and SCR) must be synchronous to the MC68302 clock. Synchronous read accesses may occur with one wait state if EMWS is also set to one.

3.8.4 Disable CPU Logic (M68000)

The MC68302 can be configured to operate solely as a peripheral to an external processor. In this mode, the on-chip M68000 CPU should be disabled by strapping DISCPU high during system reset ($\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ asserted simultaneously). The internal accesses to the MC68302 peripherals and memory may be asynchronous or synchronous. During synchronous reads, one wait state may be used if required (EMWS bit set). The following pins change their functionality in this mode:

1. $\overline{\text{BR}}$ will be an output from the IDMA and SDMA to the external M68000 bus.
2. $\overline{\text{BG}}$ will be an input to the IDMA and SDMA from the external M68000 bus.
3. $\overline{\text{BCLR}}$ will be an input to the IDMA, but will remain an output from the SDMA.

4. The interrupt controller will output its interrupt request lines ($\overline{IPL0}$, $\overline{IPL1}$, $\overline{IPL2}$) normally sent to the M68000 core, on pins $\overline{IOUT0}$, $\overline{IOUT1}$, and $\overline{IOUT2}$, respectively. $\overline{AVEC}$, $\overline{RMC}$, and $\overline{CS0}$, which share pins with $\overline{IOUT0}$, $\overline{IOUT1}$, and $\overline{IOUT2}$, respectively, are not available in this mode.

DISCPU should remain continuously high during disable CPU mode operation. Although the $\overline{CS0}$ pin is not available as an output from the device in disable CPU mode, it may be enabled to provide $\overline{DTACK}$ generation. In disable CPU mode, BR0 is initially $C000.

Accesses by an external master to the MC68302 RAM and registers may be asynchronous or synchronous to the MC68302 clock. (This feature is actually available regardless of disable CPU mode). See the SAM and EMWS bits in the SCR register for details.

In disable CPU mode, the interrupt controller may be programmed to generate or not generate interrupt vectors during interrupt acknowledge cycles. When multiple MC68302 devices share a single M68000 bus, vector generation should be prevented on all but one MC68302. When using disable CPU mode to implement an interface such as between the MC68020 and the MC68302, vector generation can be enabled. For this purpose, the VGE bit is defined.

VGE — Vector Generation Enable

0 = In disable CPU mode, the MC68302 will not output interrupt vectors during interrupt acknowledge cycles.

1 = In disable CPU mode, the MC68302 will output interrupt vectors for internal level 4 interrupts (and for levels 1, 6, and/or 7 as enabled in the interrupt controller) during interrupt acknowledge cycles.

All MC68302 functionality not expressly mentioned in this section is retained in disable CPU mode and operates identically as before.

NOTE

Even without the use of the disable CPU logic, another processor can be granted access to the IMP on-chip peripherals by requesting the M68000 bus with $\overline{BR}$. See **3.8.5 Bus Arbitration Logic** for further details.

3.8.5 Bus Arbitration Logic

The IMP bus arbiter supports three bus request sources in the following priority:

1. External bus master ($\overline{BR}$ pin)
2. SDMA for the SCCs (six channels)
3. IDMA (one channel)

When one of these sources desires the bus, the M68000 core will be forced off through an internal bus request signal ($\overline{CBR}$) from the bus arbiter to the M68000 core (see Figure 3-9). When the arbiter detects the assertion of the M68000 core bus grant ($\overline{CBG}$) signal, it asserts the requester's bus grant signal according to its priority. Thus, as seen externally, the SDMA and IDMA channels do not affect $\overline{BR}$ and $\overline{BG}$, but only $\overline{BGACK}$ (unless disable CPU mode is used).

The IMP bus arbiter also supports a M68000 core low-interrupt latency option. When the M68000 core processor has an unmasked interrupt request, it asserts an internal interrupt pending signal ($\overline{IPEND}$). The bus arbiter uses this signal according to BCLM in the SCR to assert external ($\overline{BCLR}$) and internal bus-clear (IBCLR) signals. These bus-clear signals allow the M68000 core to eliminate long latencies potentially associated with an external bus master or the IDMA, respectively.

The external $\overline{BCLR}$ is asserted whenever 1) one of the SDMA channels requests the bus when the IDMA is not the bus master or 2) the M68000 core has an unmasked pending interrupt request, provided BCLM in the SCR is set. In this case, $\overline{BCLR}$ will be asserted until the interrupt priority active (IPA) bit in the SCR is cleared. To implement this feature, $\overline{BCLR}$ would be used to force external devices to release bus ownership.

IBCLR to the IDMA is asserted whenever 1) an external bus master requests the bus ($\overline{BR}$ asserted); 2) the M68000 core has an unmasked pending interrupt request, provided BCLM in the SCR is set. In this case, $\overline{BCLR}$ will be asserted until IPA is cleared, 3) the M68000 CPU is disabled, and $\overline{BCLR}$ is asserted.

The IBCLR signal causes the IDMA to release bus ownership at the end of the current operand transfer. IBCLR is not routed to the SDMA channels since they always release bus ownership after one operand transfer.

$\overline{RMC}$ is issued by the M68000 core and can be used by the internal bus arbiter to delay issuance of $\overline{BG}$ during read-modify-write cycles. This is controlled by the RMCST bit in the SCR. Otherwise, the MC68000/MC68008 core may be forced off the bus after any bus cycle.

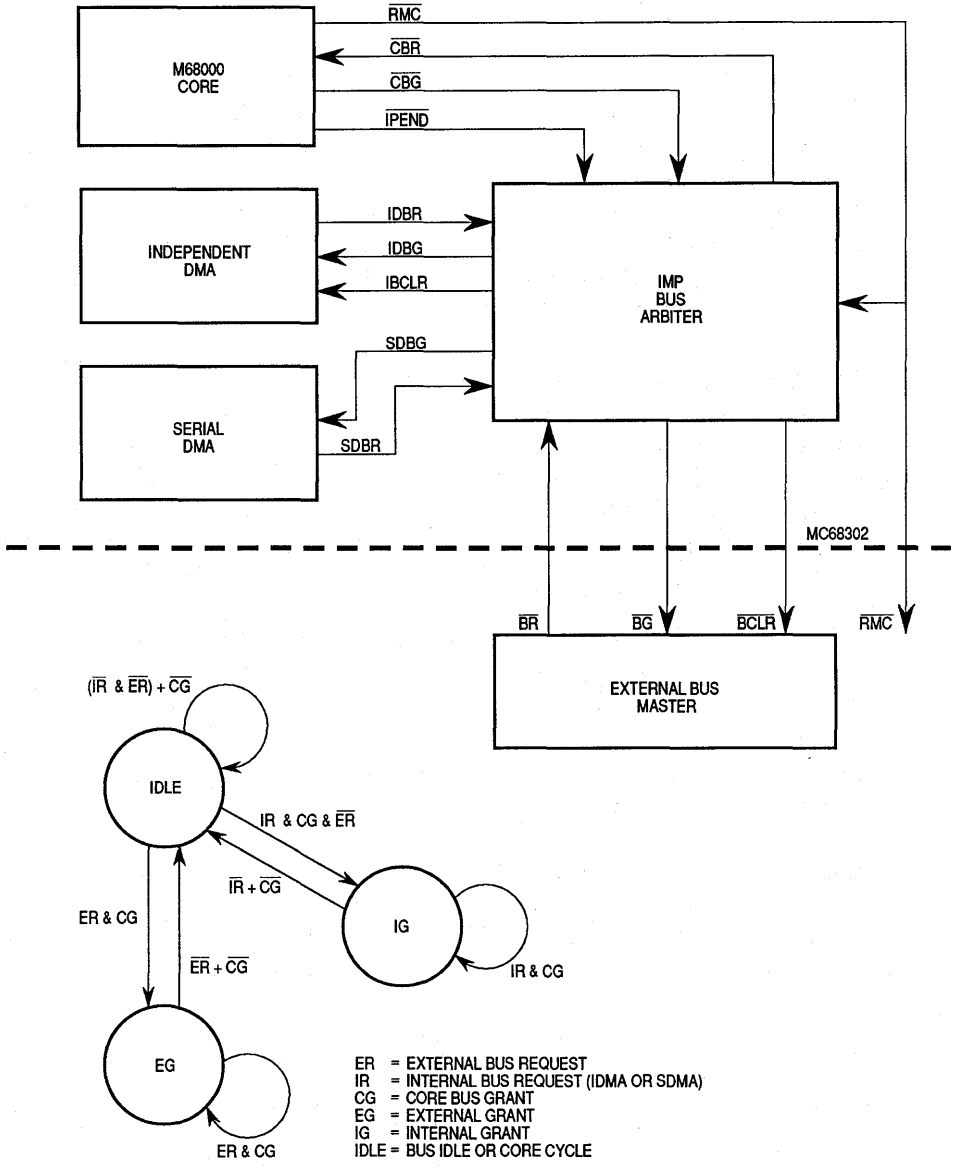


Figure 3-9. IMP Bus Arbiter

3.8.6 Hardware Watchdog

The hardware watchdog logic is used to assert $\overline{\text{BERR}}$ and set HWT when a bus cycle is not terminated by $\overline{\text{DTACK}}$ and after a programmable number of clock cycles has elapsed. The hardware watchdog logic has a 10-bit down-counter and a 4-bit prescaler. When enabled, the watchdog timer commences counting clock cycles as $\overline{\text{AS}}$ is asserted (for internal or external bus masters). The count is terminated normally by the negation of $\overline{\text{AS}}$; however, if the count reaches zero before $\overline{\text{AS}}$ is negated, $\overline{\text{BERR}}$ will be asserted until $\overline{\text{AS}}$ is negated.

The hardware watchdog logic uses four bits in the SCR.

HWDEN — Hardware Watchdog Enable
- 0 = The hardware watchdog is disabled.
- 1 = The hardware watchdog is enabled.

After system reset, this bit defaults to one to enable the hardware watchdog.

HWDCN2–HWDCN0 — Hardware Watchdog Count 2–0
- 000 = $\overline{\text{BERR}}$ is asserted after 128 clock cycles (8 μs, 16-MHz clock)
- 001 = $\overline{\text{BERR}}$ is asserted after 256 clock cycles (16 μs, 16-MHz clock)
- 010 = $\overline{\text{BERR}}$ is asserted after 512 clock cycles (32 μs, 16-MHz clock)
- 011 = $\overline{\text{BERR}}$ is asserted after 1K clock cycles (64 μs, 16-MHz clock)
- 100 = $\overline{\text{BERR}}$ is asserted after 2K clock cycles (128 μs, 16-MHz clock)
- 101 = $\overline{\text{BERR}}$ is asserted after 4K clock cycles (256 μs, 16-MHz clock)
- 110 = $\overline{\text{BERR}}$ is asserted after 8K clock cycles (512 μs, 16-MHz clock)
- 111 = $\overline{\text{BERR}}$ is asserted after 16K clock cycles (1 ms, 16-MHz clock)

After system reset, these bits default to all ones; thus, $\overline{\text{BERR}}$ will be asserted after 1 ms for a 16-MHz system clock.

NOTE

Successive timeouts of the hardware watchdog may vary slightly in length. The counter resolution is 16 clock cycles.

3.8.7 Low-Power (Standby) Modes

The low-power modes are used when no processing is required from the M68000 core processor and when it is desirable to reduce system power consumption to its minimum value. Each low-power mode is entered by issuing the M68000 STOP instruction with low power enable (LPEN) set. This

mode stops processor activity while the system control block reduces the processor clock frequency to achieve minimal power consumption. The user should disable all unrequired on-chip peripherals to obtain the minimal power consumption for a given clock frequency (IMP peripherals consume almost no power when disabled).

The clock frequency selected for the processor during low-power operation determines not only the power consumed but also the choice of recovery method. Provided the low-power frequency is sufficient to maintain the M68000 core status, the programmer may select a nondestructive return to full frequency and full power. This method of return causes the system control block to switch the processor to full frequency after an interrupt from a peripheral. The processor will then handle the interrupt in the usual manner.

In the lowest power-reduction mode, the processor frequency can be reduced to a minimum which is consistent with maintaining bus activity for other system elements. In this mode, after an interrupt occurs (from one of the 16 possible internal sources), the system control block returns the processor to full frequency and drives the M68000 core $\overline{\text{RESET}}$ for 16–32 clock cycles. This return to normal frequency and power is the alternative method that is destructive to the M68000 core status and is equivalent to switching the power onto the M68000 core for the first time. In this lowest power mode, the destructive recovery option (LPREC) must be used because the M68000 core status information is lost as soon as the mode is entered. The dual-port RAM and internal peripherals, however, retain their states.

The low-power logic uses eight bits in the SCR.

LPCD4–LPCD0 — Low-Power Clock Divider Selects
The low-power clock divider select bits (LPCD4–LPCD0) specify the divide ratio of the low-power clock divider equal to LPCD4–LPCD0 + 1. The system clock is divided by 2, then divided by the clock divider value (1 to 32). Thus, a divide ratio of 2 to 64 (LPCD4–LPCD0 0 to 31) can be selected.

After a sytem reset, these bits default to zero.

LPEN — Low-Power Enable
0 = The low-power modes are disabled.
1 = The low-power modes are enabled.

After a system reset, this bit defaults to zero to disable the low-power modes.

LPP16 — Low-Power Clock Prescale Divide by 16

0 = The low-power clock divider input clock is the main clock.

1 = The low-power clock divider input clock is the main clock divided by 16. Thus, a divide ratio of 32 to 1024 (LPCD4–LPCD0 0 to 31) can be selected.

After a system reset, this bit defaults to zero.

LPREC — Low-Power Recovery

0 = Nondestructive recovery from low power. The processor returns to full frequency and then proceeds using currently held status value. This is called low-power mode.

1 = Destructive recovery from low power. The processor returns to full frequency and then drives $\overline{\text{RESET}}$ for 16 clock cycles. This is called lowest power mode.

After a system reset, the bit defaults to zero.

3.8.8 Freeze Control

Used to freeze the activity of selected peripherals, $\overline{\text{FRZ}}$ is useful for system debugging purposes. $\overline{\text{FRZ}}$ should be negated during system reset. When $\overline{\text{FRZ}}$ is asserted:

- The CP main controller freezes its activity as long as $\overline{\text{FRZ}}$ remains asserted. No new interrupt requests and no memory accesses (internal or external) will occur, and the main controller will not access the serial channels.
- The IDMA completes any bus cycle that is in progress (after $\overline{\text{DTACK}}$ is asserted) and releases bus ownership. No further bus cycles will be started as long as $\overline{\text{FRZ}}$ remains asserted.
- Each timer can be programmed to freeze by setting the appropriate bit in the SCR. The selected timers will freeze their activity (count, capture) as long as $\overline{\text{FRZ}}$ remains asserted.

FRZ1 — Freeze Timer 1 Enable

0 = Freeze timer 1 logic is disabled.

1 = Freeze timer 1 logic is enabled.

After system reset, this bit defaults to zero.

FRZ2 — Freeze Timer 2 Enable
0 = Freeze timer 2 logic is disabled.
1 = Freeze timer 2 logic is enabled.

After system reset, this bit defaults to zero.

FRZW — Freeze Watchdog Timer Enable
0 = Freeze watchdog timer logic is disabled.
1 = Freeze watchdog timer logic is enbled.

After system reset, this bit defaults to zero.

No other MC68302 peripherals are directly affected by the freeze logic; however, consequential errors such as receiver overruns in the SCC FIFOs may occur due to the CP main controller being disabled.

3.9 DYNAMIC RAM REFRESH CONTROLLER

The communications processor (CP) main (RISC) controller may be configured to handle the dynamic RAM (DRAM) refresh task without any intervention from the M68000 core and without any external glue-logic. Use of this feature requires a timer or SCC baud rate generator (either from the MC68302 or externally), the I/O pin PB8, and two transmit buffer descriptors from SCC2 (Tx BD6 and Tx BD7).

The DRAM refresh controller routine executes in 25 clock cycles using about 10 percent of the microcontroller bandwidth and about 5 percent of the M68000 bus bandwidth assuming a refresh cycle every 15.625 μs. The refresh cycle will not be executed during a period that a bus exception (i.e., reset, halt, or bus error) is active. The refresh cycle is a standard M68000-type read cycle (an SDMA read cycle). It does not generate row address strobe (RAS) and column address strobe (CAS) to the external DRAM. Use of the DRAM refresh controller will slightly reduce the maximum possible serial data rates of the SCCs.

3.9.1 Hardware Setup

An output of timer 1 or timer 2 (the $\overline{\text{TOUT}}$ pin) or one of the SCC's baud rate generator outputs should be connected externally to PB8. A high-to-low transition on this edge causes a request to be generated to the main controller to perform one refresh cycle. The external RISC request enable (ERRE) bit in the SCR must be set to configure the DRAM refresh operation; otherwise,

PB8 operates normally. The DRAM refresh request takes priority over all SCC channels and commands given to the CP command register.

ERRE — External RISC Request Enable
0 = Normal operation.
1 = When this bit is set, a high-to-low transition on PB8 causes the CP to execute the DRAM refresh routine.

3.9.2 Initialization

The user should first initialize the refresh routine parameters in the SCC2 parameter RAM. These parameters are the DRAM low starting address, the DRAM high starting address, the DRAM address increment step (number of bytes in a row), the count (number of rows), and a temporary count. Next, the ERRE bit in the SCR register should be set. Then, upon every high-to-low transition of PB8, the refresh routine executes one refresh (read) cycle.

3.9.3 Refresh Request Calculations

A typical 1-Mbyte DRAM needs one refresh cycle every 15.625 μs. The DRAM refresh controller is configured to execute one refresh cycle per request; thus, the PB8 pin should see a high-to-low transition every 15.625 μs. This is once every 260 cycles for a 16.67-MHz clock. Note that one refresh per request minimizes the speed loss on the SCC channels.

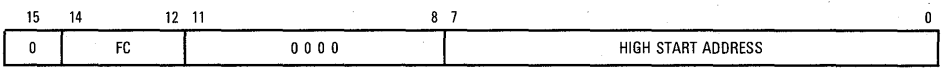
3.9.4 DRAM Refresh Memory Map

The DRAM refresh memory map replaces the SCC2 Tx BD 6 and Tx BD 7 structures in the parameter RAM. The wrap bit must therefore be set in SCC2 Tx BD 5 so that only six Tx BDs are used for SCC2. These parameters should be written before the DRAM refresh controller receives its first request, but may be read at any time. They are undefined at reset.

Address	Name	Width	Description
Base+570*	DRAM_High	Word	Dynamic RAM High Address and FC
Base+572*	DRAM_Low	Word	Dynamic RAM Low Address
Base+574*	INCREMENT	Word	Increment Step (number of bytes/row)
Base+576*	COUNT	Word	RAM Refresh Cycle Count (number of rows)
Base+578	T_ptr_H	Word	Temporary Refresh High Address and FC
Base+57A	T_ptr_L	Word	Temporary Refresh Low Address
Base+57C*	T_count	Word	Temporary Refresh Cycles Count
Base+57E	RESERVED	Word	Reserved

*Initialized by the user (M68000 core).

DRAM_High — Dynamic RAM High Address and Function Codes



This 16-bit parameter contains the dynamic RAM address space function code output during the refresh cycle and the high eight bits of the dynamic RAM starting address. This parameter should be initialized by the user before activating the refresh routine.

DRAM_Low — Dynamic RAM Low Address

This 16-bit parameter contains the lower 16 bits of the dynamic RAM starting address. This parameter should be initialized by the user before activating the refresh routine.

INCREMENT — Increment Step

This 16-bit parameter contains the number of bytes in a row. The refresh routine will increment its pointer with this parameter value every refresh cycle. This parameter should be initialized by the user before activating the refresh routine.

COUNT — RAM Refresh Cycle Count

This 16-bit parameter contains the number of rows in the DRAM. The refresh routine will execute the COUNT number of refresh cycles before wrapping back to the RAM base address. This parameter should be initialized by the user before activating the refresh routine.

T_ptr_H and T_prt_L — Temporary Pointer High and Low

These two 16-bit parameters contain the next refresh cycle address and function code to be used by the CP. They correspond to DRAM_High and DRAM_Low, respectively.

T_count — Temporary Count

This 16-bit parameter contains the number of refresh cycles that the DRAM refresh controller must still perform before it will wrap to the beginning of the DRAM. This parameter shoud be initialized to zero by the user before activating the refresh routine.

3.9.5 DRAM Refresh Controller Bus Timing

The DRAM refresh controller bus cycles are actually SDMA accesses (see **4.2 SDMA CHANNELS** for more details). All timings, signals, and arbitration characteristics of SDMA accesses apply to the DRAM refresh controller accesses. For example, DRAM refresh cycles activate the $\overline{\text{BCLR}}$ signal, just like the SDMA. Note that for debugging purposes, the function code bits may be used to distinguish DRAM refresh cycles from SDMA cycles.

SECTION 4
COMMUNICATIONS PROCESSOR (CP)

The CP includes the following modules:

- Main Controller (RISC Processor)
- Six Serial Direct Memory Access (SDMA) Channels
- A Command Set Register
- Serial Channels Physical Interface Including:
  Motorola Interchip Digital Link (IDL)
  General Circuit Interface (GCI), also known as IOM-2
  Pulse Code Modulation (PCM) Highway Interface
  Nonmultiplexed Serial Interface (NMSI) Implementing Standard Modem Signals
- Three Independent Full-Duplex Serial Communication Controllers (SCCs) Supporting the Following Protocols:
  High-Level/Synchronous Data Link Control (HDLC/SDLC)
  Universal Asynchronous Receiver Transmitter (UART)
  Binary Synchronous Communication (BISYNC)
  Synchronous/Asynchronous Digital Data Communications Message Protocol (DDCMP)
  Transparent Modes
  V.110 Rate Adaption
- Serial Communication Port (SCP) for Synchronous Communication
- Two Serial Management Controllers (SMCs) to Support the IDL and GCI Management Channels

4.1 MAIN CONTROLLER

The CP main controller is a RISC processor that services the three SCCs, the SCP, and the SMCs. Its primary reponsibilities are to work with the serial channels to implement the user-chosen protocol and to manage the SDMA channels that transfer data between the SCCs and memory. The CP main controller also executes commands issued by the M68000 core (or an external processor) and generates interrupts to the interrupt controller.

The operation of the main controller is transparent to the user, executing microcode located in a private internal ROM. Commands may be explicitly written to the main controller by the M68000 core through the CP command register. Additionally, commands and status are exchanged between the main controller and the M68000 core through the buffer descriptors of the serial channels. Also, a number of protocol-specific parameters are exchanged through several parameter RAM areas in the internal dual-port RAM.

Simultaneous access of the dual-port RAM by the main controller and the M68000 core (or external processor) is prevented. During a standard four-clock cycle access of the dual-port RAM by the M68000 core, three main controller accesses are permitted. The main controller is delayed one clock cycle, at most, in accessing the dual-port RAM.

The main controller has a priority scheduler that determines which microcode routine is called when more than one internal request is pending. Requests are serviced in the following priority:

1. DRAM Refresh Controller
2. Commands Issued to the Command Register
3. SCC1 Receive Channel
4. SCC1 Transmit Channel
5. SCC2 Receive Channel
6. SCC2 Transmit Channel
7. SCC3 Receive Channel
8. SCC3 Transmit Channel
9. SMC1 Receive Channel
10. SMC1 Transmit Channel
11. SMC2 Receive Channel
12. SMC2 Transmit Channel
13. SCP Receive Channel
14. SCP Transmit Channel

For details on the DRAM refresh controller, see **3.8.9 DRAM Refresh Controller**.

4.2 SDMA CHANNELS

Six serial (SDMA) channels are associated with the three full-duplex SCCs. Each channel is permanently assigned to service the receive or transmit operation of one of the SCCs and is always available, regardless of the SCC protocol chosen.

The SDMA channels allow flexibility in managing the data flow. The user can, on a buffer-by-buffer basis, determine whether data should be transferred between the SCCs and external memory or between the SCCs and on-chip dual-port RAM. This choice is controlled in each SCC buffer descriptor. The SCC to external memory path bypasses the dual-port RAM by allowing the SDMA channel to arbitrate for the M68000 bus directly. The SCC to dual-port RAM path saves external memory and eliminates the need to arbitrate for the bus.

The SDMA channels implement bus-cycle-stealing data transfers controlled by microcode in the CP main controller. Having no user-accessible registers associated with them, the channels are effectively controlled by the choice of SCC configuration options.

When one SDMA channel needs to transfer data to or from external memory, it will request the M68000 bus with the internal signal SDBR, wait for SDBG, and then only assert the external signal $\overline{\text{BGACK}}$ (see **3.8.5 Bus Arbitration Logic**). It remains the bus master for only one bus cycle. The six SDMA channels have priority over the IDMA controller. If the IDMA is bus master when an SDMA channel needs to transfer over the M68000 bus, the SDMA will steal a cycle from the IDMA with no arbitration overhead while $\overline{\text{BGACK}}$ remains continuously low and $\overline{\text{BCLR}}$ remains high. Each SDMA channel may be programmed with a separate function code if desired. Where possible, the SDMA channel will read 16 bits at a time. It will write 8 bits at a time except during the HDLC or transparent protocols where it writes 16 bits at a time. Each bus cycle is a standard M68000-type bus cycle. The chip select and wait state generation logic on the MC68302 may be used with the SDMA, if desired.

NOTE

When external buffer memory is used, the M68000 bus arbitration delay must be less than what would cause the SCC internal FIFOs to overrun or underrun. This aspect is discussed in more detail in **4.5 SERIAL COMMUNICATIONS CONTROLLERS (SCCs)** and in **APPENDIX A SCC PERFORMANCE**.

The SDMA will assert the external $\overline{\text{BCLR}}$ pin when it requests the bus. $\overline{\text{BCLR}}$ can be used to clear an external bus master from the external bus, if desired. For instance, $\overline{\text{BCLR}}$ can be connected through logic to the external master's $\overline{\text{HALT}}$ signal, and then be deasserted externally when the external master's $\overline{\text{AS}}$ signal is deasserted. $\overline{\text{BCLR}}$ as seen from the MC68302 is deasserted by the SDMA during its access to memory.

The SDMA keeps the M68000 bus for only one operand (8 or 16 bit) transfer before giving it up. If the SDMA begins a word operation on an 8-bit bus, it will complete that operation before giving up the bus, unless a bus exception such as reset, halt, or bus error occurs. Reset suspends and resets all SDMA activity. Halt suspends activity after the current bus cycle. For information on a bus error during an SDMA access, see **4.5.8.4 BUS ERROR ON SDMA ACCESS**.

SDMA operation occurs regardless of the value of the BCLM bit in the SCR, and thus is not affected by the low interrupt latency mechanism.

4.3 COMMAND SET

The M68000 core processor (or an external processor) issues commands to the CP by writing to the CP command register. The M68000 core should set the least significant bit (FLG) of the command register when it issues commands. The CP clears FLG after completing the command to indicate to the M68000 core that it is ready for the next command. Subsequent commands to the CP command register may be given only after FLG is cleared. The software reset (issued with the RST bit) command may be given regardless of the state of FLG, but the M68000 core should still set FLG when setting RST.

The command register, an 8-bit, memory-mapped, read-write register, is cleared by reset.

7	6	5	4	3	2	1	0
RST	GCI	OPCODE		—	CH. NUM.		FLG

RST — Software Reset Command

This bit is set by the M68000 core and cleared by the CP. This command is useful when the M68000 core wants to reset the registers and parameters for all channels (SCCs, SCP, SMCs). The main controller in the CP detects this command by hardware and resets the entire CP. Setting RST leaves the CP in the same state as a hardware reset. Note that this operation does not clear IPR bits in the interrupt controller.

GCI — CGI Commands

1 = When set in conjunction with the opcode bits, this bit allows the two GCI commands (ABORT REQUEST and TIMEOUT) to be generated. The accompanying CH. NUMBER should be 10, and FLG should be set.

OPCODE — Command Opcode (GCI Mode Only)
These bits are set by the M68000 core to define the specific GCI command. See **4.7 SERIAL MANAGEMENT CONTROLLERS (SMCs)** for more details.
00 = TRANSMIT ABORT REQUEST; the GCI receiver sends an abort request on the E bit.
01 = TIMEOUT Command
10 = Reserved
11 = Reserved

0 = When the GCI bit is zero, the commands are as follows:

OPCODE — Command Opcode
These bits are set by the M68000 core to define the specific SCC command. The command execution details depend on the protocol chosen.

The detailed command description for the UART protocol is presented in **4.5.11 UART Controller**.

The detailed command description for the HDLC protocol is presented in **4.5.12 HDLC Controller**.

The detailed command description for the BISYNC protocol is presented in **4.5.13 BISYNC Controller**.

The detailed command description for the DDCMP protocol is presented in **4.5.14 DDCMP Controller**.

The detailed command description for the V.110 protocol is presented in **4.5.15 V.110 Controller**.
00 = STOP TRANSMIT Command
01 = RESTART TRANSMIT Command
10 = ENTER HUNT MODE Command
11 = RESET RECEIVER BCS GENERATOR (used only in BISYNC mode)

Bit 3 — Reserved bit; should be set to zero.

CH. NUM. — Channel Number

These bits are set by the M68000 core to define the specific SCC channel that the command is to operate upon.

00 = SCC1
01 = SCC2
10 = SCC3
11 = Reserved

FLG — Command Semaphore Flag

The bit is set by the M68000 core and cleared by the CP.

0 = The CP is ready to receive a new command.
1 = The command register contains a command that the CP is currently processing. The CP clears this bit at the end of command execution or after reset.

4.3.1 Command Execution Latency

Commands are executed at a priority higher than the SCCs, but less than the priority of the DRAM refresh controller. The longest command, the ENTER HUNT MODE command, executes in 41 clocks. All other commands execute in less than 20 clocks. The maximum command latency is calculated as follows:

Command execution time (41 or 20) +
25 clocks if DRAM refresh controller is used +
205 clocks if any SCC is enabled with BISYNC; or
165 clocks if any SCC is enabled with Transparent; or
165 clocks if any SCC is enabled with HDLC; or
150 clocks if any SCC is enabled with UART mode; or
140 clocks if any SCC is enabled with DDCMP; else
0

For example, if HDLC and UART modes are used on the SCCs with the DRAM refresh controller operating, the maximum command latency is 41 + 25 + 165 = 231 clocks = 13 μs at 16.67 MHz. The equations assume that the DRAM refresh cycle occurs once during the command latency. Note that commands are typically given only in special error-handling situations and that the typical latency is much less than the worst case value.

4.4 SERIAL CHANNELS PHYSICAL INTERFACE

The serial channels physical interface joins the physical layer serial lines to the three SCCs and the two SMCs. (The separate three-wire SCP interface is described in **4.6 SERIAL COMMUNICATION PORT (SCP)**).

The serial channels physical interface controls both the internal route selection and time-division multiplexing for the full Integrated Services Digital Network (ISDN) bandwidth. It supports four popular buses: IDL, GCI (IOM-2), PCM highway, and NMSI.

The serial interface also supports two testing modes: echo and loopback. Echo mode provides a return signal from the physical interface by retransmitting the signal it has received. The physical interface echo mode differs from the individual SCC echo mode in that it can operate on the entire multiplexed signal rather than just on a particular SCC channel (which may further have particular bits masked). Loopback mode causes the physical interface to receive the same signal it is transmitting. The physical interface loopback mode checks more than the individual SCC loopback mode; it checks the physical interface and the internal channel routes.

When using the IDL or GCI buses, additional control functions in the frame structure are required. These functions are supported in the MC68302 through two SMC channels: SMC1 and SMC2. (For other matters relating to the SMCs, refer to **4.7 SERIAL MANAGEMENT CONTROLLERS (SMCs)**).

Refer to Figure 4-1 for the serial channels physical interface block diagram.

4.4.1 IDL Interface

The IDL interface is a full-duplex ISDN interface used to interconnect a physical layer device (such as the Motorola ISDN S/T transceiver MC145474) to the integrated multiprotocol processor (IMP). Data on five channels (B1, B2, D, A, and M) is transferred in a 20-bit frame every 125 μs, providing 160-kbps full-duplex bandwidth. The IMP is an IDL slave device that is clocked by the IDL bus master (physical layer device). The IMP provides direct connections to the MC145474. Refer to Figure 4-2 for the IDL bus signals.

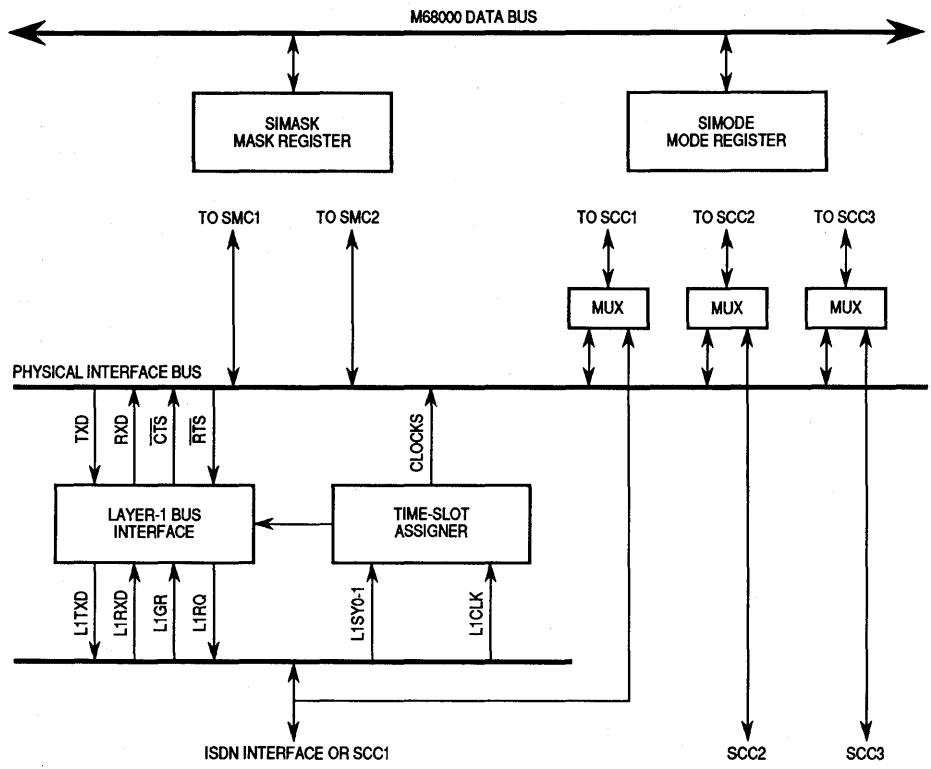


Figure 4-1. Serial Channels Physical Interface Block Diagram

The IMP has two output data strobe lines (SDS1 and SDS2) for selecting either or both the B1 and B2 channels. These signals are used for interfacing devices that do not support the IDL bus. These signals, configured by the SIMASK register, are active only for bits that are not masked. The IDL signals are as follows:

L1CLK	IDL clock; input to the IMP.
L1TXD	IDL transmit data; output from the IMP. Valid only for the bits that are supported by the IDL; three-stated otherwise.
L1RXD	IDL receive data; input to the IMP. Valid for the 20 bits of the IDL; ignored for other signals that may be present.

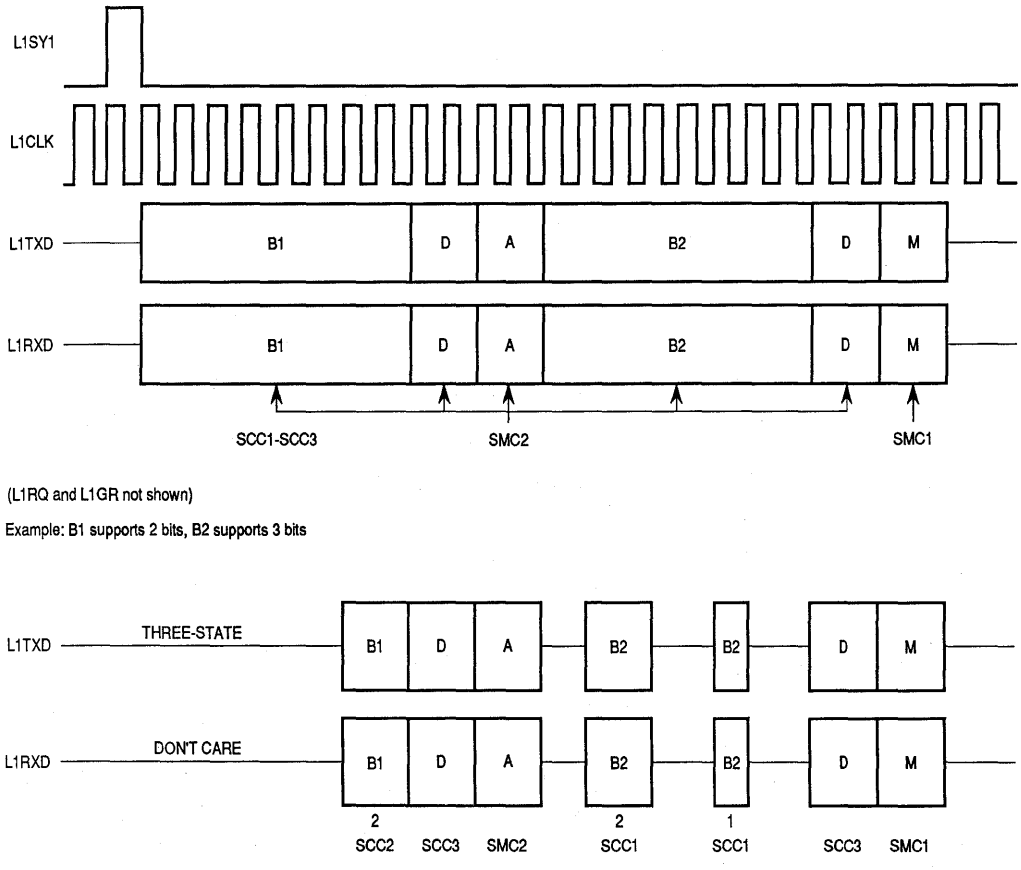


Figure 4-2. IDL Bus Signals

L1SY1	IDL SYNC signal; input to the IMP. This signal indicates that the 20 clock periods following the pulse designate the IDL frame.
L1RQ	Request permission to transmit on the D channel; output from the IMP.
L1GR	Grant permission to transmit on the D channel; input to the IMP.
SDS1	Serial data strobe 1
SDS2	Serial data strobe 2

In addition to the 144-kbps ISDN 2B+D channels, IDL provides channels for maintenance and auxiliary bandwidth. The IDL bus has five channels:

B1	64-kbps Bearer Channel
B2	64-kbps Bearer Channel
D	16-kbps Signaling Channel
M	8-kbps Maintenance Channel
A	8-kbps Auxiliary Channel

The IMP supports all five channels of the IDL bus. The following table shows where each channel can be routed. The two B channels can be concatenated and routed to the same SCC channel.

IDL Channel	Serial Controllers
D	SCC1, SCC2, SCC3
B1	SCC1, SCC2, SCC3
B2	SCC1, SCC2, SCC3
M	SMC1
A	SMC2

The IMP supports the request-grant method for contention detection on the D channel. When the IMP has data to transmit on the D channel, it asserts L1RQ. The physical layer device monitors the physical layer bus for activity on the D channel and indicates that the channel is free by asserting L1GR. The IMP samples the L1GR signal when L1SY1 is asserted. If L1GR is high (active), the IMP transmits the first zero of the opening flag in the first bit of the D channel. If a collision is detected on the D channel, the physical layer device negates L1GR. The IMP then stops its transmission and retransmits the frame when L1GR is asserted again. This is handled automatically for the first two buffers of the frame.

The IDL interface supports the CCITT I.460 recommendation for data rate adaptation. The IDL interface can access each bit of the B channel as an 8-kbps channel. A serial interface mask register (SIMASK) for the B channels specifies which bits are supported by the IDL interface. The receiver will support only the bits enabled by SIMASK. The transmitter will transmit only the bits enabled by the mask register and will three-state L1TXD otherwise.

Refer to Figure 4-2 for an example of supporting two bits in the B1 channel and three bits in the B2 channel.

4.4.2 GCI Interface

The normal mode of the GCI (also known as ISDN-Oriented Modular rev 2.2 (IOM-2)) ISDN bus is fully supported by the IMP. The IMP does not support the Telecom IC (TIC) bus and channels 1 and 2 of the Special Circuit Interface T (SCIT) interface (but does support channel 0). The IMP supports the D channel access control in S/T interface terminals, using the command/indication (C/I) channel 2 for that function.

The GCI bus consists of four lines: two data lines, a clock, and a frame synchronization line. Usually an 8-kHz frame structure defines the various channels within the 256-kbps data rate as indicated in Figure 4-3. However, the interface can also be used in a multiplexed frame structure on which up to eight physical layer devices multiplex their GCI channels. L1SY1 must provide the channel SYNC. In this mode, the data rate would be 2048 kbps.

The GCI clock rate is twice the data rate. The clock rate for the IMP must not exceed the ratio of 1:2.5 serial clock to parallel clock. Thus, for a 16.67-MHz system clock, the serial clock rate must not exceed 6.67 MHz.

The IMP also supports another line for D-channel access control — the L1GR line. This signal is not part of the GCI interface definition and may be used in proprietary interfaces.

NOTE

When the L1GR line is not used, it should be pulled high. The IMP has two data strobe lines (SDS1 and SDS2) for selecting either or both of the B1 and B2 channels and the data rate clock (L1CLK). These signals are used for interfacing devices that do not support the GCI bus. They are configured with the SIMASK register and are active only for bits that are not masked.

The GCI signals are as follows:

L1CLK	GCI clock; input to the IMP.
L1TXD	GCI transmit data; open drain output.

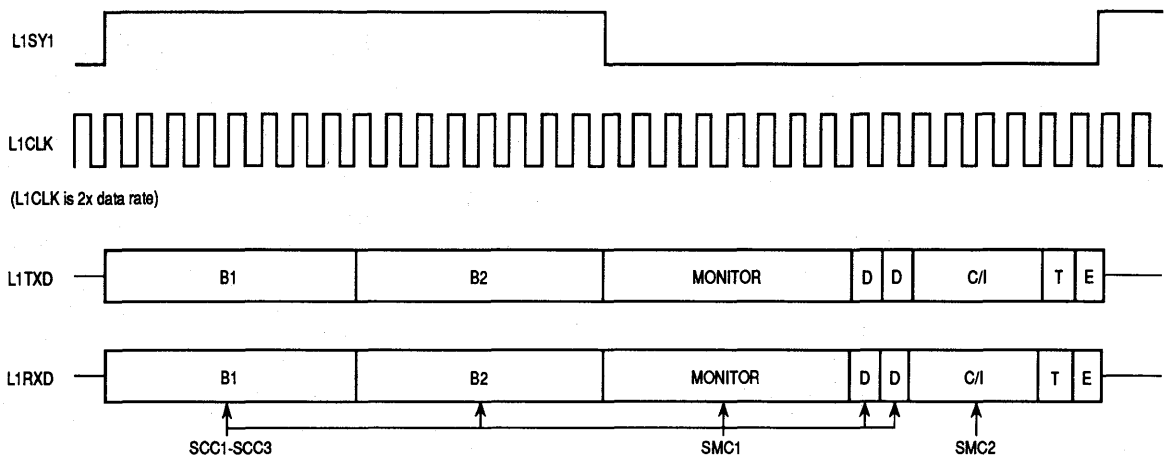


Figure 4-3. GCI Bus Signals

L1RXD	GCI receive data; input to the IMP.
L1SY1	GCI SYNC signal; input to the IMP.
L1GR	Grant permission to transmit on the D channel; input to the IMP.
SDS1	Serial data strobe 1; output from the IMP.
SDS2	Serial data strobe 2; output from the IMP.
GCIDCL	GCI interface data clock; output from the IMP.

The GCI bus has five channels:

B1	64-kbps Bearer Channel
B2	64-kbps Bearer Channel
M	64-kbps Monitor Channel
D	16-kbps Signaling Channel
C/I	48-kbps Command/Indication Channel

In addition to the 144-kbps ISDN 2B + D channels, GCI provides two channels for maintenance and control functions.

The monitor channel is used to transfer data between layer-1 devices and the control unit (i.e., the M68000 core). The command/indication channel is used to control activation/deactivation procedures or for the switching of test loops by the control unit.

The IMP supports all five channels of the GCI channel 0. The following table shows where each channel can be routed. The two B channels can be concatenated and routed to the same SCC channel.

GCI Channel 0	Serial Controllers
D	SCC1, SCC2, SCC3
B1	SCC1, SCC2, SCC3
B2	SCC1, SCC2, SCC3
M	SMC1
C/I	SMC2

The GCI interface supports the CCITT I.460 recommendation for data rate adaptation. The GCI interface can access each bit of the B channel as an 8-kbps channel. The mask register (SIMASK) for the B channels specifies which bits are supported by the GCI interface. The receiver will receive only the bits that are enabled by SIMASK; the transmitter will transmit only the bits that are enabled by SIMASK and will not drive the L1TXD pin otherwise (L1TXD in GCI mode is an open-drain output).

The IMP supports contention detection on the D channel. When the IMP has data to transmit on the D channel, it checks bit 4 of the SCIT C/I channel 2. The physical layer device monitors the physical layer bus for activity on the D channel and indicates with this bit that the channel is free. If a collision is detected on the D channel, the physical layer device sets bit 4 of C/I channel 2 to logic high. The IMP then aborts its transmission and retransmits the frame when this bit is asserted again. This procedure is handled automatically for the first two buffers of a frame. The L1GR line may also be used for access to the S interface D channel. This signal is checked by the IMP, and the physical layer device should indicate that the S interface D channel is free by asserting L1GR.

In the deactivated state, the clock pulse is disabled, and the data line is a logic one. The layer-1 device activates the IMP by enabling the clock pulses and by an indication in the channel 0 C/I channel. The IMP will then report to the M68000 core by a maskable interrupt that a valid indication is in the SMC2 receive buffer descriptor.

When the M68000 core activates the line, it sets SETZ in the serial interface mode (SIMODE) register, causing the data output from L1TXD to become a logic zero. Code 0 (command timing TIM) will be transmitted on channel 0 C/I channel to the layer-1 device until the SETZ is reset. The physical layer device will resume transmitting the clock pulses and will give an indication in the channel 0 C/I channel. The M68000 core should reset SETZ to enable data output.

4.4.3 PCM Highway Mode

The IMP can support three PCM highway channels. In this case, the two SYNC signals (L1SY0 and L1SY1) are used to route the serial lines (L1RXD and L1TXD) into SCC1, SCC2, and SCC3.

As Figure 4-4 shows, the SYNC signals can occur either one clock cycle prior to an 8-bit data channel or can envelope the 8-bit data. When no SYNC is asserted, the L1TXD is three-stated. When both SYNCS are asserted, CH-3

is selected. Each channel (CH-1, CH-2, and CH-3) can be routed independently to separate SCCs. The routing of each channel is determined by DRB-DRA for CH-1, B1RB-B1RA for CH-2, and B2RB-B2RA for CH-3.

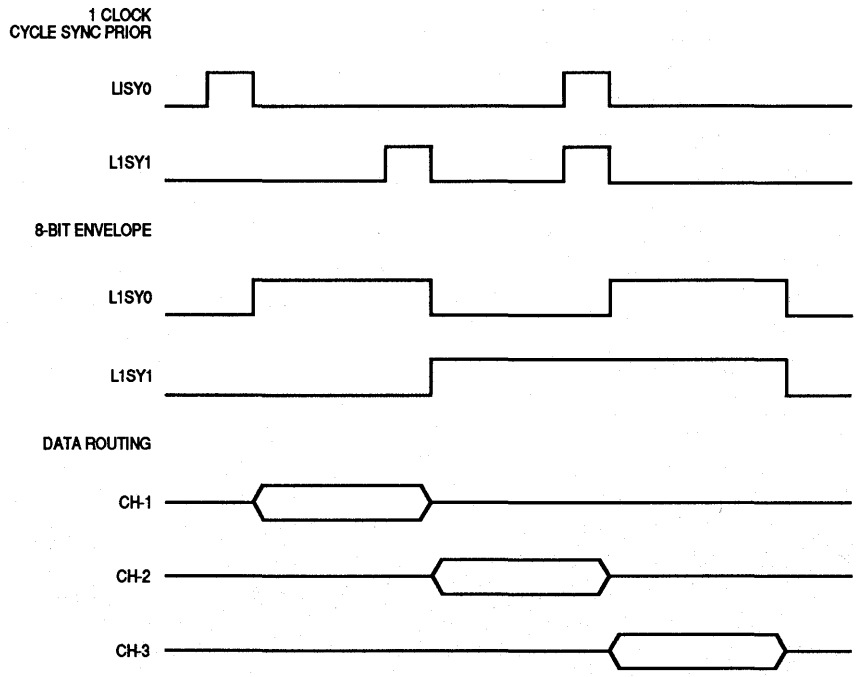


Figure 4-4. PCM Bus Signals

PCM highway mode can be used in other configurations. For instance, when the SYNC signal envelopes the data, the channel data length can be any number of bits. Channel data does not have to be contiguous in the PCM highway but rather can be separated by other channel data. Multiple channel data can also be routed to a single SCC.

4.4.4 Nonmultiplexed Serial Interface (NMSI)

The IMP supports the NMSI with modem signals. In this case, the serial interface connects the seven serial lines of the NMSI1/ISDN interface (RXD1, TXD1, RCLK1, TCLK1, CD1, CTS1, and RTS1) directly to the SCC1 controller. NMSI pins associated with SCC2 and SCC3 can be used as desired or left as general-purpose I/O port pins. See **3.3 PARALLEL I/O PORTS** for an example.

$\overline{RTS}$ is an output, while $\overline{CTS}$ and $\overline{CD}$ are inputs to the transmitter and receiver, respectively. See **4.5.3 SCC Mode Register** for addiitonal information.

4.4.5 Serial Interface Registers

There are two serial interface registers: SIMODE and SIMASK. The SIMODE register is a 16-bit register used to define the serial interface operation modes. The SIMASK register is a 16-bit register used to determine which bits are active in the B1 and B2 channels of ISDN.

4.4.5.1 SERIAL INTERFACE MODE REGISTER (SIMODE). If the IDL or GCI mode is used, this register allows the user to support any or all of the ISDN channels independently. Any extra SCC channel can then be used for other purposes in NMSI mode. The SIMODE register is a memory-mapped read-write register cleared by reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETZ	SYNC/ SCIT	SDIAG1	SDIAG0	SDC2	SDC1	B2RB	B2RA	B1RB	B1RA	DRB	DRA	MSC3	MSC2	MS1	MS0

SETZ — Set L1TXD to zero (valid only for the GCI interface)
0 = Normal operation
1 = L1TXD output set to a logic zero (used in GCI activation)

SYNC/SCIT — SYNC Mode/SCIT Select Support
SYNC is valid only in PCM mode.
0 = One pulse wide prior to the 8-bit data
1 = N pulses wide and envelopes the N-bit data

The SCIT (Special Circuit Interface T) interface mode is valid only in GCI mode.
0 = SCIT support disabled
1 = SCIT D-channel collision enabled. Bit 4 of channel 2 C/I used by the IMP for receiving indication on the availability of the S interface D channel.

SDIAG1–SDIAG0 — Serial Interface Diagnostic Mode
00 = Normal operation
01 = Automatic echo
The channel automatically retransmits the received data on a bit-by-bit basis. The receiver operates normally, but the transmitter can only retransmit received data. In this mode, L1GR is ignored.

10 = Internal loopback
The transmitter output (L1TXD) is internally connected to the receiver input (L1RXD). The receiver and the transmitter operate normally. Transmitted data appears on the L1TXD pin, and any external data received on L1RXD pin is ignored. In this mode, L1RQ is asserted normally, and L1GR is ignored.

11 = Loopback control
In this mode, the transmitter output is internally connected to the receiver input. All the SCC transmitter outputs (TXD1/L1TXD, TXD2, TXD3) will be high and their $\overline{\text{RTS}}$ lines will be negated. L1TXD will be three-stated in IDL and PCM modes. This mode is used to accomplish SCC loopback testing without affecting pins in either NMSI modes or multiplexed modes.

SDC2 — Serial Data Strobe Control 2
0 = SDS2 signal is asserted during the B2 channel
1 = SDS1 signal is asserted during the B2 channel

SDC1 — Serial Data Strobe Control 1
0 = SDS1 signal is asserted during the B1 channel
1 = SDS2 signal is asserted during the B1 channel

B2RB, B2RA — B2 Channel Route in IDL/GCI Mode or CH-3 Route in PCM Mode
00 = Channel not supported
01 = Route channel to SCC1
10 = Route channel to SCC2 (if MSC2 is cleared)
11 = Route channel to SCC3 (if MSC3 is cleared)

B1RB, B1RA — B1 Channel Route in IDL/GCI Mode or CH-2 Route in PCM Mode
00 = Channel not supported
01 = Route channel to SCC1
10 = Route channel to SCC2 (if MSC2 is cleared)
11 = Route channel to SCC3 (if MSC3 is cleared)

DRB, DRA — D-Channel Route in IDL/GCI Mode or CH-1 Route in PCM Mode
00 = Channel not supported
01 = Route channel to SCC1
10 = Route channel to SCC2 (if MSC2 is cleared)
11 = Route channel to SCC3 (if MSC3 is cleared)

MSC3 — SCC3 Connection

0 = SCC3 is connected to the multiplexed serial interface (PCM, IDL, or GCI) chosen in MS1–MS0. NMSI3 pins are all available for other purposes.

1 = SCC3 is not connected to a multiplexed serial interface but is connected directly to the NMSI3 pins or SCP pins or is not used. The choice of general-purpose I/O port pins versus SCC3 functions is made in the port A control register. The choice of SCP pins versus SCC3 functions is made in the SPMODE register.

MSC2 — SCC2 Connection

0 = SCC2 is connected to the multiplexed serial interface (PCM, IDL, or GCI) chosen in MS1–MS0. NMSI2 pins are all available for other purposes.

1 = SCC2 is not connected to a multiplexed serial interface but is either connected directly to the NMSI2 pins or not used. The choice of general-purpose I/O port pins versus SCC2 functions is made in the port A control register.

4

MS1–MS0 — Mode Supported

00 = NMSI Mode

When working in NMSI mode, SCC1 is connected directly to the seven NMSI1 pins (RXD1, TXD1, RCLK1, TCLK1, $\overline{CD1}$, $\overline{CTS1}$, and $\overline{RTS1}$). SCC2 functions can be routed to port A as NMSI functions or configured instead as PA6–PA0. Four of the SCC3 functions can be routed to port A or retained as PA11–PA8. The other three SCC3 functions ($\overline{CTS3}$, $\overline{RTS3}$, and $\overline{CD3}$) can be routed to replace the three SCP pins or else not used.

In NMSI mode, the MSC2 and MSC3 bits are ignored. The choice of general-purpose I/O port pins versus SCC2 and SCC3 functions is made in the port A control register. See **3.3 PARALLEL I/O PORTS** for an example and more information. The choice of SCP pins versus three SCC3 functions is made in the SPMODE register in the SCP. See **4.6 SERIAL COMMUNICATIONS PORT (SCP)** for more details.

01 = PCM Mode

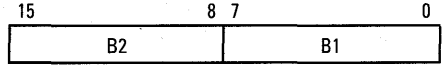
When working in PCM mode, each of the three multiplexed channels CH-1, CH-2, and CH-3 can be routed independently to each of the three SCCs. This connection is determined by the DRB, DRA, B1RB, B1RA, B2RB, and B2RA bits. SCC2 and SCC3 can be connected directly to their respective NMSI pins (if they are not needed for the PCM channels) as determined by the MSC3–MSC2 bits. In the NMSI case, the choice still exists for port/SCP functions versus SCC func-

tions as described in case 00. The MSC3–MSC2 bits override the PCM routing for a specific SCC.

10 = IDL Mode
When working in IDL/GCI mode, each ISDN channel (D, B1, and B2) can be routed independently to each of the three SCCs. This connection is determined by the DRB, DRA, B1RB, B1RA, B2RB, and B2RA bits. SCC2 and SCC3 can be connected directly to their respective NMSI pins (if they are not needed for ISDN channels) determined by the MSC3–MSC2 bits. In the NMSI case, the choice still exists for port/SCP functions versus SCC functions as described in case 00. Note that the MSC3–MSC2 bits override the ISDN connection for a specific SCC.

11 = GCI Interface
Refer to the IDL mode description.

4.4.5.2 SERIAL INTERFACE MASK REGISTER (SIMASK). The SIMASK register, a memory-mapped read-write register, is set to all ones by reset. SIMASK is used in IDL and GCI to determine which bits are active in the B1 and B2 channels. Any combination of bits may be chosen. A bit set to zero is not used by the IMP. A bit set to one signifies that the corresponding B channel bit is used for transmission and reception on the B channel.



NOTE

Bit 0 is the first bit transmitted or received on the B1 channel.

4.5 SERIAL COMMUNICATION CONTROLLERS (SCCs)

The IMP contains three independent SCCs, each of which can implement different protocols. This configuration provides the user with options for controlling up to three independent full-duplex lines implementing bridges or gateway functions or multiplexing up to three SCCs onto the same physical layer interface to implement a 2B + D ISDN basic rate channel or three channels of a PCM highway. Each protocol-type implementation uses identical buffer structures to simplify programming.

The following protocols are supported: HDLC/SDLC, BISYNC, synchronous and asynchronous DDCMP, UART, several transparent modes, and V.110 rate

adaption support. Each protocol can be implemented with IDL, GCI, PCM, or NMSI physical layer interfaces (see **4.4 SERIAL CHANNELS PHYSICAL INTERFACE**) and can be configured to operate in either echo or loopback mode. Echo mode provides a return signal from an SCC by retransmitting the received signal. Loopback mode is a local feedback connection allowing an SCC to receive the signal it is transmitting. (Echo and loopback mode for multiplexed interfaces are discussed in **4.4 SERIAL CHANNELS PHYSICAL INTERFACE**).

The receive and transmit section of each SCC is supported with one of the six dedicated SDMA channels (see **4.2 SDMA CHANNELS**). These channels transfer data between the SCCs and either external RAM or on-chip dual-port RAM. This function is transparent to the user, being enabled and controlled according to the configuration of each SCC channel. Each SCC can be clocked by either an external source (with the clock pins RCLK or TCLK) or by an internal source through a baud rate generator for each SCC channel. The baud rate generator can derive its clock from the main IMP clock or from a separate input clock. The SCC transmitter and receiver sections are independent and may be clocked at different rates.

The SCCs exhibit two types of performance limitations. The first type is a hardware clocking limit, which is the same for each SCC. The SCC clocks must not exceed a ratio of 1:2.5 serial clock (RCLK or TCLK) to parallel clock (EXTAL). Thus, for a 16.67-MHz system clock frequency, the serial clock must not exceed 6.67 MHz. The second type concerns the system data rate. The SDMA channels and CP main controller must have enough time to service the SCCs, thus preventing FIFO underruns and overruns in the SCCs. This requirement depends on a number of factors discussed in more detail in **APPENDIX A SCC PERFORMANCE**.

Each SCC supports the standard seven-line modem interface (also referred to as NMSI) with the signals RXD, TXD, RCLK, TCLK, $\overline{\text{RTS}}$, $\overline{\text{CTS}}$, and $\overline{\text{CD}}$. Other modem signals (such as DSR and DTR) may be supported through the parallel I/O pins. A block diagram of the SCC is depicted in Figure 4-5.

To provide extra modem serial output lines, the user must define I/O port A or B pins as outputs in the port A/B data direction register and write to the port A/B data register to cause the state of the pin to change. Extra serial input lines with interrupts may be supported by defining the port B pins as inputs in the port B data direction register. When a change in the state of the pin occurs, the interrupt handler may assert or negate the extra outputs to support the handshaking protocol. (See **3.3 PARALLEL I/O PORTS** for related details.)

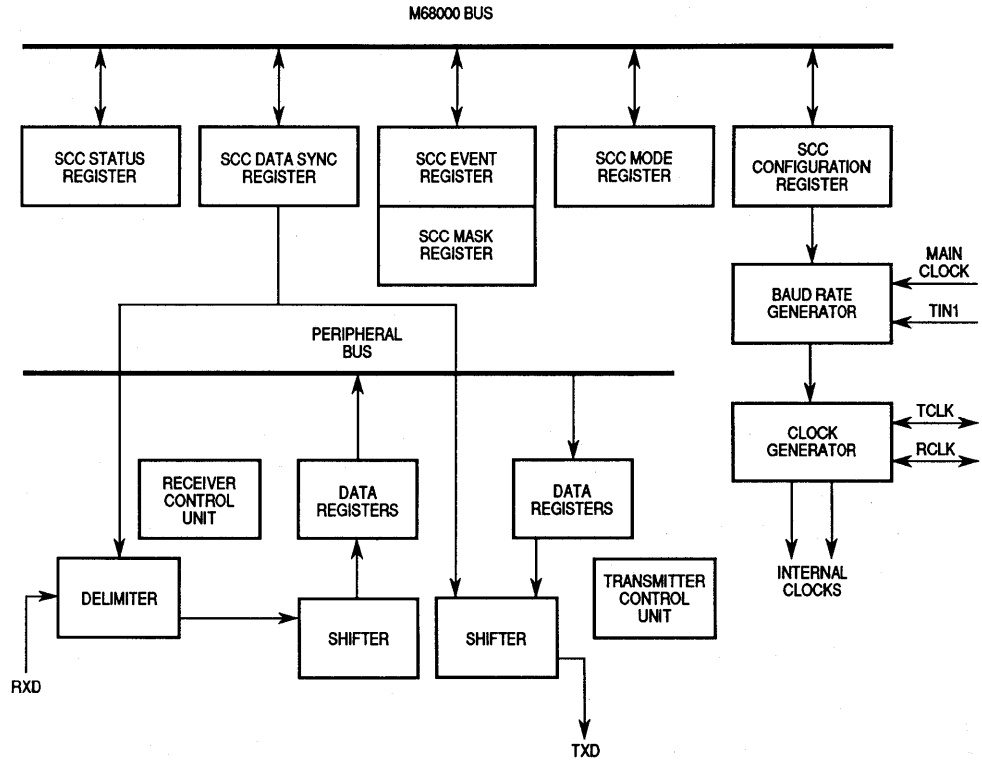


Figure 4-5. SCC Block Diagram

4.5.1 SCC Features

Each SCC channel has the following features:

- HDLC/SDLC, BISYNC, DDCMP, UART, Transparent, or V.110 Protocols
- Programmable Baud Rate Generator Driven by Main Clock or an External Clock
- Data Clocked by the Baud Rate Generator or Directly by an External Pin
- Supports Modem Signals (RXD, TXD, RCLK, TCLK, $\overline{\text{RTS}}$, $\overline{\text{CTS}}$, and $\overline{\text{CD}}$)
- Full-Duplex Operation
- Echo Mode
- Local Loopback Mode
- Baud Rate Generator Outputs Available Externally

4.5.2 SCC Configuration Register (SCON)

Each SCC controller has a configuration register that controls its operation and selects its clock source and baud rate. Figure 4-6 shows one of the three SCC baud rate generators. Each SCON is a 16-bit, memory-mapped, read-write register. The SCONs are set to $0004 by reset, resulting in the baud rate generator output clock rate being set to the main clock rate divided by 3. The baud rate generator output clock is always available externally, as shown in Table 5-8.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WOMS	EXTC	TCS	RCS	CD10	CD9	CD8	CD7	CD6	CD5	CD4	CD3	CD2	CD1	CD0	DIV4

WOMS — Wired-OR Mode Select

When WOMS is set, the TXD driver is programmed to function as an open-drain output and may be externally wired together with other TXD pins in an appropriate bus configuration. In this case, an external pullup resistor is required. When WOMS is cleared, the TXD pin operates normally with an active internal pullup.

4

NOTE

This bit is valid only in NMSI mode.

EXTC — External Clock Source

The EXTC bit selects whether the baud rate generator input clock source is the internal main clock (EXTC = 0) or external clock (EXTC = 1). If EXTC = 1, the external clock is taken from the TIN1 pin. Note that the single TIN1 pin can be used to supply a clock for all three baud rate generators.

TCS — Transmit Clock Source

The TCS bit selects either the baud rate generator output (TCS = 0) or the TCLK pin (TCS = 1) for the transmitter clock. If TCS = 0, then the baud rate generator output is driven onto the TCLK pin. This bit should be programmed to one if a multiplexed mode is chosen for the SCC.

After reset, TCLK defaults to an input and stays an input until a zero is written to TCS. For SCC2 and SCC3, TCLK can be derived directly from the RCLK pin as shown in Table 3-5.

RCS — Receive Clock Source

The RCS bit selects either the baud rate generator output (RCS = 0) or the RCLK pin (RCS = 1) for the receiver clock. If RCS = 0, then the baud rate generator output is driven onto the RCLK pin. This bit should be programmed to one if a multiplexed mode is chosen for the SCC.

After reset, RCLK defaults to an input and stays an input until a zero is written to RCS.

CD10–CD0 — Clock Divider

The clock divider bits and the prescaler determine the baud rate generator output clock rate. CD10–CD0 are used to preset an 11-bit counter that is decremented at the prescaler output rate. The counter is not otherwise accessible to the user. When the counter reaches zero, it is reloaded with the clock divider bits. Thus, a value of $7FF in CD10–CD0 produces the minimum clock rate (divide by 2049); a value of $000 produces the maximum clock rate (divide by 2).

Even when dividing by an odd number, the counter ensures a 50% duty cycle by asserting the terminal count once on a clock high and next on a clock low. The terminal count signals the counter expiration and toggles the clock.

DIV4 — SCC Clock Prescaler Divide by 4

The SCC clock prescaler bit selects a divide-by-1 or divide-by-4 prescaler for the clock divider input. The divide-by-4 option is useful in generating very slow baud rates.

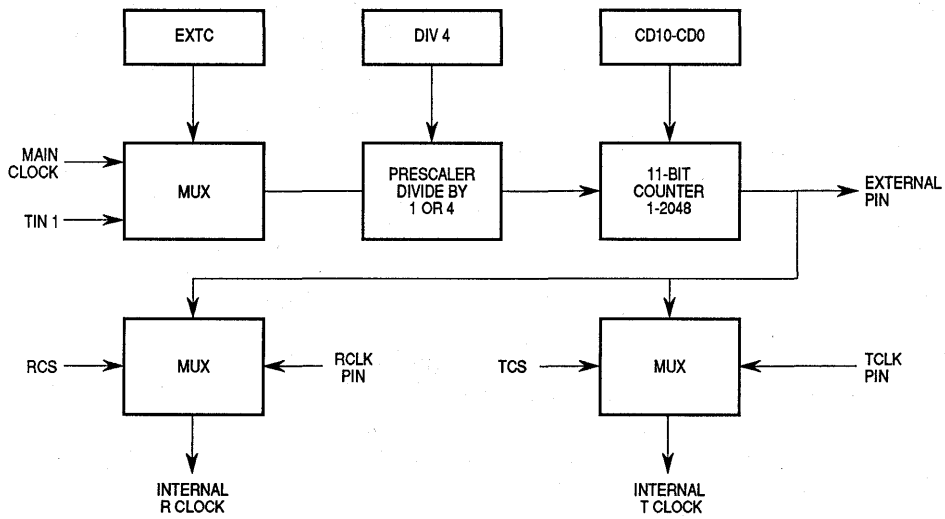


Figure 4-6. SCC Baud Rate Generator

4.5.2.1 ASYNCHRONOUS BAUD RATE GENERATOR EXAMPLES. The UART circuitry always uses a clock that is 16 × the baud rate. The ratio of the 16 × UART clock to the system parallel clock must not exceed 1:2.5. For an internally supplied clock, an integer divider value must be used; therefore, the divider must be 3 or greater. Thus, using a clock divider value of 3 (programmed as 2 in the SCON) and a 16.67-MHz crystal gives a UART clock rate of 5.56 MHz and a baud rate of 347 kbaud. Assuming again a 16.67-MHz crystal, an externally supplied UART clock on the TCLK or RCLK pins can be as high as 6.67 MHz, giving a maximum baud rate of 417 kbaud.

The baud rate using the baud rate generator is (System Clock or TIN1 clock)/(1 or 4)/(Clock Divider + 1)/16. The baud rate using the baud rate generator with an externally supplied clock to the TCLK or RCLK pins is always (TCLK or RCLK)/16.

Table 4-1 shows examples of typical bit rates of asynchronous communication and how to obtain them with the baud rate generator using an internally supplied clock.

Table 4-1. Typical Bit Rates of Asynchronous Communication

Baud Rates	EXTAL Frequency (MHz)								
	15.36			16.0			16.667		
	DIV4	DIV	Actual Frequency	DIV4	DIV	Actual Frequency	DIV4	DIV	Actual Frequency
150	1	1599	150	1	1666	149.97	1	1735	150.01
300	1	799	300	1	832	300.12	1	867	300.02
600	0	1599	600	0	1666	599.88	0	1735	600.05
1200	0	799	1200	0	832	1200.48	0	867	1200.1
2400	0	399	2400	0	416	2398.08	0	433	2400.2
4800	0	199	4800	0	207	4807.69	0	216	4800.4
9600	0	99	9600	0	103	9615.34	0	108	9556.76
19200	0	49	19200	0	51	19230.8	0	53	19290.5
38400	0	24	38400	0	25	38461.53	0	26	38581.0

4.5.2.2 SYNCHRONOUS BAUD RATE GENERATOR EXAMPLES. For synchronous communication (HDLC/SDLC, BISYNC, DDCMP, Transparent, and V.110), the internal clock is identical to the baud rate output. To obtain the desired rate, the user selects the appropriate system clock according to the following equation:

Baud rate = (System Clock or TIN1 Clock)/(Clock Divider + 1)/(1 or 4) according to the DIV4 bit

For example, to get the data rate of 64 kbps, the system clock can be 15.36 MHz, DIV4 = 0, and the Clock Divider = 239. Of course, a 64 kbps rate on the TCLK or RCLK pins could also be used.

4.5.3 SCC Mode Register (SCM)

Each SCC has a mode register. The functions of bits 5–0 are common to each protocol. The function of the specific mode bits varies according to the protocol selected by the MODE1–MODE0 bits. They are described in the relevant sections for each protocol type. Each SCM is a 16-bit, memory-mapped, read-write register. The SCMs are cleared by reset.

15 — 6	5	4	3	2	1	0
SPECIFIC MODE BITS	DIAG1	DIAG0	ENR	ENT	MODE1	MODE0

DIAG1–DIAG0 — Diagnostic Mode

00 = Normal operation ($\overline{CTS}$, $\overline{CD}$ lines under automatic control)
In this mode, the $\overline{CTS}$ and $\overline{CD}$ lines are monitored by the SCC controller. The SCC controller uses these lines to enable/disable reception and tranmission.

If $\overline{RTS}$ is programmed to be asserted by the SCC, it will be asserted once buffered data is loaded into the transmit FIFO and a falling TCLK edge occurs. The following table shows the transmit data delays.

Transmit Data Delay (TCLK Periods)

Protocol Type	From $\overline{RTS}$ Low	From $\overline{CTS}$ Low
Asynchronous Protocols	0	4
Synchronous Protocols	1	5

NOTES:
1. $\overline{RTS}$ low values assume $\overline{CTS}$ is already asserted when $\overline{RTS}$ is asserted.
2. $\overline{CTS}$ low values assume $\overline{CTS}$ met the asynchronous setup time; otherwise, an additional clock may be added.

$\overline{RTS}$ is deasserted by the SCC one clock after the last bit in a frame. The SCC latches its first bit of valid receive data on the same clock edge (rising RCLK) that samples $\overline{CD}$ as low.

01 = Loopback mode
In this mode, the transmitter output is internally connected to the receiver input while the receiver and the transmitter operate nor-

mally. The TXD pin can be programmed either to show the transmitted data or to just remain high. In this mode, the $\overline{RTS}$ line can be programmed to either be asserted normally or to remain inactive, as selected by SDIAG1–SDIAG0 in the SIMODE register. The $\overline{CTS}$ and $\overline{CD}$ lines are ignored.

10 = Automatic echo
In this mode, the channel automatically retransmits the received data on a bit-by-bit basis. The receiver operates normally, but the transmitter simply retransmits the received data. The $\overline{CTS}$ line is ignored.

11 = Software operation ($\overline{CTS}$, $\overline{CD}$ lines under software control)
In this mode, the $\overline{CTS}$ and $\overline{CD}$ lines are just inputs to the SCC event (SCCE) and status (SCCS) registers. The SCC controller does not use these lines to enable/disable reception and transmission, but leaves this task to the user.

ENR — Enable Receiver
When ENR is set, the receiver is enabled. When it is cleared, the receiver is disabled, and data is not transferred into the receive data register. If ENR is cleared during data reception, the receiver aborts the current character. To restart reception, the ENTER HUNT MODE command should be issued before ENR is set again.

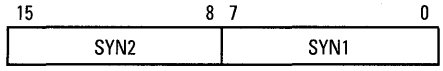
ENT — Enable Transmitter
When ENT is set, the transmitter is enabled; when ENT is cleared, the transmitter is disabled. If ENT is cleared, the transmitter will abort any data transmission, clear the transmit data FIFO and shift register, and force the TXD line high (idle). Data already in the transmit data register will not be transmitted.

The STOP TRANSMIT command additionally aborts the current frame and would normally be given to the channel before clearing ENT. The command does not clear ENT automatically. In a similar manner, to restart transmission, the user should issue the RESTART TRANSMIT command and then set ENT. The command register is described in **4.3 COMMAND SET**. The specific actions taken with each command vary somewhat according to protocol and are discussed in each protocol section.

MODE1–MODE0 — Channel Mode
00 = HDLC
01 = Asynchronous (UART and DDCMP)
10 = Synchronous DDCMP and V.110
11 = BISYNC and Promiscuous Transparent

4.5.4 SCC Data Synchronization Register (DSR)

Each DSR is a 16-bit, memory-mapped, read-write register. DSR specifies the pattern used in the frame synchronization procedure of the SCC in the synchronous protocols. In the UART protocol it is used to configure fractional stop bit transmission. After reset, the DSR defaults to $7E7E (two FLAGs); thus, no additional programming is necessary for the HDLC protocol. For BISYNC, DDCMP, and V.110, the contents of the DSR should be written before the channel is enabled.



4.5.5 Buffer Descriptors Table

Data associated with each SCC channel is stored in buffers. Each buffer is referenced by a buffer descriptor (BD). BDs are located in each channel's BD table (located in dual-port RAM). There are two such tables for each SCC channel: one is used for data received from the serial line; the other is used to transmit data. The actual buffers may reside in either external memory or internal memory (dual-port RAM). For internal memory data buffers, the data buffer pointer is in the low-order data pointer word and is an offset from the device base address to any available area in the dual-port RAM. (Data buffers may reside in the parameter RAM of an SCC if it is not enabled).

The BD table allows the user to define up to eight buffers for the transmit channel and up to eight buffers for the receive channel (see Figure 4-7). Each BD table forms a circular queue.

The format of the BDs is the same for each SCC mode of operation (HDLC, UART, DDCMP, BISYNC, V.110) and for both transmit or receive. Only the first field (containing status and control bits) differs for each protocol. The BD format is shown in Figure 4-8.

For frame-oriented protocols (HDLC, BISYNC, DDCMP, V.110), a message may reside in as many buffers as are necessary (transmit or receive). Each buffer has a maximum length of 64K – 1 bytes. The CP does not assume that all buffers of a single frame are currently linked to the BD table, but does assume that the unlinked buffers will be provided by the processor in time to be either transmitted or received. Failure to do so will result in an error being reported by the CP.

For example, assume the first six buffers of the transmit BD table have been transmitted and await processing by the M68000 core (with all eight buffers

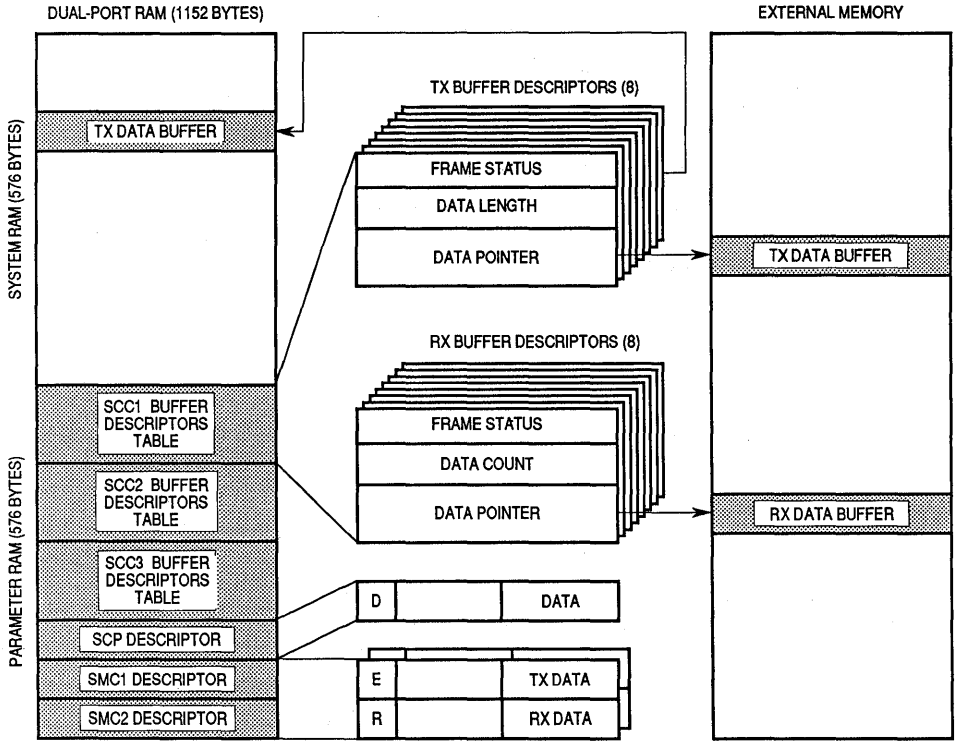


Figure 4-7. Memory Structure

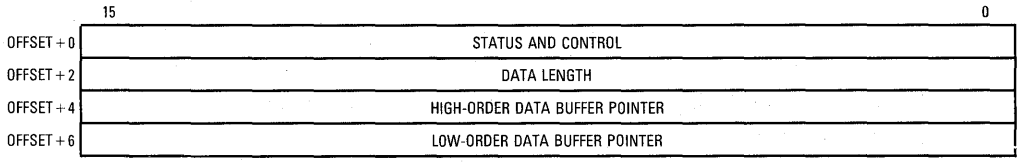


Figure 4-8. SCC Buffer Descriptor Format

used in the circular queue), and a three-buffer frame awaits transmission. The first two buffers may be linked to the remaining two entries in the table as long as the user links the final buffer into the first entry in the BD table before the IMP attempts its transmission. If the final buffer is not linked in time to the BD table by the time the CP attempts its transmission, the CP will report an underrun error.

Buffers allocated to an SCC channel may be located in either internal or external memory. Memory allocation occurs for each BD individually. If internal memory is selected, the CP uses only the lower 11 address bits (A10–0) as an offset to the internal dual-port RAM. Accesses to the internal memory by the CP are one clock cycle long and occur without arbitration. If external memory is selected, the pointers to the data buffers are used by the CP as 24 bits of address.

If function codes are used to decode the external address (e.g., in the on-chip chip select logic), the memory space (function code) of this data buffer must be set before external buffers can be accessed; it can then be changed only when the user is sure that the CP is not currently accessing external buffers for that channel. There are six separate function code registers located in the parameter RAM for the three SCC channels: three for receive data buffers (RFCR) and three for transmit data buffers (TFCR).

The CP processes the transmit BDs in a straightforward fashion. Once the transmit side of an SCC is enabled, it starts with the first BD in that SCC's transmit BD table, periodically checking a bit to see if that BD is "ready". Once it is ready, it will process that BD, reading a word at a time from its associated buffer, doing certain required protocol processing on the data, and moving resultant data to the SCC transmit FIFO. When the first buffer has been processed, the CP moves on to the next BD, again waiting for that BD's "ready" bit to be set. Thus, the CP does no look-ahead BD processing, nor does it skip over BDs that are not ready. When the CP sees the "wrap" bit set in a BD, it goes back to the beginning of the BD table, after processing of this BD is complete. After using a BD, the CP sets the "ready" bit to not-ready; thus, the CP will never use a BD twice until the BD has been confirmed by the M68000 core.

The CP uses the receive BDs in a similar fashion. Once the receive side of an SCC is enabled, it starts with the first BD in that SCC's receive BD table. Once data arrives from the serial line into the SCC, the CP performs certain required protocol processing on the data and moves the resultant data (either bytes or words at a time depending on the protocol) to the buffer pointed to by the first BD. Use of a BD is complete when there is no more room left in the buffer or when certain events occur, such as detection of an error or an end-of-frame. Whatever the reason, the buffer is then said to be "closed," and additional data will be stored using the next BD. Whenever the CP needs to begin using a BD because new data is arriving, it will check the "empty" bit of that BD. If the current BD is not empty, it will report a "busy" error. However, it will not move from the current BD until it becomes empty. When the CP sees the "wrap" bit set in a BD, it goes back to the beginning of the

BD table, after use of this BD is complete. After using a BD, the CP sets the "empty" bit to not-empty; thus, the CP will never use a BD twice until the BD has been "processed" by the M68000 core.

Each SCC has eight transmit BDs and eight receive BDs except for the transmit BD table of SCC3 which has four BDs. However, it is actually possible to regain additional Tx BDs for SCC3 as follows. The Tx BD table may be extended by two BDs to six BDs if the SMCs are not used. Additionally, all eight Tx BDs for SCC3 may be used if the following is considered: 1) the SCP and SMCs must not be used; 2) various words within the last two BDs will be changed by the CP during its initialization routine following any reset; and 3) the BERR channel number value will be written into the last BD after any SDMA bus error (see **4.5.8.4 BUS ERROR ON SDMA ACCESS**), but this is not a major concern since the CP must be reset after any SDMA bus error.

4.5.6 SCC Parameter RAM Memory Map

Each SCC maintains a section in the dual-port RAM called the parameter RAM. Each SCC parameter RAM area begins at offset $80 from each SCC base area ($400, $500, or $600) and continues through offset $BF. Refer to Table 2-8 for the placement of the three SCC parameter RAM areas. Part of each SCC parameter RAM (offset $80–$9A), which is identical for each protocol chosen, is shown in Table 4-2. Offsets $9C–$BF comprise the protocol-specific portion of the SCC parameter RAM and are discussed relative to the particular protocol chosen.

Table 4-2. SCC Parameter RAM Memory Map

Address		Name	Width	Description
SCC Base+80	*	RFCR	Byte	Rx Function Code
SCC Base+81	*	TFCR	Byte	Tx Function Code
SCC Base+82	*	MRBLR	Word	Maximum Rx Buffer Length
SCC Base+84	†		Word	Rx Internal State
SCC Base+86	†		Byte	Reserved
SCC Base+87	†	RBD#	Byte	Rx Internal Buffer Number
SCC Base+88			2 Words	Rx Internal Data Pointer
SCC Base+8C			Word	Rx Internal Byte Count
SCC Base+8E			Word	Rx Temp
SCC Base+90	†		Word	Tx Internal State
SCC Base+92	†		Byte	Reserved
SCC Base+93	†	TBD#	Byte	Tx Internal Buffer Number
SCC Base+94			2 Words	Tx Internal Data Pointer
SCC Base+98			Word	Tx Internal Byte Count
SCC Base+9A			Word	Tx Temp
SCC Base+9C SCC Base+BF				First Word of Protocol-Specific Area Last Word of Protocol-Specific Area

*Initialized by the user (M68000 core).
†Modified by the CP following a CP or system reset.

Certain parameter RAM values need to be initialized by the user before the SCC is enabled. Those values not so designated are initialized/written by the CP. Once initialized, most parameter RAM values will not need to be accessed in user software since most of the activity is centered around the transmit and receive buffer descriptors, not the parameter RAM. However, if the parameter RAM is accessed by the user, the following should be noted. The parameter RAM can be read at any time. The parameter RAM values related to the SCC transmitter can only be written 1) whenever the ENT bit in the SCM is zero or 2) after a STOP TRANSMIT command and before a RESTART TRANSMIT command. The parameter RAM values related to the SCC receiver can only be written 1) whenever the ENR bit in the SCM is zero or 2) if the receiver has previously been enabled, after the ENTER HUNT MODE command and before the ENR bit is set. See **4.5.10 Power Saving (Disabling the SCCs)** for a discussion of when the SCC registers may be changed.

The registers (see Table 4-2) that typically need to be accessed by the user are described in the following paragraphs.

4.5.6.1 DATA BUFFER FUNCTION CODE REGISTER (TFCR, RFCR). This register defines the address space of the receive (RFCR) and transmit (TFCR) data buffers.

NOTE

The value of the function code register for any channel may be equal to that of any other.

7	6	5	4	3	2	1	0
0	FC2	FC1	FC0	0	0	0	0

4.5.6.2 MAXIMUM RECEIVE BUFFER LENGTH REGISTER (MRBLR). The MRBLR for each SCC channel defines the maximum number of bytes that the IMP will write to a receive buffer before moving to the next buffer. Thus, each receive buffer should be at least MRBLR bytes in length. (Transmit buffers have no such requirement.) MRBLR must be set by the user before data is received on that channel. Since only one maximum buffer size is given per channel (and not for each buffer), this value must not exceed the size of the smallest buffer linked to this channel's BD table.

NOTE

The MRBLR value should be greater than zero in all modes. In the HDLC and transparent modes, the MRBLR should have an even value.

4.5.6.3 RECEIVER BUFFER DESCRIPTOR NUMBER (RBD#). The RBD# for each SCC channel defines the next BD to which the receiver will move data when it is in the IDLE state or defines the current BD during frame processing. The RDB# is the BD offset from the SCC base in the Rx BD table. For Rx BD 0, RBD# = $00; for Rx BD 1, RBD# = $08, etc. Upon reset, the CP main controller sets this register to zero. The user can change this register only after the ENTER HUNT MODE command has been issued. In most applications, this parameter will not need to be modified by the user.

4.5.6.4 TRANSMIT BUFFER DESCRIPTOR NUMBER (TBD#). The TBD# for each SCC channel defines the next BD from which the transmitter will move data when it is in the IDLE state or defines the current BD during frame transmission. The TDB# is the BD offset from the SCC base in the Tx BD table. For Tx BD 0, TBD# = $40; for Tx BD 1, TBD# = $48, etc. Upon reset, the CP main controller sets this register to $40. The user can change this register only after the STOP TRANSMIT command has been issued. In most applications, this parameter will not need to be modified by the user.

4.5.6.5 OTHER GENERAL PARAMETERS. Additional parameters are listed in Table 4-2. These parameters do not need to be accessed by the user in normal operation, and are listed only because they may provide helpful information for experienced users. The Rx and Tx internal data pointers are updated by the SDMA channels to show the next address in the buffer to be accessed. The Tx byte count is a down count value that is initialized with the Tx BD data length and decremented with every byte read by the SDMA channels. The Rx byte count is a down count value that is initialized with the MRBLR value and decremented with every byte written by the SDMA channels.

4.5.7 SCC Initialization

The SCCs require a number of registers and parameters to be configured after a power-on reset. The following is a proper sequence for initializing the SCCs regardless of the protocol used.

1. If SCC2 or SCC3 is used, write the parallel port A and B control registers (PACNT and PBCNT) to configure pins as parallel I/O lines or peripheral functions as needed (see **3.3 PARALLEL I/O PORTS**).
2. Write SIMODE to configure the serial channels physical interface for the three SCCs (i.e., NMSI, PCM, GCI, IDL modes). If IDL or GCI is chosen in SIMODE, write SIMASK in the serial channels physical interface (see **4.4.5 Serial Interface Registers**).
3. Write SCON (see **4.5.2 SCC Configuration Register (SCON)**).
4. Write SCM (SCC Mode) but do not set the ENT or ENR bits yet (see **4.5.3 SCC Mode Register (SCM)**).
5. Write DSR if a protocol other than HDLC is used (see specific protocol section).
6. Initialize the required values in the general-purpose parameter RAM (see **4.5.6 SCC Parameter RAM Memory**).
7. Initialize the required values in the protocol-specific parameter RAM (see specific protocol section).
8. Clear out any current events in SCCE, if desired (see specific protocol section).
9. Write SCCM to enable the interrupts in SCCE that should reach the interrupt controller (see specific protocol section).
10. Write IMR in the interrupt controller to enable the SCC interrupts to the interrupt controller (see **3.2.5.3 INTERRUPT MASK REGISTER (IMR)**).
11. Set the ENR and/or ENT bits in SCM (see **4.5.3 SCC Mode Register**).

The buffer descriptors may be initialized at any time. Notice that the command register does not need to be accessed following power-on reset. The SCCs should be reset with a hardware reset or by setting the RST bit in the command register after any dynamic change in the parallel I/O ports or serial channels physical interface configuration.

4.5.8 Interrupt Mechanism

Interrupt handling for each of the SCC channels is configured on a global per-channel basis in the interrupt pending register (IPR), the interrupt mask register (IMR), and the interrupt in-service register (ISR). Within each of these registers, one bit is used to either mask or report the presence of a pending or in-service interrupt in an SCC channel. However, an SCC interrupt may be

caused by a number of events. To allow interrupt handling for SCC-specific events, further registers are provided within the SCCs.

Up to eight events can cause the SCC to interrupt the processor. The events differ in accordance with the SCC protocol chosen. The events are handled independently for each channel by the SCC event register and the SCC mask register. All unmasked event bits must be cleared in order for the corresponding IPR bit to be cleared. The interrupt handler typically reads the event register, and then clears only those bits that it will deal with during the interrupt.

4.5.8.1 SCC EVENT REGISTER (SCCE). This 8-bit register is used to report events recognized by any of the SCCs. On recognition of an event, the SCC will set its corresponding bit in the SCC event register (regardless of the corresponding mask bit in the SCC mask register). The SCC event register is a memory-mapped register that may be read at any time. A bit is reset by writing a one (writing a zero does not affect a bit's value). More than one bit may be reset at a time. This register is cleared by reset.

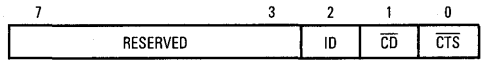
4.5.8.2 SCC MASK REGISTER (SCCM). This 8-bit read-write register allows enabling or disabling interrupt generation by the CP for specific events in each SCC channel. An interrupt will only be generated if the SCC interrupts for this channel are enabled in the IMR in the interrupt controller.

If a bit in the SCC mask register is zero, the CP will not proceed with its usual interrupt handling whenever that event occurs. Any time a bit in the SCC mask register is set, a one in the corresponding bit in the SCC event register will cause the SCC-event bit in the IPR to be set.

The bit locations in the SCC mask register are identical to those in the SCC event register. SCCM is cleared upon reset.

4.5.8.3 SCC STATUS REGISTER (SCCS). Each SCC status register reflects the current status of the line for each of the three SCCs on an independent basis. This 8-bit read-only register may be read at any time.

In the BDs for each protocol, indications are given as to whether the status of these signals has changed during the reception of a given buffer. Furthermore, in the event registers for each protocol, a maskable interrupt bit is provided to allow convenient detection of a change in signal status.



Bits 7–3 — Reserved for future use.

ID — Idle Status on the Receiver Line
This status is valid only in the HDLC and UART modes, and will be one while the $\overline{CD}$ signal is negated.

$\overline{CD}$ — Carrier Detect Status on the Channel Pin
This bit has the same polarity as the external pin.

$\overline{CTS}$ — Clear-to-Send Status on Channel Pin
This bit has the same polarity as the external pin.

When the $\overline{CTS}$ and $\overline{CD}$ lines are programmed to software control in the SCC mode register, these lines do not affect the SCC and can be used for other purposes such as a data set ready (DSR), a data terminal ready (DTR) line, or an interrupt source in the SCCE register.

4.5.8.4 BUS ERROR ON SDMA ACCESS. When a bus error occurs on an access by the SDMA channel, the CP generates a unique interrupt (see **3.2 INTERRUPT CONTROLLER**). The interrupt service routine should read the bus error channel number from the parameter RAM at BASE + 67C as follows:

0 — SCC1 Tx Channel
1 — SCC1 Rx Channel
2 — SCC2 Tx Channel
3 — SCC2 Rx Channel
4 — SCC3 Tx Channel
5 — SCC3 Rx Channel

Next, the pointer that caused the bus error can be determined by reading the Rx or Tx internal data pointer from the parameter's memory map of the particular SCC. Following this bus error, the CP must be reset with a hardware reset or the setting of the RST bit in the command register.

4.5.9 SCC Transparent Mode Support

The SCC supports several transparent receive and transmit modes, which vary according to the protocol being implemented. As used here, transparent means data transparency and is not the same as the promiscuous mode

described later in this section. The transparent modes for each protocol are as follows:

UART Mode

If the normal mode is selected (UM1–UM0 = 00 in the UART mode register) and the control characters table is empty, then the UART will transfer all received characters to memory.

HDLC Mode

When the HDLC mask register = $0000, the HDLC controller will transfer all received frames to the receive buffers, including address, control, data, and CRC. The HDLC controller still performs the standard HDLC functions of zero insertion/deletion and flag recognition. For complete bit transparency, use the promiscuous mode.

BISYNC Mode

The BISYNC mode may be used to transparently receive data that is delimited by SYNCs. To do this, the BISYNC control characters table should be empty, and SYNC/DLE stripping should be disabled. Thereafter, all data following the SYNCs will be received into the receive buffers.

There are two possible ways of recognizing SYNCs. Either the EXSYN bit in the BISYNC mode register is cleared and a SYNC is defined by the SYN1–SYN2 characters in DSR, or the EXSYN bit is set and a SYNC is determined by an external SYNC pulse.

When the BISYNC control characters table is empty, the SYNC and DLE stripping is disabled, and the BISYNC controller is in normal nontransparent mode (RTR = 0 in BISYNC mode register). In this case, the configuration is as follows:

If EXSYN in the BISYNC mode register is cleared, then the BISYNC controller transfers all characters that follow the SYN1–SYN2 sequence to the receive buffers.

If EXSYN in the BISYNC mode register is set, then the BISYNC controller transfers all characters that follow the external SYNC pulse to the receive buffers. This configuration is also used for promiscuous mode.

NOTE

The BISYNC controller can reverse the bit order in both modes.

Promiscuous (Totally Transparent) Mode

The MC68302 can both receive and transmit the entire serial bit stream transparently. This mode is configured by selecting BISYNC mode and setting the NTSYN and EXSYN bits set in the BISYNC mode register. No other BISYNC mode register bits are valid in transparent mode except REVD, and the TB, B, BR, TD, and TR bits should be cleared in the Tx BD. They are not used.

Any physical serial interface mode can be selected. If NMSI mode is used, the $\overline{CD}$ pin becomes the sync pin (see the EXSYN bit in the BISYNC mode register). The receiver will clock in data whenever the appropriate sync pin is asserted. This pin may be toggled by external logic as needed. There is no CD lost function in transparent mode when the EXSYN bit is set.

In totally transparent mode, the transmitter sends ones and negates $\overline{RTS}$ between buffers. If the Last (L) bit is not set in a Tx BD and the next BD is not ready, the transmitter will signify an underrun, and the TXE bit in the BISYNC event register will be set.

All transfers in totally transparent mode are 16-bits. The receive buffer pointer and buffer size (MRBLR) must be even, but the transmit byte count need not be even.

All of the above modes assume NMSI mode for the physical layer interface. In the multiplexed modes (GCI, IDL, and PCM), the external SYNC pin is used to achieve the proper bit synchronization on the transparent channel. Note that the SYNC signal to the BISYNC controller is not connected externally in the multiplexed modes; the BISYNC controller assumes the signal is always asserted.

4.5.10 Power Saving (Disabling the SCCs)

To save power, the SCCs may simply be disabled. Clearing the enable transmitter (ENT) bit in the SCC mode register causes the SCC transmitter to consume the least possible power; clearing the ENR bit causes a similar action for the SCC receiver.

If an SCC transmitter or receiver is not needed for a period of time, it may be disabled and re-enabled later. In this case, a sequence of operations is followed.

For the SCC transmitter, the sequence is as follows:

STOP TRANSMIT Command
Clear ENT
•
• (The SCC transmitter is now disabled)
•
RESTART TRANSMIT Command
Set ENT

For the SCC receiver, the sequence is as follows:

Clear ENR
•
• (The SCC receiver is now disabled)
•
ENTER HUNT MODE Command
Set ENR

This sequence assures that any buffers in use will be properly closed and that new data will be transferred to/from a new buffer.

While an SCC is disabled (and only while an SCC is disabled) the SCON and SCM registers may be modified. Thus, once disabled, changes such as the SCC protocol, diagnostic mode, or baud rate may be made. Such parameters cannot be modified "on-the-fly." The DSR should also only be modified while an SCC is disabled, although an exception exists to this in the UART mode concerning the transmission of partial stop bits.

The TBD# and RBD# values in the parameter RAM are not reset by the disabling process; thus, the very next BDs will be used when the SCC is re-enabled. A full software reset of the entire CP including the three SCCs is accomplished in the command register (CR). This full reset is required if any change is made to the SCC parallel I/O or serial channels physical interface configuration. The SCC does not need to be disabled or reset if only a change to a parameter RAM value is made. See **4.5.6 SCC Parameter RAM Memory Map** for a discussion of when parameter RAM values may be modified.

The above discussion on saving power is independent of a decision to use the low-power modes (see **3.8 SYSTEM CONTROL**). If a low-power mode is desired, the SCC may be disabled before entering the low-power mode, or it may be left enabled so that an SCC interrupt may bring the IMP out of the low-power mode.

One common use of the low-power mode is to disable the transmitter but leave the receiver enabled (i.e., in the hunt mode) so that an arriving frame destined for this station will cause an interrupt, waking the IMP from its low-power mode.

The low-power mode affects the M68000 core, not the SCCs. Since the SCCs are usually clocked at a far lower rate than the M68000 core, significant power savings may still be achieved with the SCCs fully enabled and the M68000 core in the low-power mode.

4.5.11 UART Controller

By appropriately setting the SCC mode register, any of the SCC channels may be configured to function as a universal asynchronous receiver transmitter (UART). The UART provides standard serial I/O using asynchronous character-oriented (start-stop) protocols. The UART may be used to communicate with other existing UART devices. Also, in conjunction with another SCC channel, it may be used in either ISDN terminal adaptor or X.25 packet assembly and disassembly (PAD) applications.

The UART provides a port for serial communication to other microprocessors, terminals, etc., either locally or through modems. It includes facilities for communication using standard asynchronous bit rates and protocols. The UART supports a multidrop mode for master/slave operation with wakeup capability on either an idle line or an address bit.

The UART uses a seven-pin interface in NMSI mode (although the UART could also be multiplexed into the IDL/GCI/PCM layer-1 interface). It transmits data from memory (internal or external) to the TXD line and receives data from the RXD line into memory. The seven dedicated serial interface pins are transmit data (TXD), receive data (RXD), receive clock (RCLK), transmit clock (TCLK), carrier detect ($\overline{CD}$), clear to send ($\overline{CTS}$), and request to send ($\overline{RTS}$). Other modem lines such as data set ready (DSR) and data terminal ready (DTR) can be supported through the parallel I/O pins.

The UART consists of separate transmit and receive sections whose operations are asynchronous with the M68000 core and may be either synchronous or asynchronous with respect to each other. Each clock can be supplied either from the baud rate generator or from the external pins.

The UART key features are as follows:

- Flexible Message-Oriented Data Buffers
- Multidrop Operation
- Receiver Wakeup on IDLE Line or Address Mode
- Eight Control Character Comparison Registers
- Two Address Comparison Registers
- Four 16-Bit Error Counters
- Programmable Data Length (7 or 8 Bits)
- Programmable 1 or 2 Stop Bits with Fractional Stop Bits
- Even/Odd/Force/No Parity Generation
- Even/Odd/No Parity Check
- Frame Error, Noise Error, Break, and IDLE Detection
- Transmits Preamble and Break Sequences
- Freeze Transmission Option
- Maintenance of Four 16-Bit Error Counters
- Provides Asynchronous Link for DDCMP Use
- Flow Control Character Transmission Supported

4.5.11.1 NORMAL ASYNCHRONOUS MODE. In the normal asynchronous mode, the receive shift register receives the incoming data on the RXD pin. The length and the format of the serial word in bits are defined by the control bits in the UART mode register. The order of reception is as follows:

Start Bit
7 or 8 Data Bits with the Least Significant Bit First
Address/Data Bit (Optional)
Parity Bit (Optional)
Stop Bits

The receiver samples each bit of the incoming data three times around its center. The value of the bit is determined by the majority of those samples. If all the samples do not agree, a noise indication counter is incremented. When a complete byte has been clocked in, the contents of the shift register are transferred to the UART receive data register. If there is an error in this character, then the appropriate error bits will be set by the IMP.

The UART may receive fractional stop bits. The next character's start bit may begin anytime after the 11th internal clock of the previous character's first stop bit (the UART uses a 16× clock).

The UART transmit shift register transmits the outgoing data on the TXD pin as shown in Figure 4-9. Data is clocked synchronously with the transmit clock, which may have either an internal or external source. The order of bit transmission is as stated for reception.

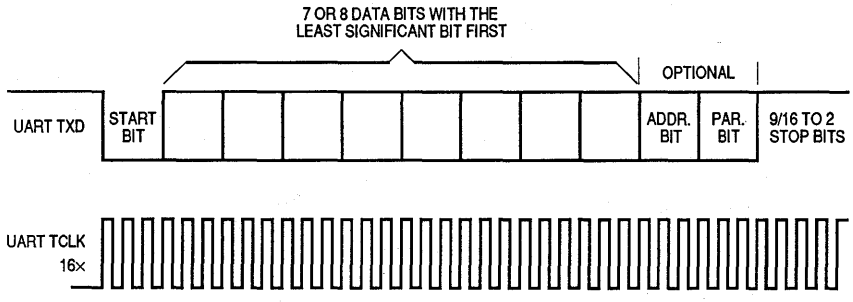


Figure 4-9. UART Frame Format

Only the data portion of the UART frame is actually stored in the data buffers. The start and stop bits are always generated and stripped by the UART controller. The parity bit may also be generated in the case of transmission, and checked during reception. Although parity is not stored in the data buffer, its value may be inferred by the reporting mechanism in the data buffer (i.e., characters with parity errors are identified). Similarly, the optional address bit is not stored in the transmit or receive data buffer, but is implied from the buffer descriptor itself.

4.5.11.2 ASYNCHRONOUS DDCMP MODE. The IMP also allows the DDCMP protocol to be run over an asynchronous connection, using the UART. The description of this operation is contained in **4.5.14 DDCMP Controller**. This operation uses the DDCMP buffer structures and the DDCMP-specific parameter RAM; however, the SCC mode register must be configured as a UART. The proper programming of the UART mode register to obtain asynchronous DDCMP is covered in **4.5.11.4 UART PROGRAMMING MODEL.**

4.5.11.3 UART MEMORY MAP. When configured to operate in UART mode, the IMP overlays the structure (see Table 4-3) onto the protocol-specific area of that SCC's parameter RAM. Refer to **2.8 MC68302 MEMORY MAP** for the

placement of the three SCC parameter RAM areas and to Table 4-2 for the other parameter RAM values.

Table 4-3. UART-Specific Parameter RAM

Address	Name	Width	Description
SCC Base+9C *	MAX_IDL	Word	Maximum IDLE Characters
SCC Base+9E	IDLC	Word	Temporary Receive IDLE Counter
SCC Base+A0 *	BRKCR	Word	Break Count Register
SCC Base+A2 *	PAREC	Word	Receive Parity Error Counter
SCC Base+A4 *	FRMEC	Word	Receive Framing Error Counter
SCC Base+A6 *	NOSEC	Word	Receive Noise Counter
SCC Base+A8 *	BRKEC	Word	Receive Break Character Counter
SCC Base+AA *	UADDR1	Word	UART ADDRESS Character 1
SCC Base+AC *	UADDR2	Word	UART ADDRESS Character 2
SCC Base+AE	RCCR	Word	Receive Control Character Register
SCC Base+B0 *	CHARACTER1	Word	CONTROL Character 1
SCC Base+B2 *	CHARACTER2	Word	CONTROL Character 2
SCC Base+B4 *	CHARACTER3	Word	CONTROL Character 3
SCC Base+B6 *	CHARACTER4	Word	CONTROL Character 4
SCC Base+B8 *	CHARACTER5	Word	CONTROL Character 5
SCC Base+BA *	CHARACTER6	Word	CONTROL Character 6
SCC Base+BC *	CHARACTER7	Word	CONTROL Character 7
SCC Base+BE *	CHARACTER8	Word	CONTROL Character 8

*Initialized by the user (M68000 core).

4.5.11.4 UART PROGRAMMING MODEL. An SCC configured as a UART uses the same data structure as the HDLC, BISYNC, and DDCMP modes. The UART data structure supports multibuffer operation. The UART may also be programmed to perform address comparison whereby messages not destined for a given programmable address are discarded. Also, the user can program the UART to accept or reject control characters. If a control character is rejected, an interrupt may be generated. The UART enables the user to transmit break and preamble sequences. Overrun, parity, noise, and framing errors are reported using the buffer descriptor (BD) table and/or error counters. An indication of the status of the line (idle) is reported through the status register, and a maskable interrupt is generated upon a status change.

In its simplest form, the UART can function in a character-oriented environment. Each character is transmitted with accompanying stop bits and parity (as configured by the user) and is received into separate one-byte buffers. Reception of each buffer may generate a maskable interrupt.

Many applications may want to take advantage of the message-oriented capabilities supported by the UART using linked buffers to receive or transmit

data. In this case, data is handled in a message-oriented environment; users can work on entire messages rather than operating on a character-by-character basis. A message may span several linked buffers. For example, rather than being interrupted after the reception of each character, a terminal driver may want to wait until an end-of-line character has been typed by a user before handling the input data.

As another example, when transmitting ASCII files, the data may be transferred as messages ending on the end-of-line character. Each message could be both transmitted and received as a linked list of buffers without any intervention from the M68000 core. This technique achieves both ease in programming and significant savings in processor overhead.

On the receive side, the user may define up to eight control characters. Each control character may be configured to designate the end of a message (such as end of line) or to generate a maskable interrupt without being stored in the data buffer. This latter option is useful when flow-control characters such as XON or XOFF need to alert the M68000 core, yet do not belong to the message being received. Flow-control characters may also be transmitted at any time.

In the message-oriented environment, the data stream is divided into buffers. However, the physical format of each character (stop bits, parity, etc.) is not altered.

4.5.11.5 UART COMMAND SET. These commands are issued to the command register described in **4.3 COMMAND SET.**

STOP TRANSMIT Command

After a hardware or software reset and the enabling of the channel by writing the SCC mode register, the channel is in the transmit enable mode and starts polling the first BD in the table approximately every eight transmit clocks.

The channel STOP TRANSMIT command disables the transmission of characters on the transmit channel. If this command is received by the UART controller during message transmission, transmission of that message is aborted. The UART completes transmission of any data already transferred to the UART FIFO (up to three characters) and then stops transmitting data. The TBD# is then made to point to the next Tx BD in the table.

The UART transmitter will transmit a programmable number of break sequences and then start to transmit idles. The number of break sequences (which may be zero) should be written to the break count register (BRKCR) before this command is given to the UART controller.

The STOP TRANSMIT command must be issued before the SCC mode register is used to disable the transmitter if the transmitter will be re-enabled at a later time.

RESTART TRANSMIT Command

The channel RESTART TRANSMIT command re-enables the transmission of characters on the transmit channel. This command is expected by the UART in three situations: after issuing a STOP TRANSMIT command, after issuing a STOP TRANSMIT and then disabling the channel using the SCC mode register, or after transmitter errors (CTS lost). The UART controller will resume transmission from the current transmitter BD number (TBD#) in the channel's Tx BD table.

If the transmitter is being re-enabled, the RESTART TRANSMIT command must be used and should be followed by the enabling of the transmitter in the SCC mode register.

ENTER HUNT MODE Command

After a hardware or software reset and the enabling of the channel by its SCC mode register, the channel is in the receive enable mode and will use the first BD in the table.

The ENTER HUNT MODE command is used to force the UART controller to abort reception of the current message and enter the hunt mode. The UART controller will resume reception using the next BD if a message was in progress. In multidrop hunt mode, the UART controller continually scans the input data stream for the address character. While not in multidrop mode, the UART controller will wait for an IDLE sequence.

If an enabled receiver has been disabled by clearing ENR in the SCC mode register, the ENTER HUNT MODE command must be given to the channel before setting ENR again. Reception will then begin with the next BD.

4.5.11.6 UART ADDRESS RECOGNITION. The UART can be configured to operate in a multidrop environment in which two modes are supported:

Automatic Multidrop Mode — The IMP automatically checks the incoming address character and accepts the data following it only if the address matches one of two 8-bit preset values. In this mode, UM1–UM0 = 11 in the UART mode register.

Nonautomatic Multidrop Mode — The IMP receives all characters. An address character is always written to a new buffer (it may be followed by data characters). In this mode, UM1–UM0 = 01 in the UART mode register.

Each UART controller has two 8-bit address registers (UADDR1 and UADDR2) for address recognition. In the automatic mode, the incoming address is checked against the UART address registers. Upon an address match, the address match (M) bit in the BD is set/cleared to indicate which address character was matched. The data following it is written to the data buffers.

NOTE

For 7-bit characters, the eighth bit in UADDR1 and UADDR2 should be zero.

4.5.11.7 UART CONTROL CHARACTERS AND FLOW CONTROL. The UART has the capability to recognize special control characters. These characters may be used when the UART functions in a message-oriented environment. Up to eight control characters may be defined by the user in the control characters table. Each of these characters may be either stored (written to the receive buffer, after which the current buffer is closed and a new receive buffer taken) or rejected. If rejected, the character is written to the received control character register (RCCR) in internal RAM, and a maskable interrupt is generated. This method is useful for notifying the user of the arrival of control characters (e.g., XOFF) that are not part of the received messages.

The UART uses a table of 16-bit entries to support control-character recognition. Each entry consists of the control character, an end-of-table bit, and a reject character bit. The control characters table is shown in Figure 4-10.

RCCR — Received Control Character Register

Upon a control character match for which the reject bit is set, the UART will write the control character into the RCCR and generate a maskable interrupt. The M68000 core must process the interrupt and read the RCCR before a second control character arrives. Failure to do so will result in the UART overwriting the first control character.

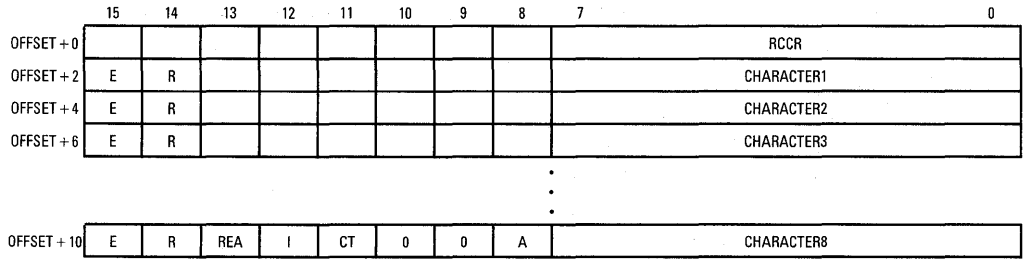


Figure 4-10. UART Control Characters Table

CHARACTER8–CHARACTER1 — Control Character Value
These fields define control characters that should be compared to the incoming character. For 7-bit characters, the eight bit (bit 7) should be zero.

E — End of Table
0 = This entry is valid. The lower eight bits will be checked against the incoming character.
1 = The entry is not valid. No valid entries lie beyond this entry.

NOTE

In tables with eight control characters, E is always zero.

R — Reject Character
0 = The character is not rejected but is written into the receive buffer. The buffer is then closed, and a new receive buffer is used if there is more data in the message. A maskable interrupt is generated.
1 = If this character is recognized, it will not be written to the receive buffer. Instead, it is written to the RCCR, and a maskable interrupt is generated. The current buffer is not closed when a control character is received with R set.

Transmission of out-of-sequence characters is also supported and is normally used for the transmission of flow control characters such as XON or XOFF. This is performed using the last (eighth) entry in the UART control characters table. The UART will poll this character whenever the transmitter is enabled for UART operation: during freeze, during buffer transmission, and when no buffer is ready for transmission. The character is transmitted at a higher priority than the other characters in the transmit buffer (if any), but does not pre-empt characters already in the transmit FIFO.

The eighth entry in the UART control characters table is defined as follows:

E — Empty
Must be one to use this entry as a flow control transmission character. To use this entry instead as a receive control characters entry, this E bit (and all other E bits in the table) should be zero.

R — Reject
Must be zero to use this entry as a flow control transmission character. For a receive control characters entry, it maintains its functionality.

REA — Ready
This bit is set by the M68000 core when the character is ready for transmission and will remain one while the character is being tansmitted. The CP clears this bit after transmission.

I — Interrupt
If set, the M68000 core will be interrupted when this character has been transmitted. (The TX bit will be set in the UART event register.)

CT — Clear-to-Send Lost
This status bit indicates that the $\overline{\text{CTS}}$ signal was negated during transmission of this character. If this occurs, the CTS bit in the UART event register will also be set.

NOTE

If the $\overline{\text{CTS}}$ signal was negated during transmission, and the CP transmits this character in the middle of buffer transmission, the $\overline{\text{CTS}}$ signal could actually have been negated either during this character's transmission or during a buffer character's transmission. In this case, the CP sets the CT bit both here and in the Tx BD status word.

A — Address
When working in a multidrop configuration, the user should include the address bit in this position.

CHARACTER8
Any 7- or 8-bit character value may be transmitted. This value may be modified only while the REA bit is cleared. A 7-bit character should comprise bits 6–0.

4.5.11.8 SEND BREAK. A break is an all-zeros character without stop bits — i.e., 9 to 13 continuous zeros. A break is sent by issuing the STOP TRANSMIT command. The UART completes transmission of any outstanding data in the FIFO and then sends 9 to 13 zeros (depending on the UM1–UM0, SL, PEN, and CL bits in the UART mode register). The UART transmits a programmable number of break characters according to the value of the break count register (BRKCR), and then reverts to idle or sends data if the RESTART TRANSMIT command was given before completion. Upon transmission of the entire set of break characters, the transmitter sends at least one high bit before transmitting any data to guarantee recognition of a valid start bit.

4.5.11.9 SEND PREAMBLE (IDLE). A preamble sequence gives the programmer a convenient way of ensuring that the line goes idle before starting a new message. The preamble sequence length is 9 to 13 consecutive ones (depending on the UM1–UM0, SL, PEN, and CL bits in the UART mode register). If the preamble bit in a BD is set, the SCC will send a preamble sequence before transmitting that data buffer.

4.5.11.10 WAKEUP TIMER. By issuing the ENTER HUNT MODE command, the user can temporarily disable the UART receiver. It will remain inactive until an idle or address character is recognized (depending on the setting of UM1–UM0).

If the UART is still in the process of receiving a message that the user has already decided to discard, the message may be aborted by issuing the ENTER HUNT MODE command. The UART receiver will be re-enabled when the message is finished by detecting the idle line (if UM1–UM0 = 00) or by the address bit of the next message (if UM0 = 1).

When the receiver is in sleep mode and a break sequence is received, the receiver will increment the BRKEC counter and generate the BRK interrupt (if enabled).

4.5.11.11 UART ERROR-HANDLING PROCEDURE. The UART controller reports character reception and transmission error conditions through the channel BDs, the error counters, and the UART event register (SCCE). The modem interface lines can also be monitored directly by the SCC status register.

Transmission Error

Clear to Send Lost During Character Transmission. When this error occurs and the channel is not programmed to control this line with software, the channel terminates buffer transmission, closes the buffer, sets the CST lost (CT) bit in the BD, and generates the TX interrupt (if enabled). The channel will resume transmission after the reception of the RESTART TRANSMIT command.

Reception Errors

1. Overrun Error. The UART controller maintains an internal three-byte FIFO for receiving data. The CP begins programming the SDMA channel (if the data buffer is in external memory) when the first byte is received into the FIFO. When a receiver FIFO overrun occurs, the channel writes the received character into the internal FIFO over the previously received character (the previous character and its status bits are lost). Then the channel writes the received character to the buffer, closes the buffer, sets overrun (OV) in the BD, and generates the RX interrupt (if enabled). In automatic multidrop mode, the receiver enters hunt mode immediately.

2. Carrier Detect Lost During Character Reception. When this error occurs and the channel is not programmed to control this line with software, the channel terminates character reception, closes the buffer, sets the carrier detect lost (CD) bit in the BD, and generates the RX interrupt (if enabled). This error's priority is the highest; the last character in the buffer is lost and other errors are not checked. In automatic multidrop mode, the receiver enters hunt mode immediately.

3. Framing Error. Framing error is reported by the UART controller when no stop bit is detected in a received data string. When this error occurs, the channel writes the received character to the buffer, closes the buffer, sets framing error (FR) in the BD, and generates the RX interrupt (if enabled). The channel also increments the framing error counter (FRMEC). When this error occurs, parity is not checked for this character. In automatic multidrop mode, the receiver enters hunt mode immediately.

4. Parity Error. When the parity error occurs, the channel writes the received character to the buffer, closes the buffer, sets parity error (PR) in the BD, and generates the RX interrupt (if enabled). The channel also increments the parity error counter (PAREC). In automatic multidrop mode, the receiver enters hunt mode immediately.

5. Noise Error. Noise error is detected by the UART controller when the three samples taken on every bit are not identical. When this error occurs, the channel writes the received character to the buffer and proceeds normally but increments the noise error counter (NOSEC).

6. IDLE Sequence. Receive IDLE (preamble) is detected by the UART controller when a character with 9 to 13 consecutive ones (depending on the UM1–UM0, SL, PEN, and CL bits in the UART mode register) is received. When an IDLE sequence is received, the channel starts to count the number of IDLE sequences received. If it reaches the MAX_IDL value, the buffer is closed and an RX interrupt is generated (if enabled). The counter is reset every time a character is received.

7. BREAK Sequence. A BREAK sequence is detected by the UART receiver when a character with zero value and framing error is received. When a BREAK sequence is received, the channel will increment the BRKEC counter, close the buffer, set the BR bit (if a buffer was currently open), and generate a BRK interrupt (if enabled). Also, if the channel was in the middle of buffer processing, the buffer is closed and an RX is generated (if enabled). A long break sequence only increments the counter once.

Error Counters

The UART maintains four 16-bit (modulo – 2**16) error counters for the receive portion of each UART controller. They can be initialized by the user when the channel is disabled. The counters are as follows:

PAREC — Parity Error Counter
FRMEC — Framing Error Counter
NOSEC — Noise Error Counter
BRKEC — BREAK Error Counter

4.5.11.12 FRACTIONAL STOP BITS. The UART transmitter can be programmed to transmit fractional stop bits. Three bits in the SCC data synchronization register (DSR) are used to program the length of the last stop bit transmitted. These DSR bits may be modified at any time. If two stop bits are transmitted, only the second one is affected. Idle characters are always transmitted as

full-length characters. In UART mode, bits 14–12 in the DSR are now decoded as follows:

14–12 of DSR

111	Last Transmit Stop Bit	16/16 (the default value after reset)
110	Last Transmit Stop Bit	15/16
. . . .		
001	Last Transmit Stop Bit	10/16
000	Last Transmit Stop Bit	9/16

The UART receiver can always receive fractional stop bits. The next character's start bit may begin anytime after the 11th internal clock of the previous character's first stop bit (the UART uses a 16×clock).

4.5.11.13 UART MODE REGISTER. Each SCC mode register is a 16-bit, memory-mapped, read-write register that controls the SCC operation. The term UART mode register refers to the protocol-specific bits (15-6) of the SCC mode register when that SCC is configured as a UART. The read-write UART mode register is cleared by reset.

15	14	13	12	11	10	9	8	7	6	5 … 0
TPM1	TPM0	RPM	PEN	UM1	UM0	FRZ	CL	RSTM	SL	COMMON SCC MODE BITS

TPM1–TPM0 — Transmitter Parity Mode
TMP1–TMP0 select the type of parity to be performed.
00 = Odd parity; always send an odd number of ones.
01 = Force low parity; always send a zero in the parity bit position.
10 = Even parity; always send an even number of ones.
11 = Force high parity; always send a one in the parity bit position.

RPM — Receiver Parity Mode
0 = Odd parity
1 = Even parity
When odd parity is selected, the receiver will count the number of ones in the data word. If the total number of ones is not an odd number, the parity bit is set to one to produce an odd number of ones. If the receiver counts an even number of ones, an error in transmission has occurred. Similarly, for even parity, an even number of ones must result from the calculation performed at both ends of the line.

PEN — Parity Enable

0 = No parity

1 = Parity is enabled for the transmitter and receiver as determined by the parity mode bits.

UM1–UM0 — UART Mode 1–0

00 = Normal UART operation. Multidrop mode is disabled for point-to-point operation and an idle-line wakeup is selected. In the idle-line wakeup mode, the UART receiver is re-enabled by an idle string of 9 to 13 consecutive ones (depending on character length and parity mode).

01 = In the multidrop mode, an addition address/data bit is transmitted with each character. The multidrop asynchronous modes are compatible with the Motorola MC68681 DUART, the Motorola MC68HC11 SCI interface, and the Motorola DSP56000 SCI interface. UM0 is also used to select the wakeup mode before enabling the receiver or issuing the ENTER HUNT MODE command.

Multidrop mode is enabled and an address bit wakeup is selected. In the address bit wakeup mode, the UART receiver is re-enabled when the last data bit (the 8th or 9th) in a character is one. This configuration means that the received character is an address, which should be processed by all inactive processors. The IMP receives the address character and writes it to a new buffer. No address recognition is performed.

10 = The DDCMP protocol is implemented over the asynchronous channel.

11 = Multidrop mode is enabled as in the 01 case, and the IMP automatically checks the address of the incoming address character and either accepts or discards the data following the address.

FRZ — Freeze Transmission

This bit allows the user to halt the UART transmitter and to continue transmission from the next character in the buffer at a later time.

0 = Normal operation (or resume transmission after FRZ is set).

1 = The UART completes transmission of any data already transferred to the UART FIFO (up to three characters) and then stops transmitting data.

CL — Character Length

0 = 7-bit character length. On receive, bit 7 in memory is written as zero. On transmit, bit 7 in memory is a don't care.

1 = 8-bit character length

RTSM — RTS Mode

0 = $\overline{\text{RTS}}$ is asserted whenever the transmitter is enabled and there are characters to transmit. $\overline{\text{RTS}}$ is negated after the last stop bit of a transmitted character when both the shift register and the transmit FIFO are empty. RTS is also negated at the end of a buffer to guarantee accurate reporting of the CTS bit in the BD.

1 = $\overline{\text{RTS}}$ is asserted whenever the transmitter is enabled.

SL — Stop Length

This bit selects the number of the stop bits transmitted by the UART. The receiver is always enabled for one stop bit. Fractional stop bits are configured in the DSR (see **4.5.11.12 FRACTIONAL STOP BITS**).

0 = One stop bit

1 = Two stop bits

4.5.11.14 UART RECEIVE BUFFER DESCRIPTOR (Rx BD). The CP reports information about each buffer of received data by its BDs. The Rx BD is shown in Figure 4-11. The CP closes the current buffer, generates a maskable interrupt, and starts to receive data in the next buffer due to any of the following events:

1. Reception of a user-defined control character (when reject (R) bit = 0)
2. Detection of an error during message processing
3. Detection of a full receive buffer
4. Reception of a programmable number of consecutive IDLE characters
5. Reception of an address character when working in multidrop mode

NOTE

In the nonautomatic multidrop mode (UM1–UM0 = 01), the address character will be written into the next buffer for comparison by the user software.

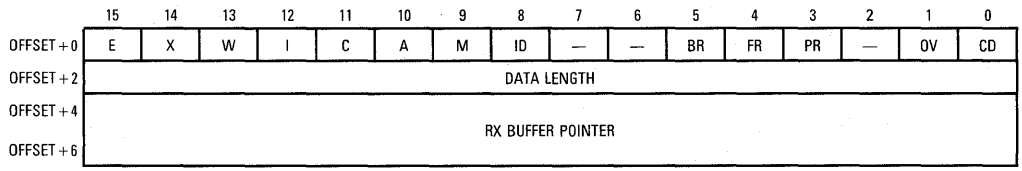


Figure 4-11. UART Receive Buffer Descriptor

The first word of the Rx BD contains the control and status bits.

E — Empty
- 0 = The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The M68000 core is free to examine or write to any fields of the BD.
- 1 = The data buffer associated with the BD is empty. This bit is used to signify that the BD and its associated buffer are available to the CP. After it sets this bit, the M68000 core should not write to any fields of this BD when this bit is set. Note that the empty bit will remain set while the CP is currently filling the buffer with received data.

X — External Buffer
- 0 = The buffer associated with this BD is in internal dual-port RAM.
- 1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)
- 0 = This is not the last BD in the Rx BD table.
- 1 = This is the last BD in the Rx BD table. After this buffer has been used, the CP will receive incoming data into the first BD in the table, allowing the user to use fewer than eight BDs to conserve internal RAM.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt
- 0 = No interrupt is generated after this buffer has been filled.
- 1 = The M68000 core will be interrupted when this buffer has been completely filled by the CP, indicating the need for the M68000 core to process the buffer. (The RX bit in the UART event register will be set to cause the interrupt.)

The following bits contain status information written by the CP after it has finished receiving data in the associated data buffer.

C — Control Character
- 0 = This buffer does not contain a control character.
- 1 = This buffer contains a user-defined control character in the last byte location.

A — Address

0 = The buffer contains data only.

1 = When working in nonautomatic multidrop mode (UM1–UM0 = 01), this bit indicates that the first byte of this buffer contains an address byte. The address comparison should be implemented in software. In automatic multidrop mode, this bit indicates that the BD contains a message received immediately following an address recognized in UADDR1 or UADDR2. This address is not written into the receive buffer.

M — Address Match

This bit is meaningful only if the A bit (bit 10) is set and UM1–UM0 = 11 in the UART mode register. Following an address match, this bit defines which address character matched the user-defined address character, enabling the UART to receive the data.

0 = The address-matched user-defined UADDR2

1 = The address-matched user-defined UADDR1

ID — Buffer Closed on Reception of IDLES

The buffer was closed due to the reception of the programmable number of consecutive IDLE sequences (defined in MAX_IDL).

Bits 7–6, 2 — Reserved for future use.

BR — Break Received

A break sequence was received while receiving data into this buffer.

FR — Framing Error

A character with a framing error was received and is located in the last byte of this buffer. A framing error is detected by the UART controller when no stop bit is detected in the receive data string.

PR — Parity Error

A character with a parity error was received and is located in the last byte of this buffer.

OV — Overrun

A receiver overrun occurred during message reception.

CD — Carrier Detect Lost

The carrier detect signal was negated during message reception.

Data Length

Data length contains the number of octets written by the CP into this BD's data buffer.

NOTE

The actual amount of memory allocated for this buffer should be greater than or equal to the contents of maximum receive buffer length register (MRBLR).

Rx Buffer Pointer

The receive buffer pointer, which always points to the first location of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.11.15 UART TRANSMIT BUFFER DESCRIPTOR (Tx BD). Data is presented to the CP for transmission on an SCC channel by arranging it in buffers referenced by the channel's Tx BD table. The CP confirms transmission (or indicates error conditions) through the BDs to inform the M68000 core that the buffers have been serviced. The Tx BD shown in Figure 4-12.

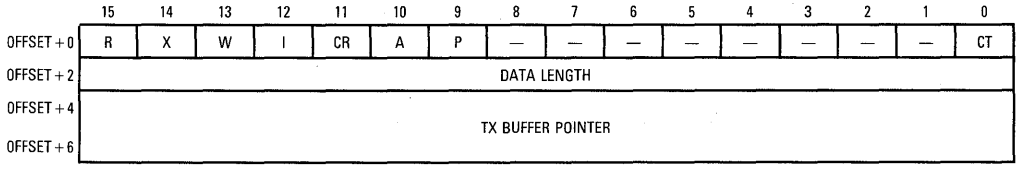


Figure 4-12. UART Transmit Buffer Descriptor

The first word of the Tx BD contains status and control bits. The following bits are prepared by the user before transmission and set by the CP after the buffer has been transmitted.

R — Ready

0 = This buffer is not currently ready for transmission. The user is free to manipulate the BD (or its associated buffer). The CP clears this bit after the buffer has been transmitted or after an error condition has been encountered.

1 = The data buffer, which has been prepared for transmission by the user, has not been transmitted or is currently transmitting. No fields of this BD may be written by the user once this bit is set.

X — External Buffer

0 = The buffer associated with this BD is in internal dual-port RAM.

1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)

0 = This is not the last BD in the Tx BD table.

1 = This is the last BD in the Tx BD table. After this buffer has been used, the CP will transmit data from the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt

0 = No interrupt is generated after this buffer has been serviced.

1 = The M68000 core will be interrupted when this buffer has been serviced by the CP. (The TX bit in the UART event register will be set to cause the interrupt.)

Bits 8–1 — Reserved for future use.

CR — Clear-to-Send Report

This bit allows a choice of no delay between buffers transmitted in UART mode (two bits of idle between buffers) versus a more accurate CTS lost error reporting.

0 = The buffer following this buffer will be transmitted with no delay (assuming it is ready), but the CT bit may not be set in the correct Tx BD, or may not be set at all in a CTS lost condition. The user is advised to monitor the CTS bit in the UART event register for an indication of CTS lost, in addition to the CT bits in the Tx BDs. The CTS bit will always be set properly.

1 = Normal CTS lost (CT bit) error reporting, and two bits of idle occur between back-to-back buffers.

A — Address

This bit is valid only in multidrop mode (UM0 = 1).

0 = This buffer contains data only.

1 = Set by the M68000 core, this bit indicates that this buffer contains address character(s). All the buffer's data will be transmitted as address characters.

P — Preamble

0 = No preamble sequence is sent.

1 = The UART sends a preamble sequence (set of 9 to 13 bits) before sending the data.

The following bits are written by the CP after it has finished transmitting the associated data buffer.

CT — CTS Lost

0 = The $\overline{\text{CTS}}$ signal remained active during transmission.

1 = The $\overline{\text{CTS}}$ signal was negated during transmission.

Data Length

The data length is the number of octets that the CP should transmit from this BD's data buffer. This value should be greater than zero.

Tx Buffer Pointer

The transmit buffer pointer, which always points to the first location of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.11.16 UART EVENT REGISTER. The SCC event register (SCCE) is called the UART event register when the SCC is operating as a UART. It is an 8-bit register used to report events recognized by the UART channel. On recognition of an event, the UART controller will set the corresponding bit in the UART event register. Interrupts generated by this register may be masked in the UART mask register.

The UART event register is a memory-mapped register that may be read at any time. A bit is reset by writing a one (writing a zero does not affect a bit's value). More than one bit may be reset at a time. All unmasked bits must be reset before the CP will clear the internal interrupt request. This register is cleared by reset.

7	6	5	4	3	2	1	0
CTS	CD	IDL	BRK	CCR	BSY	TX	RX

CTS — Clear-To-Send Status Changed
A change in the status of the $\overline{CTS}$ line was detected on the UART channel. The SCC status register may be read to determine the current status.

CD — Carrier Detect Status Changed
A change in the status of the $\overline{CD}$ line was detected on the UART channel. The SCC status register may be read to determine the current status.

IDL — IDLE Sequence Status Changed
A change in the status of the serial line was detected on the UART channel. The SCC status register may be read to determine the current status.

BRK — Break Character Received
A break character was received.

CCR — Control Character Received
A control character was received (with reject (R) character = 1) and stored in the receive control character register (RCCR).

BSY — Busy Condition
A character was received and discarded due to lack of buffers. The receiver enters hunt mode immediately.

TX — Tx Buffer
A buffer has been transmitted over the UART channel.

RX — Rx Buffer
A buffer has been received over the UART channel.

4.5.11.17 UART MASK REGISTER. The SCC mask register (SCCM) is referred to as the UART mask register when the SCC is operating as a UART. It is an 8-bit read-write register with the same bit formats as the UART event register. If a bit in the UART mask register is a one, the corresponding interrupt in the event register will be enabled. If the bit is zero, the corresponding interrupt in the event register will be masked. This register is cleared upon reset.

4.5.11.18 S-RECORDS PROGRAMMING EXAMPLE. In the following paragraphs, an example of a downloading application is given that utilizes an SCC channel as a UART controller. The application performs downloads and uploads of S records between a host computer and an intelligent peripheral through a serial asynchronous line.

The S records are strings of ASCII characters that begin with 'S' and end in an end-of-line character. This characteristic will be used to impose a message structure on the communication between the devices. Note that each device may also transmit XON and XOFF characters for flow control, which do not form part of the program being uploaded or downloaded.

The UART mode register should be set as required, with the freeze (FRZ) bit cleared and the enable transmitter/receiver (ENT, ENR) bits set. Receive buffers should be linked to the receive buffer table with the interrupt (I) bit set. For simplicity, assume that the line is not multidrop (no addresses are transmitted) and that each S record will fit into a single data buffer.

Three characters should first be entered into the UART control character table:

1. End of Line — The empty (E) bit is cleared; the reject (R) bit is cleared. When an end-of-line character is received, the current buffer is closed (the next BD taken by the IMP) and made available to the M68000 core for processing. This buffer contains an entire S record, which the processor can now check and copy to memory or disk as required.

2. XOFF — E should be cleared and R should be set. Whenever the M68000 core receives a control character received interrupt and the receive control character register contains XOFF, it should immediately stop transmitting to the other station by setting the FRZ bit in the UART mode

register. This prevents data from being lost by the other station when it runs out of receive buffers.

3. XON — XON should be received after XOFF. E should be cleared and R should be set. The FRZ bit on the transmitter should now be cleared. The IMP automatically resumes transmission of the serial line at the point at which it was previously stopped. Like XOFF, the XON character is not stored in the receive buffer.

To receive the S records, the M68000 core must only wait for the RX interrupt, indicating the reception of a complete S-record buffer. Transmission requires assembling S records into data buffers and linking them to the transmit buffer table (transmission may be temporarily halted by reception of an XOFF character). This scheme minimizes the number of interrupts received by the M68000 core (one per S record) and relieves it from the task of continually scanning for control characters.

4.5.12 HDLC Controller

When the MODE1–MODE0 bits of an SCC mode register (SCM) select the HDLC mode, then that SCC functions as an HDLC controller. The HDLC controller handles the basic functions of the HDLC/SDLC protocol on either the D channel, a B channel, or from a multiplexed serial interface (IDL, GCI (IOM-2), or PCM highway). When the HDLC controller is used to support the B or D channel of the ISDN, the SCC outputs are internally connected to the physical layer serial interface.

When an SCC in HDLC mode is used with a nonmultiplexed modem interface, then the SCC outputs are connected directly to the external pins. In this case, the serial interface uses seven dedicated pins: transmit data (TXD), receive data (RXD), receive clock (RCLK), transmit clock (TCLK), carrier detect ($\overline{CD}$), clear to send ($\overline{CTS}$), and request to send ($\overline{RTS}$). Other modem signals may be supported through the parallel I/O pins.

The HDLC controller consists of separate transmit and receive sections whose operations are asynchronous with the M68000 core and may be either synchronous or asynchronous with respect to the other SCCs. Up to eight frames may be transmitted or received without M68000 core intervention. When the HDLC controller is connected to the physical layer serial interface (ISDN applications), the receive and transmit clocks are identical and are supplied by the physical layer. In non-ISDN applications, each clock can be supplied either from the baud rate generator or externally. The baud rate generator is discussed more fully in **4.5.2 SCC Configuration Register (SCON)**.

The HDLC controller key features are as follows:

- Flexible Data Buffers with Multiple Buffers per Frame Allowed
- Separate Interrupts for Frames and Buffers (Receive and Transmit)
- Four Address Comparison Registers with Mask
- Maintenance of Five 16-Bit Error Counters
- Flag/Abort/Idle Generation/Detection
- Zero Insertion/Deletion
- NRZ/NRZI Data Encoding
- 16-Bit or 32-Bit CRC-CCITT Generation/Checking
- Detection of Nonoctet Aligned Frames
- Detection of Frames That Are Too Long
- Programmable Flags (0–15) between Successive Frames
- Automatic Retransmission in Case of Collision

A typical HDLC frame is shown in Figure 4-13.

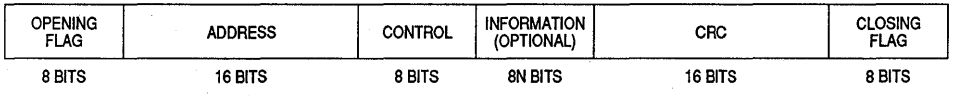


Figure 4-13. Typical HDLC Frame

4.5.12.1 HDLC CHANNEL FRAME TRANSMISSION PROCESSING. The HDLC transmitter is designed to work with almost no intervention from the M68000 core. When the M68000 core enables one of the transmitters, it will start transmitting flags or idles as programmed in the HDLC mode register. The HDLC controller will poll the first buffer descriptor (BD) in the transmit channel's BD table. When there is a frame to transmit, the HDLC controller will fetch the data from memory and start transmitting the frame (after first transmitting the user-specified minimum number of flags between frames). When the end of the current BD has been reached and the last buffer in the frame bit is set, the cyclic redundancy check (CRC), if selected, and the closing flag are appended.

Following the transmission of the closing flag, the HDLC controller writes the frame status bits into the BD and clears the ready bit. When the end of

the current BD has been reached, and the last bit is not set (working in multibuffer mode), only the ready bit is cleared. In either mode, an interrupt is issued according to the interrupt bit in the BD. The HDLC controller will then proceed to the next BD in the table. In this way, the user may be interrupted after each buffer, after a specific buffer has been transmitted, or after each frame.

To rearrange the transmit queue before the IMP has completed transmission of all buffers, issue the STOP TRANSMIT command. This technique can be useful for transmitting expedited data before previously linked buffers or for error situations. When receiving the STOP TRANSMIT command, the HDLC controller will abort the current frame being transmitted and start transmitting idles or flags. When the HDLC controller is given the RESTART TRANSMIT command, it resumes transmission.

4.5.12.2 HDLC CHANNEL FRAME RECEPTION PROCESSING. The HDLC receiver is also designed to work with almost no intervention from the M68000 core. The HDLC receiver can perform address recognition, CRC checking, and maximum frame length checking. The received frame is available to the user for performing any HDLC-based protocol.

When the M68000 core enables one of the receivers, the receiver waits for an opening flag character. When the receiver detects the first byte of the frame, the HDLC controller will compare the frame address against the user-programmable addresses. The user has four 16-bit address registers and an address mask available for address matching. The HDLC controller will compare the received address field to the user-defined values after masking with the address mask. The HDLC controller can also detect broadcast (all ones) addressed frames, if one address register is written with all ones.

If a match is detected, the HDLC controller will fetch the next BD and, if empty, will start to transfer the incoming frame to the BD's associated data buffer. When the data buffer has been filled, the HDLC controller clears the empty bit in the BD and generates an interrupt if the interrupt bit in the BD is set. If the incoming frame exceeds the length of the data buffer, the HDLC controller will fetch the next BD in the table and, if it is empty, will continue to transfer the rest of the frame to this BD's associated data buffer.

During this process, the HDLC controller will check for a frame that is too long. When the frame ends, the CRC field is checked against the recalculated value and is written to the data buffer. The data length written to the last BD in the HDLC frame is the length of the entire frame. This enables HDLC

protocols that "lose" frames to correctly recognize the frame-too-long condition. The HDLC controller then sets the last buffer in frame bit, writes the frame status bits into the BD, and clears the empty bit. The HDLC controller next generates a maskable interrupt, indicating that a frame has been received and is in memory. The HDLC controller then waits for a new frame. Back-to-back frames may be received with only a single shared flag between frames.

4.5.12.3 HDLC MEMORY MAP. When configured to operate in HDLC mode, the IMP overlays the structure shown in Table 4-4 onto the protocol-specific area of that SCC parameter RAM. Refer to **2.8 MC68302 MEMORY MAP** for the placement of the three SCC parameter RAM areas and to Table 4-2 for the other parameter RAM values.

Table 4-4. HDLC-Specific Parameter RAM

Address	Name	Width	Description
SCC Base+9C	RCRC_L	Word	Temp Receive CRC Low
SCC Base+9E	RCRC_H	Word	Temp Receive CRC High
SCC Base+A0 *	C_MASK_L	Word	Constant ($F0B8 16-Bit CRC, $DEBB 32-Bit CRC)
SCC Base+A2 *	C_MASK_H	Word	Constant ($F0B8 16-Bit CRC, $20E3 32-Bit CRC)
SCC Base+A4	TCRC_L	Word	Temp Transmit CRC Low
SCC Base+A6	TCRC_H	Word	Temp Transmit CRC High
SCC Base+A8 *	DISFC	Word	Discard Frame Counter
SCC Base+AA *	CRCEC	Word	CRC Error Counter
SCC Base+AC *	ABTSC	Word	Abort Sequence Counter
SCC Base+AE *	NMARC	Word	Nonmatching Address Received Counter
SCC Base+B0 *	RETRC	Word	Frame Retransmission Counter
SCC Base+B2 *	MFLR	Word	Max Frame Length Register
SCC Base+B4	MAX_cnt	Word	Max_Length Counter
SCC Base+B6 *	HMASK	Word	User-Defined Frame Address Mask
SCC Base+B8 *	HADDR1	Word	User-Defined Frame Address
SCC Base+BA *	HADDR2	Word	User-Defined Frame Address
SCC Base+BC *	HADDR3	Word	User-Defined Frame Address
SCC Base+BE *	HADDR4	Word	User-Defined Frame Address

*Initialized by the user (M68000 core).

4.5.12.4 HDLC PROGRAMMING MODEL. The M68000 core configures each SCC to operate in one of four protocols by the MODE1–MODE0 bits in the SCC mode register (SCM). MODE1–MODE0=00 selects HDLC mode. The HDLC controller uses the same data structure as the UART, BISYNC, and DDCMP controllers. This data structure supports multibuffer operation and address comparisons.

The receive errors (overrun, nonoctet aligned frame, $\overline{CD}$ lost, aborted frame, and CRC error) are reported through the receive BD. The transmit errors (underrun and $\overline{CTS}$ lost) are reported through the transmit BD. An indication about the status of the lines (idle, $\overline{CD}$, and $\overline{CTS}$) is reported through the SCC status register (SCCS), and a maskable interrupt is generated upon a status change in any one of those lines.

4.5.12.5 HDLC COMMAND SET. The following commands are issued to the command register.

STOP TRANSMIT Command

After a hardware or software reset and the enabling of the channel in the SCC mode register, the channel is in the transmit enable mode and starts polling the first BD in the table approximately every eight transmit clocks.

The channel STOP TRANSMIT command disables the transmission of frames on the transmit channel. If this command is received by the HDLC controller during frame transmission, transmission of that frame is aborted after the contents of the FIFO are transmitted (up to four words), and the TBD# is made to point to the next BD. No new BD is accessed, and no new frames are transmitted for this channel. The transmitter will transmit an abort sequence (if the command was given during frame transmission) and then begin to transmit flags or idles as indicated by the HDLC mode register.

This command is useful for performing frame retransmission. The M68000 core may issue the STOP TRANSMIT command, reorganize the transmit BD table, and issue the RESTART TRANSMIT command. The STOP TRANSMIT command may also be used in the X.25 protocol to send a reject frame or a link reset command.

The STOP TRANSMIT command must be issued before the SCC mode register is used to disable the transmitter if the transmitter is to be re-enabled at a later time.

RESTART TRANSMIT Command

The RESTART TRANSMIT command re-enables the transmission of characters on the transmit channel. This command is expected by the HDLC controller after a STOP TRANSMIT command, after a STOP TRANSMIT command and disabling the channel in its SCC mode register, or after

transmitter error (underrun or CTS lost when no automatic frame retransmission is performed). The HDLC controller will resume transmission from the current transmitter BD (TBD#) in the channel's transmit BD table.

If the transmitter is being re-enabled, the RESTART TRANSMIT command must be used and should be followed by the enabling of the transmitter in the SCC mode register.

ENTER HUNT MODE Command

After a hardware or software reset and the enabling of the channel by its SCC mode register, the channel is in the receive enable mode and will use the first BD in the table.

The ENTER HUNT MODE command is generally used to force the HDLC receiver to abort reception of the current frame and enter the hunt mode. In the hunt mode, the HDLC controller continually scans the input data stream for the flag sequence. After receiving the command, the current receive buffer is closed, and the CRC is reset. Further frame reception will use the next BD.

If an enabled receiver has been disabled by clearing ENR in the SCC mode register), the ENTER HUNT MODE command must be given to the channel before setting ENR again. Subsequent frames will then be received, starting with the next BD.

4.5.12.6 HDLC ADDRESS RECOGNITION. Each HDLC controller has five 16-bit registers for address recognition: one mask register and four address registers (HMASK, HADDR1, HADDR2, HADDR3, and HADDR4). The HDLC controller reads the frame's address from the HDLC receiver, checks it against the four address register values, and then masks the result with the user-defined HMASK. A one in HMASK represents a bit position for which address comparison should occur; a zero represents a masked bit position. Upon an address match, the address and the data following are written into the data buffers. When the addresses are not matched and the frame is error-free, the nonmatching address received counter (NMARC) is incremented.

NOTE

For 8-bit addresses, mask out the eight high-order bits in the HMASK register.

4.5.12.7 HDLC MAXIMUM FRAME LENGTH REGISTER (MFLR). The HDLC controller checks the length of an incoming HDLC frame against the user-defined value given in this 16-bit register. If this limit is exceeded, the remainder of the incoming HDLC frame is discarded, and the LG (Rx frame too long) bit is set in the last BD belonging to that frame. The HDLC controller waits to the end of the frame and reports the frame status and the frame length in the last BD. MFLR is defined as all the in-frame bytes between the opening flag and the closing flag (address, control, data, and CRC). MAX_CNT is a temporary downcounter used to track the frame length.

4.5.12.8 HDLC ERROR-HANDLING PROCEDURE. The HDLC controller reports frame reception and transmission error conditions using the channel BDs, the error counters, and the HDLC event register. The modem interface lines can also be directly monitored in the SCC status register.

Transmission Errors:

1. Transmitter Underrun. When this error occurs, the channel terminates buffer transmission, closes the buffer, sets the underrun (UN) bit in the BD, and generates the TXE interrupt (if enabled). The channel will resume transmission after the reception of the RESTART TRANSMIT command. The transmit FIFO size is four words.

2. Clear-To-Send Lost (Collision) During Frame Transmission. When this error occurs and the channel is not programmed to control this line with software, the channel terminates buffer transmission, closes the buffer, sets the CTS lost (CT) bit in the BD, and generates the TXE interrupt (if enabled). The channel will resume transmission after the RESTART TRANSMIT command is given.

NOTE

If this error occurs on the first or second buffer of the frame and the retransmit enable (RTE) bit in the HDLC mode register is set, the channel will retransmit the frame when the $\overline{\text{CTS}}$ line becomes active again. When working in ISDN mode with D-channel collision possibility, to ensure the retransmission method functions properly, the first and second data buffers should contain more than 10 bytes of data if multiple buffers per frame are used. (Small frames consisting of a single buffer are not subject to this requirement). The channel will also increment the retransmission counter (RETRC).

Reception Errors:

1. Overrun Error. The HDLC controller maintains an internal three-word FIFO for receiving data. The CP begins programming the SDMA channel (if the data buffer is in external memory) and updating the CRC when the first word is received in the FIFO. When a receive FIFO overrun occurs, the channel writes the received data byte to the internal FIFO over the previously received byte. The previous data byte and the frame status are lost. Then the channel closes the buffer with the overrun (OV) bit in the BD set and generates the RXF interrupt (if enabled). The receiver then enters the hunt mode.

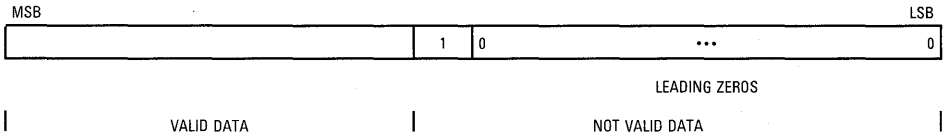
   Even if the overrun occurs during a frame whose address is not matched in the address recognition logic, a BD of length two will be opened to report the overrun, and the RXB interrupt will be generated (if enabled).

2. Carrier Detect Lost During Frame Reception. When this error occurs and the channel is not programmed to control this line with software, the channel terminates frame reception, closes the buffer, sets the carrier detect lost (CD) bit in the BD, and generates the RXF interrupt (if enabled). This error has the highest priority. The rest of the frame is lost, and other errors are not checked in that frame. The receiver then enters the hunt mode.

3. Abort Sequence. An abort sequence is detected by the HDLC controller when seven or more consecutive ones are received. When this error occurs, the channel closes the buffer by setting the Rx abort sequence (AB) bit in the BD and generates the RXF interrupt (if enabled). The channel also increments the abort sequence counter (ABTSC). The CRC and nonoctet error status conditions are not checked on aborted frames. The receiver then enters hunt mode.

4. Nonoctet Aligned Frame. When this error occurs, the channel writes the received data to the data buffer, closes the buffer, sets the Rx nonoctet aligned frame (NO) bit in the BD, and generates the RXF interrupt (if enabled). The CRC error status should be disregarded on nonoctet frames. After a nonoctet aligned frame is received, the receiver enters hunt

mode. (An immediately following back-to-back frame will be received.) The nonoctet data may be derived from the last word in the data buffer as follows:



5. CRC Error. When this error occurs, the channel writes the received CRC to the data buffer, closes the buffer, sets the CR bit in the BD, and generates the RXF interrupt (if enabled). The channel also increments the CRC error counter (CRCEC). After receiving a frame with a CRC error, the receiver enters hunt mode. (An immediately following back-to-back frame will be received.) CRC checking cannot be disabled, but the CRC error may be ignored if checking is not required.

Error Counters

The CP maintains five 16-bit (modulo – 2**16) error counters for each HDLC controller. They can be initialized by the user when the channel is disabled. The counters are as follows:

DISFC — Discarded Frame Counter (error-free frames but no free buffers)
CRCEC — CRC Error Counter (includes frames not addressed to the user or frames received in the BSY condition, but does not include overrun errors)
ABTSC — Abort Sequence Counter
NMARC — Nonmatching Address Received Counter (error-free frames only)
RETRC — Frame Retransmission Counter (due to collision)

4.5.12.9 HDLC MODE REGISTER. Each SCC mode register is a 16-bit, memory-mapped, read-write register that controls the SCC operation. The term HDLC mode register refers to the protocol-specific bits (15-6) of the SCC mode register when that SCC is configured for HDLC. The read-write HDLC mode register is cleared by reset.

15	14	13	12	11	10	9	8	7	6	5	0
NOF3	NOF2	NOF1	NOF0	C32	FSE	—	RTE	FLG	ENC	COMMON SCC MODE BITS	

NOF3–NOF0 — Minimum Number of Flags between Frames or before Frames (0 to 15 Flags)

If NOF3–NOF0 = 0000, then no flags will be inserted between frames. Thus, the closing flag of one frame will be followed immediately by the opening flag of the next frame in the case of back-to-back frames.

C32 — CRC16/CRC32

0 = 16-bit CCITT CRC ($X^{16}+X^{12}+X^5+1$)

1 = 32-bit CCITT CRC ($X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X^1+1$)

FSE — Flag Sharing Enable

0 = Normal operation

1 = If NOF3–NOF0 = 0000, then a single shared flag is transmitted between back-to-back frames. Other values of NOF3–NOF0 are decremented by one when FSE is set. This is useful in Signalling System #7 applications.

Bit 9 — Reserved for future use.

RTE — Retransmit Enable

0 = No retransmission

1 = Retransmit enable

Retransmission only occurs if the $\overline{CTS}$ lost happens on the first or second buffer of the frame.

FLG — Transmit Flags/Idles between Frames and Control the $\overline{RTS}$ Pin

0 = Send ones between frames; $\overline{RTS}$ is negated between frames. The HDLC controller can transmit ones in both the NRZ and NRZI data encoding formats.

1 = Send flags between frames. $\overline{RTS}$ is always asserted.

ENC — Data Encoding Format

0 = Non-return to zero (NRZ). A one is a high level; a zero is a low level.

1 = Non-return to zero inverted (NRZI). A one is represented by no change in the level; a zero is represented by a change in the level. The receiver decodes NRZI, but a clock must be supplied. The transmitter encodes NRZI.

4.5.12.10 HDLC RECEIVE BUFFER DESCRIPTOR (Rx BD). The HDLC controller uses the Rx BD to report information about the received data for each buffer. The Rx BD is shown in Figure 4-14.

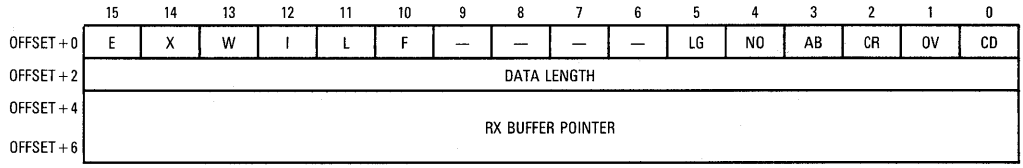


Figure 4-14. HDLC Receive Buffer Descriptor

The first word of the Rx BD contains control and status bits. Bits 15–10 are written by the user before the buffer is linked to the Rx BD table, and bits 5–0 are set by the CP following frame reception. Bit 15 is set by the M68000 core when the buffer is available to the HDLC controller; it is cleared by the HDLC controller when the buffer is full.

4

E — Empty

0 = The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The M68000 core is free to examine or write to any fields of the BD.

1 = The data buffer associated with the BD is empty. This bit signifies that the BD and its associated buffer are available to the HDLC controller. The M68000 core should not write to any fields of this BD after it sets this bit. The empty bit will remain set while the HDLC controller is currently filling the buffer with received data.

X — External Buffer

0 = The buffer associated with this BD is in internal dual-port RAM.

1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)

0 = This is not the last BD in the Rx BD table.

1 = This is the last BD in the Rx BD table. After this buffer has been used, the HDLC controller will receive incoming data into the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant operation may occur.

I — Interrupt
0 = No interrupt is generated after this buffer has been used.
1 = The M68000 core will be interrupted when this buffer has been used by the HDLC controller. (The RXB or RXF bits in the HDLC event register will be set to cause the interrupt.)

The following status bits are written by the HDLC controller after the received data has been placed into the associated data buffer.

L — Last in Frame
This bit is set by the HDLC controller when this buffer is the last in a frame. This implies the reception of a closing flag or reception of an error, in which case one or more of the CD, OV, AB, and LG bits are set. The HDLC controller will write the number of frame octets to the data length field.
0 = This buffer is not the last in a frame.
1 = This buffer is the last in a frame.

F — First in Frame
This bit is set by the HDLC controller when this buffer is the first in a frame.
0 = The buffer is not the first in a frame.
1 = The buffer is the first in a frame.

Bits 9–6 — Reserved for future use.

LG — Rx Frame Length Violation
A frame length greater than the maximum defined for this channel was recognized (only the maximum-allowed number of bytes is written to the data buffer).

NO — Rx Nonoctet Aligned Frame
A frame that contained a number of bits not exactly divisible by eight was received.

AB — Rx Abort Sequence
A minimum of seven consecutive ones was received during frame reception.

CR — Rx CRC Error
This frame contains a CRC error.

OV — Overrun

A receiver overrun occurred during frame reception.

CD — Carrier Detect Lost

The carrier detect signal was negated during frame reception. This bit is valid only when working in NMSI mode.

Data Length

The data length is the number of octets written to this BD's data buffer by the HDLC controller. When this BD is the last BD in the frame (L = 1), the data length contains the total number of frame octets (including two or four bytes for CRC).

NOTE

The actual amount of memory allocated for this buffer should be even and greater than or equal to the contents of maximum receive buffer length register (MRBLR).

Rx Buffer Pointer

The receive buffer pointer, which always points to the first location of the associated data buffer, may reside in either internal or external memory.

NOTE

The Rx buffer pointer must be even.

4.5.12.11 HDLC TRANSMIT BUFFER DESCRIPTOR (Tx BD). Data is presented to the HDLC controller for transmission on an SCC channel by arranging it in buffers referenced by the channel's Tx BD table. The HDLC controller confirms transmission (or indicates error conditions) using the BDs to inform the M68000 core that the buffers have been serviced. The Tx BD is shown in Figure 4-15.

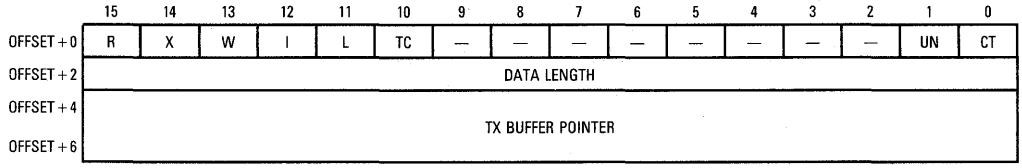


Figure 4-15. HDLC Transmit Buffer Descriptor

The first word of the Tx BD contains status and control bits. Bits 15–10 are prepared by the user before transmission; bits 1–0 are set by the HDLC controller after the buffer has been transmitted. Bit 15 is set by the user when the buffer and BD have been prepared and is cleared by the HDLC controller after the frame has been transmitted.

R — Ready

0 = This buffer is not currently ready for transmission. The user is free to manipulate this BD (or its associated buffer). The HDLC controller clears this bit after the buffer has been fully transmitted or after an error condition has been encountered.

1 = The data buffer, which has been prepared for transmission by the user, has not yet transmitted. No fields of this BD may be written by the user once this bit is set.

X — External Buffer

0 = The buffer associated with this BD is in internal dual-port RAM.

1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)

0 = This is not the last BD in the Tx BD table.

1 = This is the last BD in the Tx BD table. After this buffer has been used, the HDLC controller will transmit data from the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt

0 = No interrupt is generated after this buffer has been serviced.

1 = The M68000 core will be interrupted when this buffer has been serviced by the HDLC controller. (At least one of TXB or TXE in the HDLC event register will be set to cause the interrupt.)

L — Last

0 = This is not the last buffer in the frame.

1 = This is the last buffer in the current frame.

TC — Tx CRC

This bit is valid only when the last (L) bit is set.

0 = Transmit only the idle/flag sequence after the last data byte.

1 = Transmit the CRC sequence after the last data byte.

Bits 9–2 — Reserved for future use.

The following status bits are written by the HDLC controller after it has finished transmitting the associated data buffer.

UN — Underrun
The HDLC controller encountered a transmitter underrun condition while transmitting the associated data buffer.

CT — CTS Lost
CTS in NMSI mode or L1GR (layer-1 grant) in IDL/GCI mode was lost during frame transmission.

Data Length
The data length is the number of octets the HDLC controller should transmit from this BD's data buffer. The value of this field should be greater than zero.

Tx Buffer Pointer
The transmit buffer pointer, which contains the address of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.12.12 HDLC EVENT REGISTER. The SCC event register (SCCE) is called the HDLC event register when the SCC is operating as an HDLC controller. It is an 8-bit register used to report events recognized by the HDLC channel. Upon recognition of an event, the HDLC controller sets its corresponding bit in the HDLC event register. Interrupts generated by this register may be masked in the HDLC mask register.

The HDLC event register is a memory-mapped register that may be read at any time. A bit is reset by writing a one; writing a zero does not affect a bit's value. More than one bit may be reset at a time. All unmasked bits must be reset before the CP will clear the internal interrupt request. This register is cleared by reset.

7	6	5	4	3	2	1	0
CTS	CD	IDL	TXE	RXF	BSY	TXB	RXB

CTS — Clear-To-Send Status Changed
A change in the status of the $\overline{\text{CTS}}$ line was detected on the HDLC channel. The SCC status register may be read to determine the current status.

CD — Carrier Detect Status Changed
A change in the status of the $\overline{CD}$ line was detected on the HDLC channel. The SCC status register may be read to determine the current status.

IDL — IDLE Sequence Status Changed
A change in the status of the serial line was detected on the HDLC channel. The SCC status register may be read to determine the current status.

TXE — Tx Error
An error (CTS lost or underrun) occurred on the transmitter channel.

RXF — Rx Frame
A complete frame has been received on the HDLC channel.

BSY — Busy Condition
A frame was received and discarded due to lack of buffers.

TXB — Tx Buffer
A buffer has been transmitted on the HDLC channel.

RXB — Rx Buffer
A buffer has been received on the HDLC channel that was not a complete frame.

4.5.12.13 HDLC MASK REGISTER. The SCC mask register (SCCM) is referred to as the HDLC mask register when the SCC is operating as an HDLC controller. It is an 8-bit read-write register that has the same bit formats as the HDLC event register. If a bit in the HDLC mask register is a one, the corresponding interrupt in the event register will be enabled. If the bit is zero, the corresponding interrupt in the event register will be masked. This register is cleared upon reset.

4.5.13 BISYNC Controller

By appropriately setting the SCC mode register, any of the SCC channels may be configured to function as a BISYNC controller. The BISYNC controller handles the basic functions of the BISYNC protocol in normal mode and in transparent mode.

The SCC in BISYNC mode can work with IDL, GCI (IOM2), PCM highway, or NMSI interfaces. When the SCC in BISYNC mode is used with a modem interface (NMSI), the SCC outputs are connected directly to the external pins. The modem interface uses seven dedicated pins: transmit data (TXD), receive data (RXD) receive clock (RCLK), transmit clock (TCLK), carrier detect ($\overline{CD}$), clear to send ($\overline{CTS}$), and request to send ($\overline{RTS}$). Other modem lines can be supported using the parallel I/O pins.

The BISYNC controller consists of separate transmit and receive sections whose operations are asynchronous with the M68000 core and may be either synchronous or asynchronous with respect to the other SCCs. Each clock can be supplied from either the internal baud rate generator or from external pins. More information on the baud rate generator is available in **4.5.2 SCC Configuration Register (SCON)**.

The main BISYNC controller features are as follows:

- Flexible Data Buffers
- Eight Control Character Recognition Registers
- Automatic SYNC1 and SYNC2 Detection
- SYNC/DLE Stripping and Insertion
- CRC16 and LRC Generation/Checking
- Parity (VRC) Generation/Checking
- Supports BISYNC Transparent Operation (Use of DLE Characters)
- Supports Promiscuous (Totally Transparent) Reception and Transmission
- Maintains Parity Error Counter
- External SYNC Support
- Reverse Data Mode
- Four Commands

Typical BISYNC frames are shown in Figure 4-16.

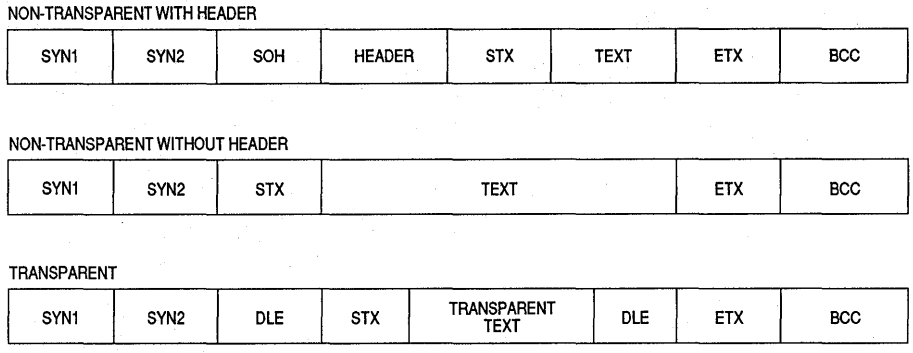


Figure 4-16. Typical BISYNC Frames

4.5.13.1 BISYNC CHANNEL FRAME TRANSMISSION PROCESSING. The BISYNC transmitter is designed to work with almost no intervention from the M68000 core. When the M68000 core enables the BISYNC transmitter, it will start transmitting SYN1–SYN2 pairs (located in the data synchronization register) or idle as programmed in the BISYNC mode register. The BISYNC controller polls the first buffer descriptor (BD) in the transmit channel's BD table. When there is a message to transmit, the BISYNC controller will fetch the data from memory and start transmitting the message (after first transmitting the SYN1–SYN2 pair).

When a BD's data has been completely transmitted, the last in message (L) bit is checked. If both the L bit is set, and transmit BCS bit are set in that BD, the BISYNC controller will append the CRC16/LRC. Subsequently, the BISYNC controller writes the message status bits into the BD and clears the ready bit. It will then start transmitting SYNCs or IDLEs as programmed in the BISYNC mode register. When the end of the current BD has been reached and the last bit is not set (working in multibuffer mode), only the ready bit is cleared. In both cases, an interrupt is issued according to the interrupt (I) bit in the BD. By appropriately setting the I bit in each BD, interrupts can be generated after the transmission of each buffer, a specific buffer, or each block. The BISYNC controller will then proceed to the next BD in the table.

If no additional buffers have been presented to the BISYNC controller for transmission, an in-frame underrun is detected, and the BISYNC controller begins transmitting either SYNCs or IDLEs. If the BISYNC controller was in transparent mode, the BISYNC controller transmits DLE-SYNC pairs.

Characters are included in the block check sequence (BCS) calculation on a per-buffer basis. Each buffer can be independently programmed to be included or excluded from the BCS calculation, and any characters to be excluded from the BCS calculation must reside in a separate buffer. The BISYNC controller can reset the BCS generator before transmitting a specific buffer. When functioning in transparent mode, the BISYNC controller automatically inserts a DLE before transmitting a DLE character. In this case, only one DLE is used in the calculation of the BCS.

The BISYNC controller may also be used to transmit characters in a promiscuous (totally transparent) mode. See **4.5.9 SCC Transparent Mode Support**.

4.5.13.2 BISYNC CHANNEL FRAME RECEPTION PROCESSING. Although the BISYNC receiver is designed to work with almost no intervention from the M68000 core, it allows user intervention on a per-byte basis if necessary. The BISYNC receiver can perform CRC16, longitudinal redundancy check (LRC), or vertical redundancy check (VRC) checking, SYNC stripping in normal mode, DLE-SYNC stripping and stripping of the first DLE in DLE-DLE pairs in transparent mode, and control character recognition. A control character is one belonging to the control characters shown in Figure 4-17.

When the M68000 core enables the BISYNC receiver, it will enter hunt mode. In this mode, as data is shifted into the receiver shift register one bit at a time, the contents of the register are compared to the contents of the SYN1–SYN2 fields in the data synchronization register. If the two are not equal, the next bit is shifted in, and the comparison is repeated. When the registers match, the hunt mode is terminated, and character assembly begins. The BISYNC controller is now character synchronized and will perform SYNC stripping and message reception. The BISYNC controller will revert to the hunt mode when it is issued the ENTER HUNT MODE command, upon recognition of some error condition, or upon reception of an appropriately defined control character.

When receiving data, the BISYNC controller updates the BCS bit (CR) in the BD for every byte transferred. When the data buffer has been filled, the BISYNC controller clears the empty (E) bit in the BD and generates an interrupt if the interrupt (I) bit in the BD is set. If the incoming data exceeds the length of the data buffer, the BISYNC controller will fetch the next BD in the table and, if it is empty, will continue to transfer data to this BD's associated data buffer.

When a BCS is received, it is checked and written to the data buffer. The BISYNC controller sets the last bit, writes the message status bits into the BD, and clears the empty bit. Then it generates a maskable interrupt, indicating that a block of data has been received and is in memory. Note that the SYNC in the nontransparent mode or DLE-SYNC pairs in the transparent mode (i.e., an underrun condition) are not included in the BCS calculations.

The BISYNC controller may also be used to receive characters in a promiscuous (totally transparent) mode. See **4.5.9 SCC Transparent Mode Support**.

4.5.13.3 BISYNC MEMORY MAP. When configured to operate in BISYNC mode, the IMP overlays the structure illustrated in Table 4-5 onto the protocol-specific area of that SCC parameter RAM. Refer to **2.8 MC68302 MEMORY MAP** for the placement of the three SCC parameter RAM areas and Table 4-2 for the other parameter RAM values.

Table 4-5. BISYNC-Specific Parameter RAM

Address	Name	Width	Description
SCC Base+9C	RCRC	Word	Temp Receive CRC
SCC Base+9E	CRCC	Word	CRC Constant
SCC Base+A0 *	PRCRC	Word	Preset Receiver CRC16/LRC
SCC Base+A2	TCRC	Word	Temp Transmit CRC
SCC Base+A4 *	PTCRC	Word	Preset Transmitter CRC16/LRC
SCC Base+A6	RES	Word	Reserved
SCC Base+A8	RES	Word	Reserved
SCC Base+AA *	PAREC	Word	Receive Parity Error Counter
SCC Base+AC *	BSYNC	Word	BISYNC SYNC Character
SCC Base+AE *	BDLE	Word	BISYNC DLE Character
SCC Base+B0 *	CHARACTER1	Word	CONTROL Character 1
SCC Base+B2 *	CHARACTER2	Word	CONTROL Character 2
SCC Base+B4 *	CHARACTER3	Word	CONTROL Character 3
SCC Base+B6 *	CHARACTER4	Word	CONTROL character 4
SCC Base+B8 *	CHARACTER5	Word	CONTROL Character 5
SCC Base+BA *	CHARACTER6	Word	CONTROL Character 6
SCC Base+BC *	CHARACTER7	Word	CONTROL Character 7
SCC Base+BE *	CHARACTER8	Word	CONTROL Character 8

*Initialized by the user (M68000 core).

The M68000 core configures each SCC to operate in one of four protocols by the MODE1–MODE0 bits in the SCC mode register. MODE1–MODE0=11 selects the BISYNC mode of operation. The SYN1–SYN2 synchronization characters are programmed in the data synchronization register (see **4.5.4 Data Synchronization Register (DSR)**).

The BISYNC controller uses the same basic data structure as the UART, HDLC, and DDCMP controllers. Receive and transmit errors are reported through their respective BDs. The status of the line is reflected in the SCC status register, and a maskable interrupt is generated upon each status change.

There are two basic ways of handling the BISYNC channels. First, data may be inspected on a per-byte basis, with the BISYNC controller interrupting the M68000 core upon receipt of every byte of data. Second, the BISYNC controller may be operated so that software is only necessary for handling the first two to three bytes of data; subsequent data (until the end of the block) can be handled by the BISYNC controller without interrupting the M68000 core. See **4.5.13.14 PROGRAMMING THE BISYNC CONTROLLERS** for more information.

4.5.13.4 BISYNC COMMAND SET. The following commands are issued to the command register.

STOP TRANSMIT Command

After a hardware or software reset and the enabling of the channel using the SCC mode register, the channel is in the transmit enable mode and starts polling the first BD in the table approximately every eight transmit clocks.

The STOP TRANSMIT command aborts transmission after the contents of the FIFO are transmitted (up to three bytes in BISYNC and up to four words in promiscuous mode) without waiting until the end of the buffer is reached. SYNC characters consisting of SYNC-SYNC or DLE-SYNC pairs (according to the transmitter mode) will be continually transmitted until transmission is re-enabled by issuing the RESTART TRANSMIT command. The STOP TRANSMIT command may be used when it is necessary to abort transmission and transmit an EOT control sequence. The EOT sequence should be the first buffer presented to the BISYNC controller for transmission after re-enabling transmission.

The STOP TRANSMIT command must be issued before the SCC mode register is used to disable the transmitter if the transmitter is to be re-enabled at a later time.

NOTE

The BISYNC controller will remain in the transparent or normal mode after receiving the STOP TRANSMIT or RESTART TRANSMIT commands.

RESTART TRANSMIT Comand

The RESTART TRANSMIT command is used to begin or resume transmission from the current Tx BD number (TBD#) in the channel's Tx BD table. When this command is received by the channel, it will start polling the ready bit in this BD. This command is expected by the BISYNC controller after a STOP TRANSMIT command, after the STOP TRANSMIT command and the disabling of the channel in its mode register, or after a transmitter error (underrun or CTS lost) occurs.

If the transmitter is being re-enabled, the RESTART TRANSMIT command must be used and should be followed by the enabling of the transmitter in the SCC mode register.

RESET BCS CALCULATION Command

The RESET BCS CALCULATION command resets the receive BCS accumulator immediately. For example, it may be used to reset the BCS after recognizing a control character, signifying that a new block is commencing (such as SOH).

ENTER HUNT MODE Command

After a hardware or software reset and the enabling of the channel in the SCC mode register, the channel is in the receive enable mode and will use the first BD in the table.

The ENTER HUNT MODE command is used to force the BISYNC controller to abort reception of the current block and enter the hunt mode. In the hunt mode, the BISYNC controller continually scans the input data stream for the SYN1–SYN2 sequence as programed in the data synchronization register. After receiving the command, the current receive buffer is closed, and the BCS is reset. Message reception continues using the next BD.

If an enabled receiver has been disabled (by clearing ENR in the SCC mode register), the ENTER HUNT MODE command must be given to the channel before setting ENR again.

4.5.13.5 BISYNC CONTROL CHARACTER RECOGNITION. The BISYNC controller can recognize special control characters. These characters are used to "customize" the BISYNC protocol implemented by the BISYNC controller and may be used to aid its operation in a DMA-oriented environment. Their main use is for receive buffers longer than one byte. In single-byte buffers, each byte can easily be inspected, and control character recognition should be disabled.

The purpose of the control characters table is to enable automatic recognition (by the BISYNC controller) of the end of the current block. See **4.5.13.14 PROGRAMMING THE BISYNC CONTROLLERS** for more information. Since the BISYNC controller imposes no restrictions on the format of the BISYNC blocks, user software must respond to the received characters and inform the BISYNC controller of mode changes and certain protocol events (e.g., resetting the BCS). However, correct use of the control characters table allows the remainder of the block to be received without interrupting the user software.

Up to eight control characters may be defined. These characters inform the BISYNC controller that the end of the current block has been reached and whether a BCS is expected following this character. For example, the end of text (ETX) character implies both an end of block (ETB) and a BCS should be received. An enquiry (ENQ) character designates end of block without a subsequent BCS. All the control characters are written into the data buffer.

The BISYNC controller uses a table of 16-bit entries to support control character recognition. Each entry consists of the control character, an end-of-table bit, a BCS expected bit, and a hunt mode bit. The control characters table is shown in Figure 4-17.

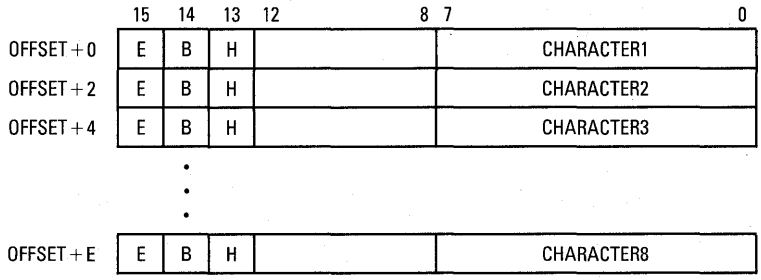


Figure 4-17. BISYNC Control Characters Table

CHARACTER8–CHARACTER1 — Control Character Value
These fields define control characters.

NOTE

When using 7-bit characters with parity, the parity bit should be included in the control character value.

E — End of Table

0 = This entry is valid. The lower eight bits will be checked against the incoming character.

1 = The entry is not valid. No valid entries exist beyond this entry.

NOTE

In tables with eight control characters, E should be zero in all eight positions.

B — BCS Expected

0 = The character is written into the receive buffer. The buffer is then closed.

1 = The character is written into the receive buffer. The receiver waits for one LRC or two CRC bytes of BCS and then closes the buffer. This should be used for ETB, ETX, and ITB.

NOTE

A maskable interrupt is generated after the buffer is closed.

H — Enter Hunt Mode

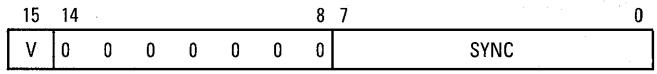
0 = The BISYNC controller will maintain character synchronization after closing this buffer.

1 = The BISYNC controller will enter hunt mode after closing the buffer. When the B bit is set, the controller will enter hunt mode after the reception of the BCS.

4.5.13.6 BSYNC-BISYNC SYNC REGISTER. The 16-bit, memory-mapped, read-write BSYNC register is used to define the BISYNC stripping and insertion of the SYNC character. When an underrun occurs during message transmission, the BISYNC controller will insert SYNC characters until the next data buffer is available for transmission. When the BISYNC receiver is not in hunt mode and a SYNC character has been received, the receiver will discard this character if the valid (V) bit is set.

NOTE

When using 7-bit characters with parity, the parity bit should be included in the SYNC register value.

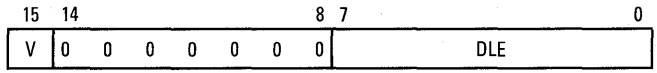


4.5.13.7 BDLE-BISYNC DLE REGISTER. The 16-bit, memory-mapped, read-write BDLE register is used to define the BISYNC stripping and insertion of the DLE character. When the BISYNC controller is in transparent mode and an underrun occurs during message transmission, the BISYNC controller inserts DLE-SYNC pairs until the next data buffer is available for transmission.

When the BISYNC receiver is in transparent mode and a DLE character is received, the receiver discards this character and excludes it from the BCS if the valid (V) bit is set. If the second (next) character is a SYNC character, the BISYNC controller discards it and excludes it from the BCS. If the second character is a DLE, the BISYNC controller will write it to the buffer and include it in the BCS. If the character is not a DLE or SYNC, the BISYNC controller will examine the control characters table and act accordingly. If the character is not in the table, the buffer will be closed with the DLE follow character error (DL) bit set. If the V bit is not set, the receiver will treat the character as a normal character.

NOTE

When using 7-bit characters with parity, the parity bit should be included in the DLE register value.



4.5.13.8 BISYNC ERROR-HANDLING PROCEDURE. The BISYNC controller reports message reception and transmission error conditions using the channel BDs, the error counters, and the BISYNC event register. The modem interface lines can also be directly monitored in the SCC status register.

Transmission Errors:

1. Transmitter Underrun. When this error occurs, the channel terminates buffer transmission, closes the buffer, sets the underrun (UN) bit in the BD, and generates the TXE interrupt (if enabled). The channel resumes transmission after the reception of the RESTART TRANSMIT command. Underrun cannot occur between frames. The FIFO size is three bytes in BISYNC and four words in promiscuous (totally transparent) mode.

2. Clear-To-Send Lost During Message Transmission. When this error occurs and the channel is not programmed to control this line with software, the channel terminates buffer transmission, closes the buffer, sets the CTS lost (CT) bit in the BD, and generates the TXE interrupt (if

enabled). The channel will resume transmission after the reception of the RESTART TRANSMIT command.

Reception Errors:

1. Overrun Error. The BISYNC controller maintains an internal three-byte FIFO for receiving data. The FIFO size is three words in promiscuous mode. The CP begins programming the SDMA channel (if the data buffer is in external memory) and updating the CRC when the first word is received into the FIFO. If a FIFO overrun occurs, the BISYNC controller writes the received data byte to the internal FIFO over the previously received byte. The previous character and its status bits are lost. Following this, the channel closes the buffer, sets the overrun (OV) bit in the BD, and generates the RX interrupt (if enabled). The receiver then enters hunt mode immediately.

2. Carrier Detect Lost During Message Reception. When this error occurs and the channel is not programmed to control this line with software, the channel terminates message reception, closes the buffer, sets the carrier detect lost (CD) bit in the BD, and generates the RX interrupt (if enabled). This error is the highest priority; the rest of the message is lost and no other errors are checked in the message. The receiver then enters hunt mode immediately.

3. Parity Error. When this error occurs, the channel writes the received character to the buffer and sets the PR bit in the BD. The channel terminates message reception, closes the buffer, sets the PR bit in the BD, and generates the RX interrupt (if enabled). The channel also increments the parity error counter (PAREC), and the receiver then enters hunt mode immediately.

4. CRC Error. The channel updates the CRC error (CR) bit in the BD every time a character is received, with a byte delay (eight serial clocks) between the status update and the CRC calculation. When using control character recognition to detect the end of the block and cause the checking of the CRC that follows, the channel closes the buffer, sets the CR bit in the BD, and generates the RX interrupt (if enabled).

Error Counter

The CP main controller maintains one 16-bit (modulo – 2**16) error counter for each BISYNC controller. It can be initialized by the user when the channel is disabled. The counter is as follows:

PAREC — Parity Error Counter (on received characters)

4.5.13.9 BISYNC MODE REGISTER. Each SCC mode register is a 16-bit, memory-mapped, read-write register that controls the SCC operation. The term BISYNC mode register refers to the protocol-specific bits (15-6) of the SCC mode register when that SCC is configured for BISYNC. The read-write BISYNC mode register is cleared by reset.

15	14	13	12	11	10	9	8	7	6	5 — 0
PM	EXSYN	NTSYN	REVD	BCS	—	RTR	RBCS	SYNF	ENC	COMMON SCC MODE BITS

PM — Parity Mode
0 = Odd Parity
1 = Even Parity
This bit is valid when the BCS bit is cleared. When odd parity is selected, the transmitter will count the number of ones in the 7-bit data character. If the total is not an odd number, then the parity bit is made equal to one to make an odd number of ones. Then, if the receiver counts an even number of ones, an error in transmission has occurred. In the same manner, for even parity, an even number must result from the calculation performed at both ends of the line.

EXSYN — External Sync Mode
When this mode is selected, the receiver expects external logic to indicate the beginning of the data field using the $\overline{CD1}$/L1SY1 pin, if SCC1 is used, and the $\overline{CD2}$ and $\overline{CD3}$ pins, respectively, if SCC2 or SCC3 are used in this mode. In this mode, there will be no carrier detect function for the SCC.

When the channel is programmed to work through the serial channels physical interface (IDL or GCI) and EXSYN is set, the layer-1 logic carries out the synchronization using the L1SY1 pin. In PCM mode, the L1SY1–L1SY0 pins are used. In NMSI mode, the $\overline{CD}$ pins (and the $\overline{CD}$ timing) are used to synchronize the data.

If this bit is cleared, the BISYNC controller will look for the SYN1–SYN2 sequence in the data synchronization register.

NTSYN — No Transmit SYNC
When this bit is set, the SCC operates in a promiscuous, totally transparent mode. See **4.5.9 SCC Transparent Mode Support** for details.

REVD — Reverse DATA
When this bit is set, the receiver and transmitter will reverse the character bit order, transmitting the most significant bit first. This bit is valid in promiscuous mode.

BCS — Block Check Sequence

0 = LRC

For even LRC, the PRCRC and PTCRC preset registers in the BISYNC-specific parameter RAM should be initialized to zero before the channel is enabled. For odd LRC, the PRCRC and PTCRC registers should be initialized to ones.

The receiver will check character parity when BCS is programmed to LRC and the receiver is not in transparent mode. The transmitter will transmit character parity when BCS is programmed to LRC and the transmitter is not in transparent mode. Use of parity in BISYNC assumes the use of 7-bit data characters.

1 = CRC16

The PRCRC and PTCRC preset registers should be initialized to a preset value of all zeros or all ones before the channel is enabled. In both cases, the transmitter sends the calculated CRC non-inverted, and the receiver checks the CRC against zero. Eight-bit characters (without parity) are configured when CRC16 is chosen.

Bit 10 — Reserved for future use.

RTR — Receiver Transparent Mode

0 = The receiver is placed in normal mode with SYNC stripping and control character recognition operative.

1 = The receiver is placed in transparent mode. SYNCs, DLEs, and control characters are only recognized after a leading DLE character. The receiver will calculate the CRC16 sequence, even if programmed to LRC while in transparent mode. PRCRC should be first initialized to the CRC16 preset value before setting this bit.

RBCS — Receive Block Check Sequence

The BISYNC receiver internally stores two BCS calculations with a byte delay (eight serial clocks) between them. This enables the user to examine a received data byte and then decide whether or not it should be part of the BCS calculation. This is useful when control character recognition and stripping is desired to be performed in software. The bit should be set (or reset) within the time taken to receive the following data byte. When this bit is reset, the BCS calculations exclude the latest fully received data byte. When RBCS is set, the BCS calculations continue normally.

0 = Disable receive BCS

1 = Enable receive BCS

SYNF — Transmit SYN1–SYN2 or IDLE between Messages and Control the $\overline{RTS}$ Pin

0 = Send ones between messages; $\overline{RTS}$ is negated between messages. The BISYNC controller can transmit ones in both NRZ and NRZI encoded formats.

1 = Send SYN1–SYN2 pairs between messages; $\overline{RTS}$ is always asserted.

ENC — Data Encoding Format

0 = Non-return to zero (NRZ). A one is a high level; a zero is a low level.

1 = Non-return to zero inverted (NRZI). A one is represented by no change in the level; a zero is represented by a change in the level. The receiver decodes NRZI, but a clock must be supplied. The transmitter encodes NRZI.

4.5.13.10 BISYNC RECEIVE BUFFER DESCRIPTOR (Rx BD). The CP reports information about the received data for each buffer using BD. The Rx BD is shown in Figure 4-18. The CP closes the current buffer, generates a maskable interrupt, and starts to receive data into the next buffer after one of the following events:

Receiving a user-defined control character
Detecting an error
Detecting a full receive buffer
Issuing the ENTER HUNT MODE command

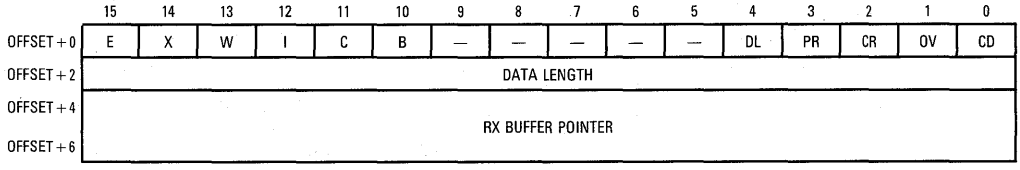


Figure 4-18. BISYNC Receive Buffer Descriptor

The first word of the Rx BD contains control and status bits.

E — Empty

0 = The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The M68000 core is free to examine or write to any fields of this BD.

1 = The data buffer associated with this BD is empty. This bit signifies that the BD and its associated buffer are available to the CP. After it sets this bit, the M68000 core should not write to any fields of this

BD when this bit is set. The empty bit will remain set while the CP is currently filling the buffer with received data.

X — External Buffer
0 = The buffer associated with this BD is in internal dual-port RAM.
1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)
0 = This is not the last BD in the Rx BD table.
1 = This is the last BD in the Rx BD table. After this buffer has been used, the CP will receive incoming data into the first BD in the table. Setting this bit allows the use of fewer than eight BD to conserve internal RAM.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt
0 = No interrupt is generated after this buffer has been used.
1 = The M68000 core will be interrupted when this buffer has been closed by the BISYNC controller. (The RX bit in the BISYNC event register will be set to cause this interrupt.)

The following status bits are written by the CP after the received data has been placed into the associated data buffer.

C — Control Character
The last byte in the buffer is a user-defined control character.
0 = The last byte of this buffer does not contain a control character.
1 = The last byte of this buffer contains a control character.

B — BCS Received
The last bytes in the buffer contain the received BCS.
0 = This buffer does not contain the BCS.
1 = This buffer contains the BCS. A control character may also reside one byte prior to this BCS.

Bits 9–5 — Reserved for future use.

DL — DLE Follow Character Error
While in transparent mode, a DLE character was received, and the next

character was not DLE, SYNC, or a valid entry in the control characters table.

PR — Parity Error
A character with a parity error was received and is the last byte of this buffer.

CR — BCS Error
BCS error (CR) is updated every time a byte is written into the buffer. The CR bit includes the calculation for the current byte. By clearing the RBCS bit in the BISYNC mode register within eight serial clocks, the user can exclude the current character from the message BCS calculation.

OV — Overrun
A receiver overrun occurred during message reception.

CD — Carrier Detect Lost
The carrier detect signal was negated during message reception.

Data Length
The data length is the number of octets that the CP has written into this BD's data buffer, including the BCS (if selected). Data length should initially be set to zero by the user and is incremented each time a received character is written to the data buffer.

NOTE

The actual buffer size should be greater than or equal to the MRBLR.

Rx Buffer Pointer
The receive buffer pointer, which always points to the first location of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.13.11 BISYNC TRANSMIT BUFFER DESCRIPTOR (Tx BD). Data is presented to the CP for transmission on an SCC channel by arranging it in buffers referenced by the channel's Tx BD table. The CP confirms transmission (or indicates error conditions) using the BDs to inform the processor that the buffers have been serviced. The Tx BD is shown in Figure 4-19.

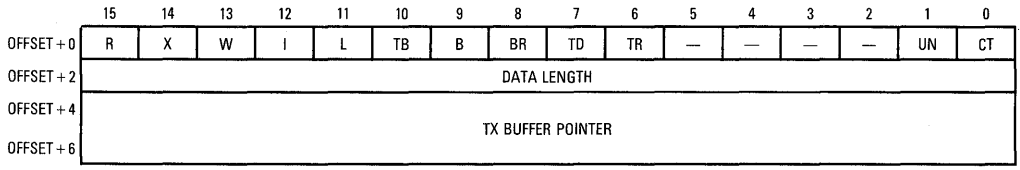


Figure 4-19. BISYNC Transmit Buffer Descriptor

The first word of the Tx BD contains status and control bits. These bits are prepared by the user before transmission and are set by the CP after the buffer has been transmitted.

R — Ready

0 = This buffer is not currently ready for transmission. The user is free to manipulate this BD (or its associated buffer). The CP clears this bit after the buffer has been fully transmitted or after an error condition has been encountered.

1 = The data buffer has been prepared for transmission by the user (but not yet transmitted). No fields of this BD may be written by the user once this bit is set.

X — External Buffer

0 = The buffer associated with this BD is in internal dual-port RAM.

1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)

0 = This is not the last BD in the Tx BD table.

1 = This is the last BD in the Tx BD table. After this buffer has been used, the CP will transmit data from the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt

0 = No interrupt is generated after this buffer has been serviced.

1 = The M68000 core will be interrupted when this buffer has been serviced by the CP. (At least one of TX or TXE in the BISYNC event register will se set to cause this interrupt.)

L — Last in Message

0 = The last character in the buffer is not the last character in the current block.

1 = The last character in the buffer is the last character in the current block. The transmitter will enter (remain in) normal mode after sending the last character in the buffer and the BCS (if enabled).

TB — Transmit BCS

This bit is valid only when the L bit is set.

0 = Transmit the SYN1-SYN2 sequence or IDLE (according to the SYNF bit in the BISYNC mode register) after the last character in the buffer.

1 = Transmit the BCS sequence after the last character. The BISYNC controller will also reset the BCS generator after transmitting the BCS sequence.

B — BCS Enable

0 = Buffer consists of characters to be excluded from the BCS accumulation.

1 = Buffer consists of characters to be included in the BCS accumulation.

BR — BCS Reset

0 = The BCS accumulation is not reset.

1 = The transmitter BCS accumulation is reset (used for STX or SOH) before sending the data buffer.

TD — Transmit DLE

0 = No automatic DLE transmission before the data buffer.

1 = The transmitter will transmit a DLE character before sending the data buffer, which saves writing the first DLE to a separate data buffer when working in transparent mode.

TR — Transparent Mode

0 = The transmitter will enter (remain in) the normal mode after sending the data buffer. In this mode, the transmitter will automatically insert SYNCs in an underrun condition.

1 = The transmitter enters or remains in transparent mode after sending the data buffer. In this mode, the transmitter automatically inserts DLE-SYNC pairs in the underrun condition. Underrun occurs when the BISYNC controller finishes a buffer with L set to zero and the next BD is not available. The transmitter also checks all characters before sending them; if a DLE is detected, another DLE is automatically sent. The user must insert a DLE or program the BISYNC controller to insert it (using TD) before each control character required. The transmitter will calculate the CRC16 BCS even if the BCS bit in the BISYNC mode register is programmed to LRC. The PTCRC should be initialized to the CRC16 preset before setting this bit.

The following status bits are written by the CP after it has finished transmitting the associated data buffer.

UN — Underrun
The BISYNC controller encountered a transmitter underrun condition while transmitting the associated data buffer.

CT — CTS Lost
$\overline{\text{CTS}}$ in NMSI mode or L1GR in IDL/GCI mode was lost during message transmission.

Data Length
The data length is the number of octets that the CP should transmit from this BD's data buffer. The data length should be greater than zero.

Tx Buffer Pointer
The transmit buffer pointer, which always points to the first byte of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.13.12 BISYNC EVENT REGISTER. The SCC event register (SCCE) is referred to as the BISYNC event register when the SCC is programmed as a BISYNC controller. It is an 8-bit register used to report events recognized by the BISYNC channel. On recognition of an event, the BISYNC controller sets the corresponding bit in the BISYNC event register. Interrupts generated by this register may be masked in the BISYNC mask register.

The BISYNC event register is a memory-mapped register that may be read at any time. A bit is reset by writing a one (writing a zero does not affect a bit's value). More than one bit may be reset at a time. All unmasked bits must be reset before the CP will negate the internal interrupt request signal. This register is cleared by reset.

7	6	5	4	3	2	1	0
CTS	CD	—	TXE	RCH	BSY	TX	RX

CTS — Clear-To-Send Status Changed
A change in the status of the serial line was detected on the BISYNC channel. The SCC status register may be read to determine the current status.

CD — Carrier Detect Status Changed
A change in the status of the serial line was detected on the BISYNC channel. The SCC status register may be read to determine the current status.

Bit 5 — Reserved for future use.

TXE — Tx Error
An error (CTS lost or underrun) occurred on the transmitter channel.

RCH — Receive Character
A character has been received and written to the buffer.

BSY — Busy Condition
A character was received and discarded due to lack of buffers. The receiver will resume reception after an ENTER HUNT MODE command.

TX — Tx Buffer
A buffer has been transmitted.

RX — Rx Buffer
A complete buffer has been received on the BISYNC channel. The channel closes the buffer due to one of these events:
Reception of a user-defined control character
Reception of an error
Detection of a full receive buffer

4.5.13.13 BISYNC MASK REGISTER. The SCC mask register (SCCM) is referred to as the BISYNC mask register when the SCC is operating as a BISYNC controller. It is an 8-bit read-write register that has the same bit format as the BISYNC event register. If a bit in the BISYNC mask register is a one, the corresponding interrupt in the event register will be enabled. If the bit is zero, the corresponding interrupt in the event register will be masked. This register is cleared upon reset.

4.5.13.14 PROGRAMMING THE BISYNC CONTROLLERS. There are two general techniques that the software may employ to handle data received by the BISYNC controllers. The simplest way is to allocate single-byte receive buffers, request (in the status word in each BD) an interrupt on reception of each buffer (i.e., byte), and implement the BISYNC protocol entirely in software on a byte-by-byte basis. This simple approach is flexible and may be adapted

to any BISYNC implementation. The obvious penalty is the overhead caused by interrupts on each received character.

A more efficient method is as follows. Multibyte buffers are prepared and linked to the receive buffer table. Software is used to analyze the first (two to three) bytes of the buffer to determine what type of block is being received. When this has been determined, reception can continue without further intervention to the user's software until a control character is encountered. The control character signifies the end of the block, causing the software to revert back to a byte-by-byte reception mode.

To accomplish this, the RCH bit in the BISYNC mask register should initially be set, enabling an interrupt on every byte of data received. This allows the software to analyze the type of block being received on a byte-by-byte basis. After analyzing the initial characters of a block, the user should either set the receiver transparent mode (RTR) bit in the BISYNC mode register or issue the RESET BCS CALCULATION command. For example, if DLE-STX is received, transparent mode should be entered. By setting the appropriate bit in the BISYNC mode register, the BISYNC controller automatically strips the leading DLE from <DLE-character> sequences. Thus, control characters are only recognized when they follow a DLE character. The RTR bit should be cleared after a DLE-ETX is received.

Alternatively, after receiving an SOH, the RESET BCS CALCULATION command should be issued. This command causes the SOH to be excluded from BCS accumulation and the BCS to be reset. Note that the RBCS bit in the BISYNC mode register (used to exclude a character from the BCS calculation) is not needed here since SYNCs and leading DLEs (in transparent mode) are automatically excluded by the BISYNC controller.

After recognizing the type of block above, the RCH interrupt should be masked. Data reception then continues without further interruption of the M68000 core until the end of the current block is reached. This is defined by the reception of a control character matching that programmed in the receive control characters table.

The control characters table should be set to recognize the end of the block as follows:

Control Characters	E	B	H
ETX	0	1	1
ITB	0	1	0
ETB	0	1	1
ENQ	0	0	0
Next Entry	1	X	X

After the end of text (ETX), a BCS is expected; then the buffer should be closed. Hunt mode should be entered when line turnaround occurs. ENQ characters are used to abort transmission of a block. For the receiver, the ENQ character designates the end of the block, but no CRC is expected.

Following control character reception (i.e., end of the block), the RCH bit in the BISYNC mask register should be set, re-enabling interrupts for each byte of received data.

4.5.14 DDCMP Controller

By setting its SCC mode register (SCM), any of the SCC channels may be configured to function as a DDCMP controller. The DDCMP link can be either synchronous (by programming the MODE1–MODE0 bits of the SCC mode register to DDCMP), or asynchronous (by programming the MODE1–MODE0 bits of the SCC mode register to ASYNC and setting the DDCMP bit in the UART mode register). The DDCMP controller handles the basic functions of the DDCMP protocol in both cases.

The SCC in DDCMP mode can work in either IDL, GCI, PCM highway, or NMSI interfaces. When the SCC is used with a modem interface (NMSI), the serial outputs are connected directly to the external pins. The modem interface uses seven dedicated pins: transmit data (TXD), receive data (RXD), receive clock (RCLK), transmit clock (TCLK), carrier detect ($\overline{CD}$), clear to send ($\overline{CTS}$), and request to send ($\overline{RTS}$). Other modem lines can be supported through the parallel I/O pins.

The DDCMP controller consists of separate transmit and receive sections whose operations are asynchronous with the M68000 core and may be either synchronous or asynchronous with respect to the other SCCs. Each clock can be supplied either from the baud rate generator or externally. More infor-

mation on the baud rate generator is available in **4.5.2 SCC Configuration Register (SCON)**.

The DDCMP controller key features are as follows:

- Synchronous or Asynchronous DDCMP Links Supported
- Flexible Data Buffers
- Four Address Comparison Registers with Mask
- Automatic Frame Synchronization
- Automatic Message Synchronization by Searching for SOH, ENQ, or DLE
- CRC16 Generation/Checking
- NRZ/NRZI Data Encoding
- Maintenance of Four 16-Bit Error Counters

Typical DDCMP frames are shown in Figure 4-20.

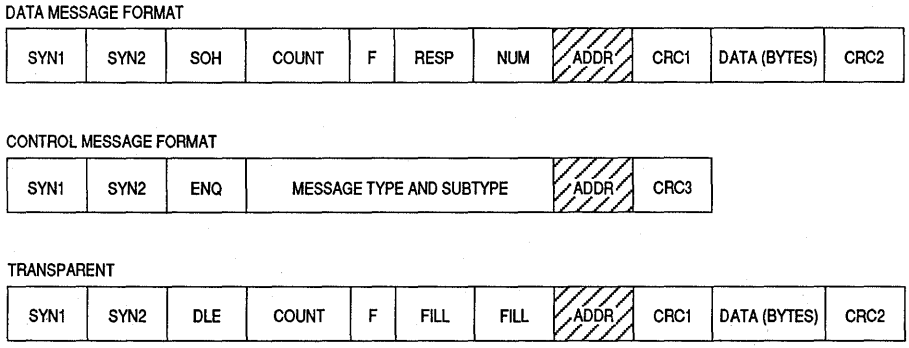


Figure 4-20. Typical DDCMP Frames

4.5.14.1 DDCMP CHANNEL FRAME TRANSMISSION PROCESSING. The DDCMP transmitter is designed to work with almost no intervention from the M68000 core (or other external processor). When the M68000 core enables the DDCMP transmitter and the link is synchronous, it starts transmitting SYN1–SYN2 pairs (programmed in the data synchronization register) or IDLEs as determined in the DDCMP mode register. The DDCMP controller polls the first buffer descriptor (BD) in the channel's transmit BD table. When there is a message to transmit, the DDCMP controller fetches the data from memory and starts transmitting the message (after first transmitting the SYN1–SYN2 pair when the link is synchronous).

When a BD has been completely transmitted, the transmit CRC (TC) bit is checked in the BD. If set, the DDCMP controller appends one of the block checks: CRC1, CRC2, or CRC3 for the header field, data message, or control messages, respectively. Next, the DDCMP controller writes the buffer's status bits into the BD and clears the ready bit in the BD. It then proceeds to the next BD in the table. When the last bit (L) is set and the TC bit is set in that BD, the DDCMP controller appends the CRC2 block check to the data field. This bit is also used for transmitting CRC3 in control messages. Next, it writes the buffer status bits into the BD and clears the ready bit. Finally, on synchronous links, either SYN1–SYN2 pairs or IDLEs (as programmed in the DDCMP mode register) are transmitted. When the end of the current BD has been reached and the last bit is not set (working in multibuffer mode or sending back-to-back messages), only the status bits are written. In either case, when a BD has been completely transmitted, an interrupt is issued if the interrupt (I) bit in the BD is set and the event is not masked in the DDCMP mask register. The appropriate setting of the I bit in each BD allows the user to be interrupted after transmission of each buffer, a specific buffer, or each message.

4.5.14.2 DDCMP CHANNEL FRAME RECEPTION PROCESSING. The DDCMP receiver is also designed to work with almost no intervention from the M68000 core. The DDCMP receiver performs automatic SYN1–SYN2 synchronization on synchronous links and start/stop synchronization on asynchronous links. Automatic message synchronization is achieved by searching for the special starting characters SOH, ENQ, or DLE and making address comparisons with a mask. When the M68000 core enables the DDCMP receiver on synchronous links, it enters hunt mode. In this mode, as data is shifted into the receiver shift register one bit at a time, the contents of the register are compared to the SYN1–SYN2 fields of the data synchronization register (see **4.5.4 SCC Data Synchronization Register (DSR)**). If the two are not equal, the next bit is shifted in, and the comparison is repeated. When the registers match, hunt mode is terminated, and character assembly begins. However, if a character is not SOH, ENQ, DLE, or SYNC, hunt mode is again entered. On asynchronous links, byte synchronization is achieved by the start/stop protocol of the UART. The DDCMP controller is now byte-synchronized and performs SYN1–SYN2 stripping, until receiving one of the three user-defined special starting bytes (SOH for data messages, ENQ for control messages, and DLE for maintenance messages).

If a match is detected, the DDCMP controller fetches the next BD and, if it is empty, starts to transfer the incoming header to the BD's associated data buffer. The DDCMP controller counts the bytes of the fixed-length header

and compares the received header address field to the four user-defined values after masking the result with the address mask. When a match is detected, the DDCMP controller continues to transfer the incoming message to the data buffer. The header CRC field (CRC1) is checked and is written to the data buffer. The DDCMP controller updates the CRC error (CR) bit, sets the header (H) bit, writes the message type and status bits into the BD, and clears the empty bit. It next generates a maskable receive block interrupt (RBK), indicating that a header has been received and is in memory. If the header was a control message, the DDCMP controller waits for a new message.

If there is no match in address comparison and the header is error free, the DDCMP controller will use the same buffer for the next message. To maintain synchronization, the DDCMP controller counts the data length based on the count field contained in the header.

When the data buffer has been filled, the DDCMP controller clears the empty bit in the BD and generates a maskable received buffer interrupt (RBD). If the incoming message exceeds the length of the data buffer, the DDCMP controller fetches the next BD in the table, and, if it is empty, continues to transfer the rest of the message to the new data buffer. When the message ends, the CRC2 field is checked and written to the data buffer. The DDCMP controller sets the last bit, writes the message type and other status bits into the BD, and clears the empty bit. Following this, it generates an RBK, indicating that a message has been received and is in memory. The DDCMP controller then waits for a new message.

4.5.14.3 DDCMP MEMORY MAP. When configured to operate in DDCMP mode, the IMP overlays the structure illustrated in Table 4-6 onto the protocol-specific area of that SCC's parameter RAM. Refer to **2.8 MC68302 MEMORY MAP** for the placement of the three SCC parameter RAM areas and to Table 4-2 for the other parameter RAM values.

Table 4-6. DDCMP-Specific Parameter RAM

Address	Name	Block	Description
SCC Base+9C	RCRC	Word	Temp Receive CRC
SCC Base+9E	CRCC	Word	CRC16 Constant
SCC Base+A0 *	PCRC	Word	Preset CRC16
SCC Base+A2	TCRC	Word	Temp Transmit CRC
SCC Base+A4 *	DSOH	Word	DDCMP SOH Character
SCC Base+A6 *	DENQ	Word	DDCMP ENQ Character
SCC Base+A8 *	DDLE	Word	DDCMP DLE Character
SCC Base+AA *	CRC1EC	Word	CRC1 Error Counter
SCC Base+AC *	CRC2EC	Word	CRC2 Error Counter
SCC Base+AE *	NMARC	Word	Nonmatching Address Received Counter
SCC Base+B0 *	DISMC	Word	Discard Message Counter
SCC Base+B2	RMLG	Word	Received Message Length
SCC Base+B4	RMLG_CNT	Word	Received Message Length Counter
SCC Base+B6 *	DMASK	Word	User Defined Frame Address Mask
SCC Base+B8 *	DADDR1	Word	User Defined Frame Address
SCC Base+BA *	DADDR2	Word	User Defined Frame Address
SCC Base+BC *	DADDR3	Word	User Defined Frame Address
SCC Base+BE *	DADDR4	Word	User Defined Frame Address

*Initialized by the user (M68000 core).

4.5.14.4 DDCMP PROGRAMMING MODEL. The M68000 core configures each SCC to operate in one of four protocols by the MODE1–MODE0 bits in the SCC mode register. If MODE1–MODE0=10, DDCMP operation is selected with synchronous links. For asynchronous links, MODE1–MODE0=01 (ASYNC) should be selected, and the DDCMP bit in the UART mode register should be set. The SYN1–SYN2 synchronization characters are programmed in the data synchronization register (DSR). See **4.5.4 SCC Data Synchronization Register (DSR)** for more programming information. The DDCMP controller uses the same basic data structure as the UART, HDLC, and BISYNC controllers.

The DDCMP controller generates and checks the CRC16 message trailer. It can be preset to ones or zeros by writing to the preset CRC (PCRC) register before enabling the receiver or the transmitter. The received message length (RMLG) is the header byte count value as determined by the receiver, and the received message length counter (RMLG_CNT) is the temporary received data downcounter.

Receive and transmit errors are reported in their respective BDs. The line status signals ($\overline{CD}$ and $\overline{CTS}$) may be read in the SCC status register and a maskable interrupt is generated upon each status change (see **4.5.2 SCC Configuration Register (SCON)**).

4.5.14.5 DDCMP COMMAND SET. The following commands are issued to the command register:

STOP TRANSMIT Command

After a hardware or software reset and the enabling of the channel in the SCC mode register, the channel is in the transmit enable mode and starts polling the first BD in the table approximately every eight transmit clocks.

The channel STOP TRANSMIT command disables the transmission of messages on the transmit channel. If this command is received by the DDCMP controller during message transmission, message transmission is aborted after the contents of the FIFO (up to four bytes) are transmitted. No new BD is accessed, and no new messages are transmitted for this channel. Upon receipt of this command, the transmitter aborts the message transmission (if currently transmitting) and then transmits SYN1–SYN2 pairs or IDLEs as determined by the DDCMP mode register.

The STOP TRANSMIT command must be issued before the SCC mode register is used to disable the transmitter if the transmitter is to be re-enabled at a later time.

RESTART TRANSMIT Command

The RESTART TRANSMIT command re-enables the transmission of characters on the transmit channel. This command is expected by the DDCMP controller after a STOP TRANSMIT command, after a STOP TRANSMIT command followed by the disabling of the channel in its SCC mode register, or after a transmitter error (underrun or CTS lost during data or maintenance message header fields). The DDCMP controller will resume transmission from the current transmitter BD number (TBD#) in the channel's transmit BD table.

If the channel is being re-enabled, the RESTART TRANSMIT command must be used and should be followed by the enabling of the transmitter in the SCC mode register.

ENTER HUNT MODE Command

After a hardware or software reset and the enabling of the channel in the SCC mode register, the channel is in the receive enable mode and will use the first BD in the table.

The ENTER HUNT MODE command is used to force the DDCMP controller to abort reception of the current message and enter hunt mode. In hunt mode, the DDCMP controller continually scans the input data stream for

the SYN1–SYN2 sequence on synchronous links. Then for synchronous or asynchronous links, the DDCMP controller scans the input bytes for the starting byte of one of the messages. After receiving the command, the current receive buffer is closed, and the CRC is reset. Message reception continues using the next BD.

If an enabled receiver has been disabled (by clearing ENR in the SCC mode register), the ENTER HUNT MODE command must be given to the channel before setting ENR again.

4.5.14.6 DDCMP CONTROL CHARACTER RECOGNITION. The DDCMP controller can recognize three special control characters. These characters are used to synchronize the message and allow the DDCMP controller to function in a DMA-controlled environment.

DSOH–DDCMP SOH Register

The 8-bit DSOH register is used to synchronize data messages by the DDCMP controller. When the DDCMP controller is not in hunt mode (byte synchronization is now established), it searches for the SOH character to start processing data messages. The DDCMP controller transfers the header and the data fields of the message to the buffer, checks the header and data CRCs, counts the data field up to the value contained in the header byte count field, and compares the header address field against the user-defined addresses. The DSOH register is a memory-mapped read-write register.

DENQ–DDCMP ENQ Register

The 8-bit DENQ register is used to synchronize control messages by the DDCMP controller. When the DDCMP controller is not in hunt mode (byte synchronization is established), it searches for the ENQ character to start processing control messages. The DDCMP controller transfers the message to the buffer, checks the CRC, and compares the message address field against the user-defined addresses. The DENQ register is a memory-mapped read-write register.

DDLE–DDCMP DLE Register

The 8-bit DDLE register is used to synchronize maintenance messages by the DDCMP controller. When the DDCMP controller is not in hunt mode (byte synchronization is established), it searches for the DLE character to start processing the maintenance messages. The DDCMP controller transfers the header and the data fields of the message to the buffer, checks the header and data CRCs, counts the data field up to the value contained

in the header byte count field, and compares the header address field against the user-defined addresses. The DDLE register is a memory-mapped read-write register.

4.5.14.7 DDCMP ADDRESS RECOGNITION. Each DDCMP controller has five 16-bit registers to support address recognition: one mask register and four address registers (DMASK, DADDR1, DADDR2, DADDR3, and DADDR4). The DDCMP controller reads the message address from the receiver, masks it with the user-defined DMASK bits, and then checks the result against the four address register values. A one in DMASK indicates a bit position where a comparison should take place; a zero masks the comparison. For 8-bit address comparison, the high byte of DMASK should be zero.

4.5.14.8 DDCMP ERROR-HANDLING PROCEDURE. The DDCMP controller reports message reception and transmission errors using the channel BDs, the error counters, and the DDCMP event register. The modem interface lines can also be directly monitored with the SCC status register.

Transmission errors:

1. Transmitter Underrun. When this error occurs, the channel terminates buffer transmission, closes the buffer, sets the underrun (UN) bit in the BD, and generates the transmit error (TXE) interrupt (if enabled). The channel will resume transmission after the reception of the RESTART TRANSMIT command. The FIFO size is three bytes.

NOTE

This error can occur only on synchronous links.

2. Clear-To-Send Lost (Collision) During Message Transmission. When this error occurs and the channel is not programmed to control this line with software, the channel terminates buffer transmission, closes the buffer, sets the CTS lost (CT) bit in the BD, and generates the transmit error (TXE) interrupt (if enabled). The channel resumes transmission after the reception of the RESTART TRANSMIT command.

Reception Errors:

1. Carrier Detect Lost During Message Reception. When this error occurs and the channel is not programmed to control this line with software, the channel terminates message reception, closes the buffer, sets the

carrier detect lost (CD) bit in the BD, and generates the receive block (RBK) interrupt (if enabled). This error has the highest priority. The rest of the message is lost, and other errors in that message are not checked.

The channel will enter hunt mode immediately. It is possible that a SYN1–SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this "header" will have a CRC error.

2. Overrun Error. The DDCMP controller maintains an internal three-byte FIFO for receiving data. The CP begins programming the SDMA channel (if the data buffer is in external memory) and updating the CRC when the first word is received into the FIFO. If the receive FIFO overrun error occurs, the channel writes the received data byte to the internal FIFO on top of the previously received byte. The previous data byte is lost. Then the channel closes the buffer, sets the overrun (OV) bit in the BD, and generates the receive block (RBK) interrupt (if enabled).

   The channel will enter hunt mode immediately. It is possible that a SYN1–SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this "header" will have a CRC error.

3. CRC1 (Header CRC) Error. When this error occurs, the channel writes the received CRC to the data buffer, closes the buffer, sets the CRC error (CR) bit in the BD, generates the RBK interrupt (if enabled), increments the error counter (CRC1EC), and enters hunt mode.

   When this error occurs on data- and maintenance-message header fields, the channel will enter hunt mode immediately. It is possible that a SYN1–SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this "header" will have a CRC error.

4. CRC2 (Data or Maintenance CRC) or CRC3 (Control Message) Error. When this error occurs, the channel writes the received CRC to the data buffer, closes the buffer, sets the CRC error (CR) bit in the BD, and generates the RBK interrupt (if enabled). The channel also increments the CRC2EC counter and enters hunt mode.

5. Framing Error. A framing error is detected by the DDCMP controller when no stop bit is detected in a received data string. When this error occurs, the channel writes the received character to the buffer, closes

the buffer, sets the framing error (FR) bit in the BD, and generates the RBK interrupt (if enabled). When this error occurs, parity is not checked for this character.

The channel will enter hunt mode immediately. It is possible that a SYN1–SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this "header" will have a CRC error.

NOTE

This error can occur only on asynchronous links.

6. Parity Error. When a parity error occurs, the channel writes the received character to the buffer, closes the buffer, sets the parity error (PR) bit in the BD, and generates the RBK interrupt (if enabled).

   The channel will enter hunt mode immediately. It is possible that a SYN1-SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this "header" will have a CRC error.

NOTE

This error can occur only on asynchronous links.

Error Counters

The CP maintains four 16-bit (modulo – 2**16) error counters for each DDCMP controller. They can be initialized by the user when the channel is disabled. The counters are as follows:

CRC1EC — CRC1 Error Counter
CRC2EC — CRC2/CRC3 Error Counter
NMARC — Nonmatching Address Received Counter (updated only when the frame is error-free)
DISMC — Discarded Messages (received messages when there are no free buffers and the frame is error-free)

4.5.14.9 DDCMP MODE REGISTER. Each SCC mode register is a 16-bit, memory-mapped, read-write register that controls the SCC operation. The term DDCMP mode register refers to the protocol-specific bits (15-6) of the SCC mode register when that SCC is configured for DDCMP. The read-write DDCMP mode register is cleared by reset.

15	14	13	12	11	10	9	8	7	6	5	0
NOS3	NOS2	NOS1	NOS0	—	V.110	—	—	SYNF	ENC	COMMON SCC MODE BITS	

NOS3–NOS0 — Minimum Number of SYN1–SYN2 Pairs between Messages (1 to 16 SYNC Pairs)

If NOS3–NOS0 = 0000, then 1 SYNC pair will be transmitted; if NOS3–NOS0 = 1111, then 16 SYNC pairs will be transmitted.

NOTE

With appropriate programming of the transmit BD (TC = 1 and L = 0), it is possible to transmit back-to-back messages.

Bits 11, 9–8 — Reserved for future use.

V.110 — V.110 Mode

0 = DDCMP mode; synchronous DDCMP is chosen.

1 = V.110 mode; the V.110 protocol description is in **4.5.15 V.110 Controller**.

SYNF — Transmit SYN1–SYN2 or IDLE between Messages and Control the $\overline{\text{RTS}}$ Pin

0 = Send ones between messages. $\overline{\text{RTS}}$ is negated between messages.

NOTE

The DDCMP controller can transmit ones in both NRZ and NRZI data encoded formats.

1 = Send SYN1–SYN2 pairs between messages. $\overline{\text{RTS}}$ is always asserted. Note that SYN1 and SYN2 may be the same character.

ENC — Data Encoding Format

0 = Nonreturn to Zero (NRZ). A one is a high level; a zero is a low level.

1 = Nonreturn to Zero Inverted (NRZI). A one is represented by no change in the level; a zero is represented by a change in the level. The receiver decodes NRZI, but a clock must be supplied. The transmitter encodes NRZI.

4.5.14.10 DDCMP RECEIVE BUFFER DESCRIPTOR (Rx BD). The CP reports information about the received data for each buffer using the BDs. The Rx BD is shown in Figure 4-21. The CP closes the current buffer, generates a maskable interrupt, and starts to receive data in the next buffer after any of the following events:

Receiving the received message length number of bytes (RMLG)
Detecting an error
Detecting a full receive buffer
Issuing the ENTER HUNT MODE command

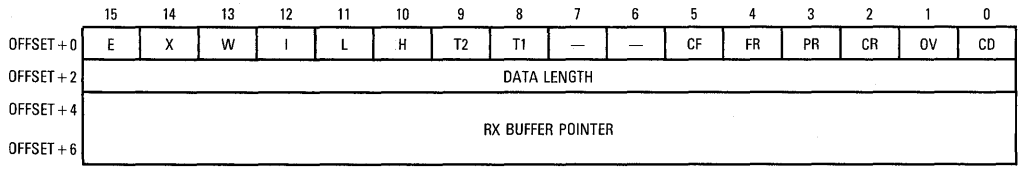


Figure 4-21. DDCMP Receive Buffer Descriptor

The first word of the Rx BD contains control and status bits. Bits 15–12 are written by the user before the buffer is linked to the Rx BD table, and bits 5–0 and 11–8 are set by the IMP following message reception. Bit 15 determines whether the M68000 core or the CP may currently access the BD.

E — Empty
- 0 = The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The M6800 core is free to examine or write to any fields of the BD.
- 1 = The data buffer associated with this BD is empty. This bit signifies that the BD and its associated buffer are available to the DDCMP controller. The M68000 core should not write to any fields of this BD after it sets this bit. Note that the empty bit will remain set while the DDCMP controller is currently filling the buffer with received data.

X — External Buffer
- 0 = The buffer associated with this BD is in internal dual-port RAM.
- 1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)
- 0 = This is not the last BD in the Rx BD table.
- 1 = This is the last BD in the Rx BD table. After this buffer has been used, the DDCMP controller places incoming data in to the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt
0 = No interrupt is generated after this buffer has been closed.
1 = The M68000 core is interrupted when this buffer has been closed by the DDCMP controller. (The RBD or RBK bits in the DDCMP event register will be set to cause this interrupt.)

The following status bits are written by the DDCMP controller after it has finished receiving data in the associated data buffer.

L — Last in Message
0 = The buffer is not the last in a message.
1 = The buffer is the last in a message.

H — Header in Buffer
0 = The buffer does not contain a message header.
1 = The buffer contains a message header.

NOTE

To correctly identify buffers containing headers, the buffer size should be eight or more bytes in length so that the header will fit in a single buffer.

T2,T1 — Message Type
00 = Data message
01 = Control message
10 = Maintenance message
11 = Reserved

Bits 7–6 — Reserved for future use.

CF — CRC Follow Error
The character following the CRC for this message was not one of SOH, ENQ, DLE, SYN, or IDLE. The receiver then enters hunt mode.

FR — Framing Error
A character with a framing error was received. The associated character may be found at the last location in this buffer. A framing error is detected

by the UART controller when no stop bit is detected in the receive data string.

NOTE

This error can occur only on asynchronous DDCMP links.

PR — Parity Error
A character with a parity error was received. The associated character may be found at the last location in this buffer.

NOTE

This error can occur only on asynchronous DDCMP links.

CR — Rx CRC Error
A message with a CRC error was received in the header (CRC1) or data (CRC2) fields or a control message (CRC3).

OV — Overrun
A receiver overrun occurred during message reception.

CD — Carrier Detect Lost
The $\overline{CD}$ signal was deasserted during message reception. This bit is valid only when working in NMSI mode.

Data Length
The data length is the number of octets that the DDCMP controller has written to this BD's data buffer.

NOTE

The actual buffer size should be greater than or equal to eight (to ensure the header is received in one buffer).

Rx Buffer Pointer
This pointer contains the address of the associated data buffer and may be even or odd. The buffer may reside in either internal or external memory.

4.5.14.11 DDCMP TRANSMIT BUFFER DESCRIPTOR (Tx BD). Data is presented to the CP for transmission over an SCC channel by arranging it in buffers referenced by the channel's Tx BD table. The CP confirms transmission (or indicates error conditions) using the BDs to inform the M68000 core that the buffers have been serviced. The Tx BD is shown in Figure 4-22.

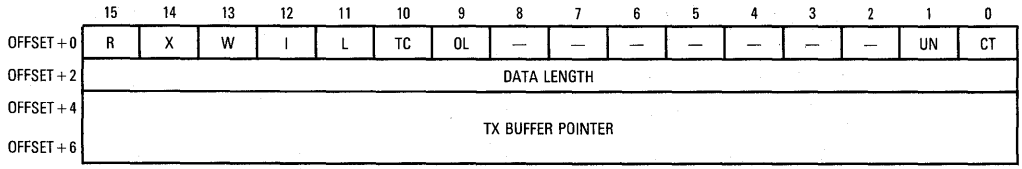


Figure 4-22. DDCMP Transmit Buffer Descriptor

The first word contains status and control bits. Bits 15–9 are prepared by the user before transmission. Bits 1–0 are set by the DDCMP controller after the buffer has been transmitted. Bit 15 is set by the user when the buffer and BD have been prepared and is cleared by the DDCMP controller when the message is transmitted.

R — Ready

0 = This buffer is not currently ready for transmission. The user is free to manipulate this BD (or its associated buffer). The DDCMP controller clears this bit after the buffer has been completely transmitted (or after an error condition is encountered).

1 = The data buffer has been prepared for transmission by the user (but not yet transmitted). No fields of this BD may be written by the user once this bit is set.

X — External Buffer

0 = The buffer associated with this BD is in internal dual-port RAM.

1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)

0 = This is not the last BD in the Tx BD table.

1 = This is the last BD in the Tx BD table. After this buffer has been used, the DDCMP controller will transmit data from the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt

0 = No interrupt is generated after this buffer has been serviced.

1 = The M68000 core will be interrupted when this buffer has been serviced by the DDCMP controller. (At least one of TX or TXE in the DDCMP event register will be set to cause this interrupt.)

L — Last

0 = This buffer is not the last in the message.

1 = The last bit is set by the processor to indicate that this buffer is the last buffer in the current message.

NOTE

The DDCMP controller checks the TC bit, not the last bit, to determine whether to append the CRC sequence. The DDCMP controller will transmit the programmable number of SYN1–SYN2 pairs before transmitting the next buffer (message) when the last bit is set.

TC — Tx CRC

0 = Do not transmit a CRC sequence after the buffer's last data byte.

1 = Transmit a CRC16 sequence after the buffer's last data byte .

When the last bit is not set but TC is set (e.g., in a header buffer), the DDCMP controller will append the next buffer immediately following the CRC sequence. The preset value for the CRC16 calculation is located in the PCRC register and should be initialized to all zeros or all ones.

OL — Optional Last

This bit allows the user to transmit abutted messages in DDCMP.

0 = Normal operation. The SYNF bit in the DDCMP mode register determines the pattern transmitted between messages.

1 = Abutted messages. The CP checks the ready bit of the next Tx BD after processing the current BD, and, if set, abutts the next message to the current message. If the ready bit is not set, the SYNF bit determines the transmitted pattern.

Bits 8–2 — Reserved for future use.

The following status bits are written by the DDCMP controller after it has finished transmitting the associated data buffer.

UN — Underrun

The DDCMP controller encountered a transmitter underrun condition while transmitting the associated data buffer.

NOTE

This error can occur only on synchronous links.

CT — CTS Lost
$\overline{\text{CTS}}$ in NMSI mode or grant in IDL/GCI mode was lost during message transmission.

Data Length
The data length is the number of octets that the DDCMP controller should transmit from this BD's data buffer. The data length should be greater than zero.

Tx Buffer Pointer
This pointer, which contains the address of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.14.12 DDCMP EVENT REGISTER. The SCC event register (SCCE) is referred to as the DDCMP event register when the SCC is configured for DDCMP. It is an 8-bit register used to report events recognized by the DDCMP channel. On recognition of an event, the DDCMP controller sets its corresponding bit in this register. Interrupts generated by this register may be masked in the DDCMP mask register.

The DDCMP event register is a memory-mapped register that may be read at any time. A bit is reset by writing a one (writing zero does not affect a bit's value). More than one bit may be reset at a time. All unmasked bits must be reset before the CP will clear the internal interrupt request. This register is cleared by reset.

7	6	5	4	3	2	1	0
CTS	CD	—	TXE	RBK	BSY	TX	RBD

CTS — Clear-To-Send Status Changed
A change in the status of the $\overline{\text{CTS}}$ line was detected on the DDCMP channel. The SCC status register may be read to determine the current status.

CD — Carrier Detect Status Changed
A change in the status of the $\overline{\text{CD}}$ line was detected on the DDCMP channel. The SCC status register may be read to determine the current status.

Bit 5 — Reserved for future use.

TXE — Tx Error
An error (CTS lost or underrun) occurred on the transmitter channel.

RBK — Receive Block
A complete block has been received on the DDCMP channel. A block is defined as reception of a complete header, a complete message, or a receiver error condition.

BSY — Busy Condition
A data byte was received and discarded due to lack of buffers. The receiver will enter hunt mode automatically.

TX — Tx Buffer
A buffer has been transmitted over the DDCMP channel.

RBD — Rx Buffer
A buffer has been received on the DDCMP channel that was not a complete block.

4.5.14.13 DDCMP MASK REGISTER. The SCC mask register (SCCM) is referred to as the DDCMP mask register when the SCC is operating as a DDCMP controller. It is an 8-bit read-write register that has the same bit format as the DDCMP event register. If a bit in the DDCMP mask register is a one, the corresponding interrupt in the event register will be enabled. If the bit is zero, the corresponding interrupt in the event register will be masked. This register is cleared upon reset.

4.5.15 V.110 Controller

The V.110 controller is discussed in the following paragraphs.

4.5.15.1 BIT RATE ADAPTION OF SYNCHRONOUS DATA SIGNALING RATES UP TO 19.2 kbps. The V.110 synchronous bit rate adaption block diagram within the terminal adaptor is shown in Figure 4-23.

This function may be implemented with two SCCs, one of which is configured for V.110 operation. Step 1 (RA1) rate adaption can be achieved using one SCC channel programmed to promiscuous (totally transparent) mode (see **4.5.13 BISYNC Controller**). This SCC will transfer the data between the R interface and IMP memory. The M68000 core must be programmed to format the data in memory according to the V.110 protocol to create the V.110 80-bit frame. Another SCC is used to transfer the data between IMP memory and the S/T interface. This SCC should be programmed for V.110 operation, which provides the conversion of the data rate to 64 kbps. Data may be

transmitted and received on 1, 2, or 4 bits of an ISDN B channel as programmed in the SIMASK register.

NOTE

V.110 contains a requirement (under further study) for control information on the R interface (i.e., RTS, CTS, CD, DTR, and DSR), conveyed by the S bits in the V.110 frame, not to have a different transmission delay than the user data conveyed by the D8–D1 bits. This very time-critical aspect of the standard is not supported by the IMP. In this case, provision would need to be made by the user to guarantee correct sampling times for this information to correspond with the user data. The IMP, however, can detect changes in these signals and issue appropriate interrupts to the M68000 core, allowing the function to be fully implemented in a slightly longer time period.

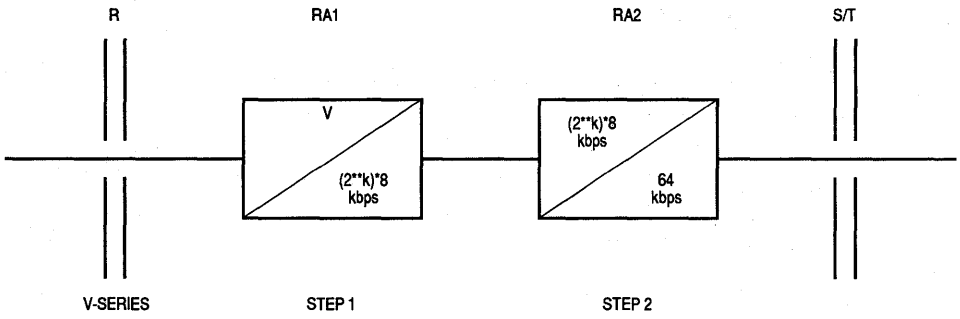


Figure 4-23. Two-Step Synchronous Bit Rate Adaption

4.5.15.2 RATE ADAPTION OF 48- AND 56-kbps USER RATES TO 64 kbps. This function may again be implemented with two SCCs; however, in this case, the SCC connected to the B channel is programmed to promiscuous (totally transparent) mode rather than for V.110 operation (see **4.5.9 SCC Transparent Support**). The M68000 core will need to format the framing pattern in the 48-kbps conversion case. For the 56-kbps rate conversion, however, the B channel mask (SIMASK) in the serial channel physical interface can be used.

4.5.15.3 ADAPTION FOR ASYNCHRONOUS RATES UP TO 19.2 kbps. The V.110 asynchronous bit rate adaption block diagram within the terminal adaptor is shown in Figure 4-24.

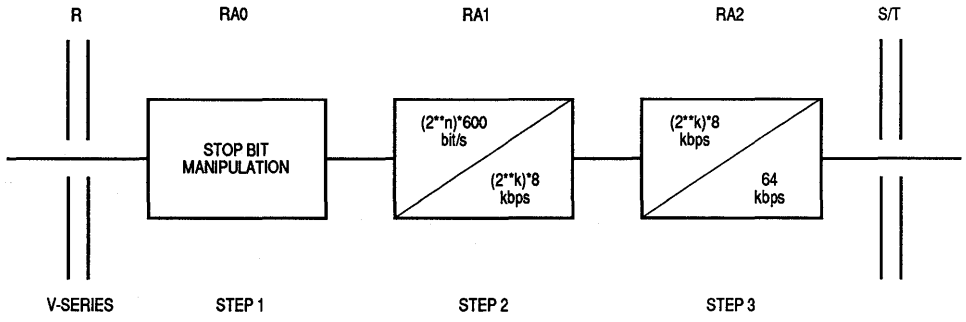


Figure 4-24. Three-Step Asynchronous Bit Rate Adaption

This function may be implemented in two SCCs. One SCC operates as a UART; the other SCC operates as a V.110 controller. The M68000 core formats the data for transmission by the V.110 at the 64 kbps data rate. Thus, the RA1 step is hidden in software.

4

4.5.15.4 V.110 CONTROLLER OVERVIEW. By the appropriate setting of its SCC mode register, any of the SCC channels may be configured to function as a V.110 controller. MODE1–MODE0 bits the SCC mode register should be programmed to DDCMP, and the V.110 bit in the DDCMP mode register should be set. The V.110 controller has the ability to receive and transmit V.110 80-bit frames. The processing of those frames is handled by the M68000 core in software.

The V.110 receiver will synchronize on the 17-bit alignment pattern of the frame:

00000000	1xxxxxxx	1xxxxxxx	1xxxxxxx	1xxxxxxx
1xxxxxxx	1xxxxxxx	1xxxxxxx	1xxxxxxx	1xxxxxxx

After achieving frame synchronization, the receiver will transfer the frame data to a receive buffer (the leading one will be the MSB). The V.110 controller will write nine bytes of data to the buffer (discarding the first byte of all zeros). The M68000 core should unformat the data in memory according to the V.110 protocol to create the data buffer; it may then use another SCC controller to transmit this data to the R interface.

The V.110 transmitter will transmit a data buffer transparently with a bit swap (the MSB will be transmitted first) onto a B channel. The data buffer should contain the 17-bit alignment pattern. Another SCC controller may be used to

receive data from the R interface. The M68000 core should then format the data according to the V.110 protocol to create the V.110 80-bit frame data buffer. The V.110 controller will then transmit it onto the B channel.

The V.110 controller operates on the ISDN physical interface using either IDL or GCI (IOM-2) over one of the B channels. NMSI and PCM physical interfaces are also possible. The data synchronization register (DSR) should be programmed to 'xxxxxxx1 00000000'b to achieve the proper frame synchronization (see **4.5.4 SCC Data Synchronization Register (DSR)**).

4.5.15.5 V.110 PROGRAMMING MODEL. The M68000 core configures each SCC to operate in one of the protocols by the MODE1–MODE0 bits in the SCC mode register. If MODE1–MODE0 = 10, the synchronous link DDCMP protocol is selected. The V.110 bit should also be set in the DDCMP mode register. The V.110 controller uses the same basic data structure as the DDCMP controller, the same command set, and the same event and mask registers for interrupt generation.

4.5.15.6 V.110 ERROR-HANDLING PROCEDURE. The V.110 controller reports frame reception and transmission error conditions using the channel buffer descriptors (BDs) and the V.110 event register.

Transmission Errors:

1. Transmitter Underrun. When this error occurs, the channel terminates buffer transmission, closes the buffer, sets the underrun (UN) bit in the BD, and generates the transmit error (TXE) interrupt (if enabled). The channel will resume transmission after the reception of the RESTART TRANSMIT command. The FIFO size is three bytes.

Reception Errors:

1. Overrun Error. The V.110 controller maintains an internal three-byte length FIFO for receiving data. When the receive FIFO overrun error occurs, the channel writes the received data byte to the internal FIFO on top of the previously received byte (the previous data byte is lost). Then the channel closes the buffer, sets the overflow (OV) bit in the BD, and generates the receive frame (RXF) interrupt (if enabled). The channel will automatically enter hunt mode.

2. Synchronization Error. A synchronization error is detected by the V.110 controller when the MSB of every byte is not one. When this error occurs, the channel writes the received byte to the buffer and continues to receive the V.110 frame. When the frame ends, the channel closes the buffer, sets the synchronization error (SE) bit in the BD, and generates the RXF interrupt (if enabled). The channel will automatically enter the hunt mode.

4.5.15.7 V.110 RECEIVE BUFFER DESCRIPTOR (Rx BD). The CP reports information about the received data for each buffer using the BDs. The Rx BD is shown in Figure 4-25. The CP closes the current buffer, generates a maskable interrupt, and starts to receive data in the next buffer after any of the following events:

Receiving of 10 bytes (80-bit frame)
Detecting of an error
Issuing the ENTER HUNT MODE command

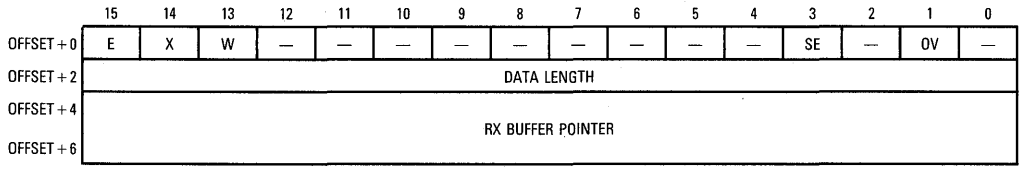


Figure 4-25. V.110 Receive Buffer Descriptor

The first word of the Rx BD contains control and status bits. Bits 15–13 are written by the user before the buffer is linked to the Rx BD table, and bits 1 and 3 are set by the IMP following message reception. Bit 15 is set by the M68000 core when the buffer is available to the V.110 controller and is cleared by the V.110 controller after filling the buffer.

E — Empty
- 0 = The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The M68000 core is free to examine or write to any fields of the BD.
- 1 = The data buffer associated with this BD is empty. This bit signifies that the BD and its associated buffer are available to the V.110 controller. The M68000 core should not write to any fields of this BD after it sets this bit. The empty bit will remain set while the V.110 controller is currently filling the buffer with received data.

X — External Buffer
0 = The buffer associated with this BD is in internal dual-port RAM.
1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)
0 = This is not the last BD in the Rx BD table.
1 = This is the last BD in the Rx BD table. After this buffer has been used, the V.110 controller receives incoming data by placing it in the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

Bits 12–4, 2, 0 — Reserved for future use.

SE — Synchronization Error
A frame with a synchronization error was received. A synchronization error is detected by the V.110 controller when the MSB of a byte (except the all-zeros byte) is not one.

OV — Overrun
A receiver overrun occurred during message reception.

Data Length
The data length is the number of octets that the V.110 controller has written to this BD data buffer. The V.110 controller will write nine bytes of data to the buffer. It will not write the all-zeros byte to the buffer.

NOTE

The actual buffer size should be greater than or equal to 10 bytes.

Rx Buffer Pointer
This pointer contains the address of the associated data buffer. The buffer may reside in either internal or external memory.

NOTE

The Rx buffer pointer must be even.

4.5.15.8 V.110 TRANSMIT BUFFER DESCRIPTOR (Tx BD). Data is presented to the CP for transmission on an SCC channel by arranging it in buffers referenced

by the channel's Tx BD table. The CP confirms transmission (or indicates error conditions) using the BDs to inform the M68000 core whether the buffers have been serviced. The Tx BD is shown in Figure 4-26.

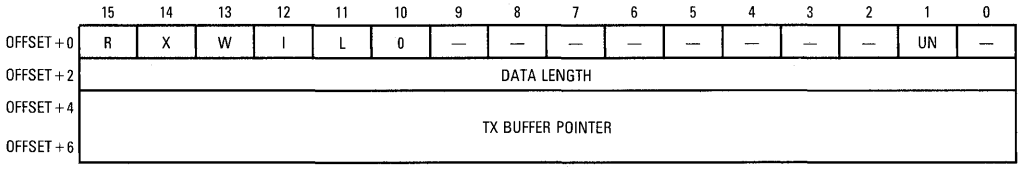


Figure 4-26. V.110 Transmit Buffer Descriptor

The first word contains status and control bits. Bits 15–10 are prepared by the user before transmission. Bit 1 is set by the V.110 controller after the buffer has been transmitted. Bit 15 is set by the user and cleared by the V.110 controller.

R — Ready
- 0 = This buffer is not currently ready for transmission. The user is free to manipulate this BD and its associated buffer. The V.110 controller clears this bit after the buffer has been fully transmitted (or after an error condition is encountered).
- 1 = The data buffer has been prepared for transmission by the user (but not yet transmitted). No fields of this BD may be written by the user once this bit is set.

X — External Buffer
- 0 = The buffer associated with this BD is in internal dual-port RAM.
- 1 = The buffer associated with this BD is in external memory.

W — Wrap (Final BD in Table)
- 0 = This is not the last BD in the Tx BD table.
- 1 = This is the last BD in the Tx BD table. After this buffer has been used, the V.110 controller will transmit data from the first BD in the table.

NOTE

The user is required to set the wrap bit in one of the first eight BDs; otherwise, errant behavior may occur.

I — Interrupt
- 0 = No interrupt is generated after this buffer has been serviced.
- 1 = The M68000 core processor will be interrupted when this buffer has

been serviced by the V.110 controller. (At least one of TX or TXE in the V.110 event register will be set to cause this interrupt.)

L — Last
0 = This buffer is not the last in the message.
1 = This bit is set by the processor to indicate that this buffer is the last buffer in the current frame. The V.110 controller will transmit ones until the next BD is ready.

Bit 10 — Must be set to zero by the user.

Bits 9–2, 0 — Reserved for future use.

The following bits are written by the V.110 controller after it has finished transmitting the associated data buffer.

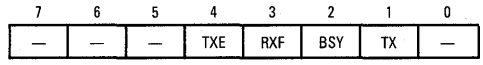
UN — Underrun
The V.110 controller encountered a transmitter underrun condition while transmitting the associated data buffer.

Data Length
The data length is the number of octets that the V.110 controller should transmit from this BD's data buffer. The data length should be greater than zero.

Tx Buffer Pointer
This pointer, which contains the address of the associated data buffer, may be even or odd. The buffer may reside in either internal or external memory.

4.5.15.9 V.110 EVENT REGISTER. The SCC event register (SCCE) is referred to as the V.110 event register when the SCC is configured as a V.110 controller. It is an 8-bit register used to report events recognized by the V.110 channel. On recognition of an event, the V.110 controller sets its corresponding bit in this register. Interrupts generated by this register may be masked in the V.110E mask register.

The V.110 event register is a memory-mapped register that may be read at any time. A bit is reset by writing a one (writing a zero does not affect a bit's value). More than one bit may be reset at a time. All unmasked bits must be reset before the CP will clear the internal interrupt request signal. This register is cleared by reset.



Bits 7–5, 0 — Reserved for future use.

TXE — Tx Error
An error (underrun) occurred on the transmitter channel.

RXF — Receive Frame
A complete frame has been received on the V.110 channel.

BSY — Busy Condition
A data byte was received and discarded due to lack of buffers. The receiver will automatically enter hunt mode.

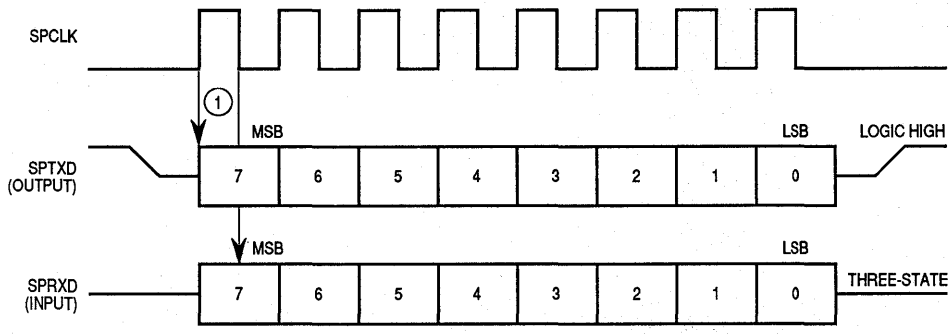
TX — Tx Buffer
A buffer has been transmitted.

4.5.15.10 V.110 MASK REGISTER. The SCC mask register (SCCM) is referred to as the V.110 mask register when the SCC is operating as a V.110 controller. It is an 8-bit read-write register that has the same bit format as the V.110 event register. If a bit in the V.110 mask register is a one, the corresponding interrupt in the event register will be enabled. If the bit is zero, the corresponding interrupt in the event register will be masked. This register is cleared upon reset.

4.6 SERIAL COMMUNICATION PORT (SCP)

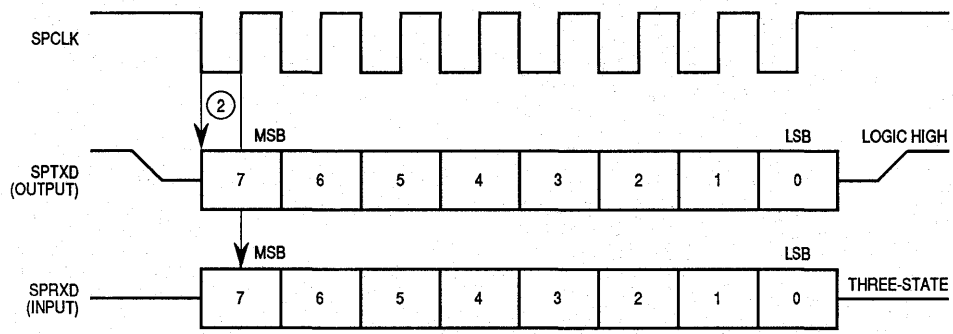
The SCP (see Figure 4-27) is a full-duplex, synchronous, character-oriented channel that provides a three-wire interface (receive, transmit, and clock). The SCP consists of independent transmitter and receiver sections and a common clock generator. The transmitter and receiver sections use the same clock, which is derived from the main clock by a separate on-chip baud rate generator. Since the MC68302 is an SCP master for this serial channel, it generates both the enable and the clock signals.

The SCP allows the MC68302 to exchange status and control information with a variety of serial devices, using a subset of the Motorola serial peripheral interface (SPI). These devices include industry-standard CODECs as well as other microcontrollers and peripherals.



NOTE: Transmitted data bits shift on rising edges; received bits are sampled on falling edges.

(a) CI = 0



NOTE: Transmitted data bits shift on falling edges; received bits are sampled on rising edges.

(b) CI = 1

Figure 4-27. SCP Timing

The SCP enable signals, which can be implemented using the general-purpose I/O pins, are used to enable one of several potential SCP slave devices. The clock signal (SPCLK) shifts the received data (SPRXD) in and shifts the transmitted data (SPTXD) out. The clock is gated; it operates only while data is being transferred and is idle otherwise.

The SCP can be configured to operate in a local loopback mode, which is useful for local diagnostic functions.

Note that the least significant bit of the SCP is labeled as data bit 0 on the serial line; whereas, other devices, such as the MC145554 CODEC, may label the most significant bit as data bit 0. The MC68302 SCP bit 7 (most significant bit) is shifted out first.

The SCP key features are as follows:

- Three-Wire Interface (SPTXD, SPRXD, and SPCLK)
- Full-Duplex Operation
- Clock Rate up to 4.096 MHz
- Programmable Clock Generator
- Local Loopback Capability for Testing

4.6.1 SCP Programming Model

The SCP mode register consists of the upper eight bits of SPMODE. The SCP mode register, an internal read-write register that controls both the SCP operation mode and clock source, is cleared by reset.

15	14	13	12	11	10	9	8
STR	LOOP	CI	PM3	PM2	PM1	PM0	EN

STR — Start Transmit

When set, this bit causes the SCP controller to transmit eight bits from the SCP transmit/receive buffer descriptor (BD), and to receive eight bits of data in this same BD. This bit is cleared automatically after one system clock cycle.

LOOP — Loop Mode

When set, the loop mode bit selects local loopback operation. The ones complement of the transmitter output is internally connected to the receiver input; the receiver and transmitter operate normally except that SPRXD is ignored. When cleared, this bit selects normal operation.

CI — Clock Invert

When set, the CI bit inverts the SCP clock polarity. When CI is zero, transmitted data bits shift on rising clock edges, and received bits are sampled on falling edges. When the SCP is idle, the clock is low. While CI is one, transmitted data bits are shifted on falling edges, and received bits are sampled on rising edges. In this case, when the SCP is idle, the clock is high.

PM3–PM0 — Prescale Modulus Select

The prescale modulus select bits specify the divide ratio of the prescale divider in the SCP clock generator. The divider value is 4*("PM3–PM0" + 1) giving a clock divide ratio of 4 to 64 in multiples of 4. With a 16.384-MHz system clock, the maximum SCP clock is 4.096 MHz.

EN — Enable SCP

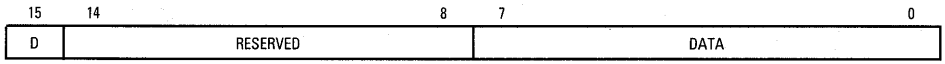
When set, this bit enables the SCP operation and connects the external pins SPRXD/$\overline{\text{CTS3}}$, SPTXD/$\overline{\text{RTS3}}$, and SPCLK/$\overline{\text{CD3}}$ internally to the SCP. When cleared, the SCP is put into a reset state consuming minimal power, and the three pins are connected back to SCC3.

NOTE

When SCC3 is programmed to allow automatic control of the $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$ lines, the user may not modify the EN bit.

4.6.2 SCP Transmit/Receive Buffer Descriptor

The transmit/receive BD contains the data to be transmitted (written by the M68000 core) and the received data (written by the SCP). The done (D) bit indicates that the received data is valid and is cleared by the SCP.



4.6.3 SCP Transmit/Receive Processing

The IMP SCP always functions in the master mode. Thus, in a typical exchange of messages, the IMP transmits a message to an external peripheral (SCP slave) which, in turn, sends back a reply. When the IMP works with more than one slave, it can use the general-purpose parallel I/O pins as enable (select) signals. To begin the data exchange, the M68000 core writes the data to be transmitted into the transmit/receive BD, setting the done bit. The M68000 core should then set the start transmit (STR) bit in the SPMODE register to start transmission of data. STR is cleared by hardware after one system clock cycle.

Upon recognizing the STR bit, the SCP also begins receiving eight bits of data. It writes the data into the transmit/receive BD, clears the done bit, and issues a maskable interrupt to the IMP interrupt controller. When working in a polled environment, the done bit should be set by the M68000 core before

setting the STR bit so that received replies may be easily recognized by the software.

4.7 SERIAL MANAGEMENT CONTROLLERS (SMCs)

The SMC key features are as follows:

- Two Modes of Operation:
  IDL — SMC1 supports the maintenance channel and SMC2 supports the auxiliary channel
  GCI (IOM-2) — SMC1 supports the monitor channel and SMC2 supports the C/I channel
- Full-Duplex Operation
- Local Loopback Capability for Testing

4.7.1 Overview

The SMCs are two synchronous, full-duplex serial management control (SMC) ports. The SMC ports may be configured to operate in either Motorola interchip digital link (IDL) or general circuit interface (GCI) modes. GCI is also known as ISDN oriented modular 2 (IOM-2). See **4.4 SERIAL CHANNELS PHYSICAL INTERFACE** for the details of configuring the IDL and GCI interfaces. The SMC ports are not used when the physical serial interface is configured for PCM highway or NMSI modes.

4.7.1.1 USING IDL WITH THE SMCs. In this mode, SMC1 transfers the maintenance (M) bits of the IDL to and from the internal RAM, and SMC2 transfers the auxiliary (A) bits to and from the internal RAM. The CP generates a maskable interrupt upon reception/transmission of eight bits. The SMC1 and SMC2 receivers can be programmed to work in hunt-on-zero mode, in which the receiver will search the line signals for a zero bit. When it is found, the receiver will transfer data to the internal RAM.

4.7.1.2 USING GCI WITH THE SMCs. In this mode, SMC1 controls the GCI monitor channel.

SMC1 Transmission

The monitor channel is used to transfer commands to the layer-1 component. The M68000 core writes the data byte into the SMC1 Tx BD. SMC1 will transmit the data on the monitor channel.

The SMC1 channel transmitter can be programmed to work in one of two modes:

Transparent Mode

In this mode, SMC1 transmits the monitor channel data and the A and E control bits transparently into the channel. When the M68000 core has not written new data to the buffer, the SMC1 transmitter will retransmit the previous monitor channel data and the A and E control bits.

Monitor Channel Protocol

In this mode, SMC1 transmits the data and handles the A and E control bits according to the GCI monitor channel protocol. When using the monitor channel protocol, the user may issue the TIMEOUT command to solve deadlocks in case of bit errors in the A and E bit positions on data line. The IMP will transmit an abort on the E bit.

SMC1 Reception

The SMC1 receiver can be programmed to work in one of two modes:

Transparent Mode

In this mode, SMC1 receives the data, moves the A and E control bits transparently into the SMC1 receive BD, and generates a maskable interrupt. The SMC1 receiver discards new data when the M68000 core has not read the receive BD.

Monitor Channel Protocol

In this mode, SMC1 receives data and handles the A and E control bits according to the GCI monitor channel protocol. When a received data byte is stored by the CP in the SMC1 receive BD, a maskable interrupt is generated.

When using the monitor channel protocol, the user may issue the TRANSMIT ABORT REQUEST command. The IMP will then transmit an abort request on the A bit.

SMC2 Controls the GCI Command/Indication (C/I) Channel

SMC2 Transmission

The M68000 core writes the data byte into the SMC2 Tx BD. SMC2 will transmit the data continuously on the C/I channel to the physical layer device.

SMC2 Reception

The SMC2 receiver continuously monitors the C/I channel. When a change in data is recognized and this value is received in two successive frames, it will be interpreted as valid data. The received data byte is stored by the CP in the SMC2 receive BD, and a maskable interrupt is generated.

The receive and transmit clocks are derived from the same physical clock (L1CLK) and are only active while serial data is transferred between the SMC controllers and the serial interface.

When SMC loopback mode is chosen, SMC transmitted data is routed to the SMC receiver. Transmitted data appears on the L1TXD pin, unless the SDIAG1–SDIAG0 bits in the SIMODE register are programmed to "loopback control" (see **4.4 SERIAL CHANNELS PHYSICAL INTERFACE**).

4.7.2 SMC Programming Model

The operating mode of both SMC ports is defined by SMC mode, which consists of the lower eight bits of SPMODE. As previously mentioned, the upper eight bits program the SCP.

7	6	5	4	3	2	1	0
—	—	SMD2	SMD1	SMD0	LOOP	EN2	EN1

Bits 7–6 — These bits are reserved and should be set to zero.

SMD2–SMD0 — SMC Mode Support

000 = GCI — The monitor channel is not used.
001 = GCI — The monitor channel data and the A and E control bits are received and transmitted transparently by the IMP.
01X = GCI — The monitor channel data and the A and E control bits are internally controlled according to the monitor channel protocol.
100 = IDL — The M and A channels are in hunt-on-zero mode.
101 = IDL — Only the M channel is in hunt-on-zero mode.
110 = IDL — Only the A channel is in hunt-on-zero mode.
111 = IDL — Regular operation; no channel is in hunt-on-zero mode.

LOOP — Local Loopback Mode

0 = Normal mode
1 = Local loopback mode

EN2 — SMC2 Enable
0 = Disable SMC2
1 = Enable SMC2

EN1 — SMC1 Enable
0 = Disable SMC1
1 = Enable SMC1

4.7.3 SMC Commands

The following commands issued to the CP command register (see **4.3 COMMAND SET)** are used only when GCI is selected for the serial channels physical interface.

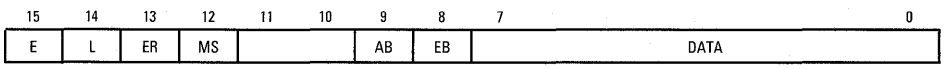
TRANSMIT ABORT REQUEST Command
This receiver command may be issued when the IMP implements the monitor channel protocol. When issued, the IMP sends an abort request on the A bit.

TIMEOUT Command
This transmitter command may be issued when the IMP implements the monitor channel protocol. It is issued because the device is not responding or because GCI A bit errors are detected. When issued, the IMP sends an abort request on the E bit.

4.7.4 SMC Memory Structure and Buffers Descriptors

The CP uses several memory structures and memory-mapped registers to communicate with the M68000 core. All the structures detailed in the following paragraphs reside in the dual-port RAM of the IMP (see Figure 3-4). The SMC buffer descriptors allow the user to define one data byte at a time for each transmit channel and receive one data byte at a time for each receive channel.

4.7.4.1 SMC1 RECEIVE BUFFER DESCRIPTOR. The CP reports information about the received byte using this (BD).



E — Empty

0 = This bit is cleared by the CP to indicate that the data byte associated with this BD is now available to the M68000 core.

1 = This bit is set by the M68000 core to indicate that the data byte associated with this BD is empty.

In GCI mode, when the IMP implements the monitor channel protocol, the IMP will wait until this bit is set by the M68000 core before acknowledging the monitor channel data. In other modes (transparent GCI and IDL), additional received data bytes will be discarded until the empty bit is set by the M68000 core.

L — Last (EOM)

This bit is valid only in GCI mode when the IMP implements the monitor channel protocol. This bit is set when the end-of-message (EOM) indication is received on the E bit.

NOTE

When this bit is set, the data byte is not valid.

ER — Error Condition

This bit is valid only in GCI mode when the IMP implements the monitor channel protocol and the L bit is set. This bit is set when an error condition occurs on the monitor channel protocol. A new byte is transmitted before the IMP acknowledges the previous byte.

MS — Data Mismatch

This bit is valid only in GCI mode when the IMP implements the monitor channel protocol. This bit is set when two different consecutive bytes are received and is cleared when the last two consecutive bytes match. The IMP waits for the reception of two identical consecutive bytes before writing new data to the receive BD.

Bits 11–10 — Reserved for future use.

AB — Received A Bit

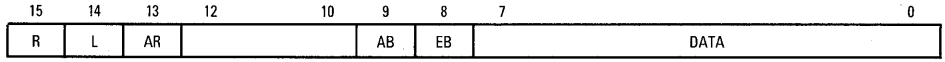
This bit is valid only in GCI mode when the the monitor channel is in transparent mode.

EB — Received E Bit

This bit is valid only in GCI mode when the the monitor channel is in transparent mode.

Data — Data Field
The data field contains the byte of data received by SMC1.

4.7.4.2 SMC1 TRANSMIT BUFFER DESCRIPTOR. The CP reports information about this transmit byte through the BD.



R — Ready
0 = This bit is cleared by the CP after transmission. The Tx BD is now available to the M68000 core.
1 = This bit is set by the M68000 core to indicate that the data byte associated with this BD is ready for transmission.

In GCI mode, when the IMP implements the monitor channel protocol, it will clear this bit after receiving an acknowledgement on the A bit. When the SMC1 data should be transmitted and this bit is cleared, the channel will retransmit the previous data until new data is provided by the M68000 core.

L — Last (EOM)
This bit is valid only in GCI mode when the IMP implements the monitor channel protocol. When this bit is set, the SMC1 channel will transmit the buffer's data and then the end of message (EOM) indication on the E bit.

AR — Abort Request
This bit is valid only in GCI mode when the IMP implements the monitor channel protocol. This bit is set by the IMP when an abort request was received on the A bit. The SMC1 transmitter will transmit EOM on the E bit.

Bits 12–10 — Reserved for future use.

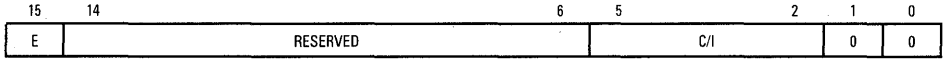
AB — Transmit A Bit Value
This bit is valid only in GCI mode when the the monitor channel is in transparent mode.

EB — Transmit E Bit Value
This bit is valid only in GCI mode when the the monitor channel is in transparent mode.

Data — Data Field

The data field contains the data to be transmitted by SMC1.

4.7.4.3 SMC2 RECEIVE BUFFER DESCRIPTOR. In the IDL mode, this BD is identical to the SMC1 receive BD. In the GCI mode, SMC2 is used to control the C/I channel.



E — Empty

0 = This bit is cleared by the CP to indicate that the data bits associated with this BD are now available to the M68000 core.

1 = This bit is set by the M68000 core to indicate that the data bits associated with this BD have been read.

4

NOTE

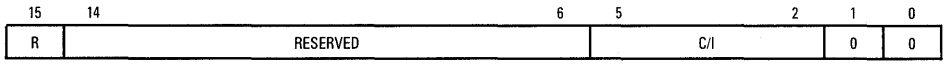
Additional data received will be discarded until the empty bit is set by the M68000 core.

Bits 14–6 — These bits are reserved and should be set to zero by the M68000 core.

C/I — Command/Indication Channel Data

Bits 1–0 — The CP always writes these bits with zeros.

4.7.4.4 SMC2 TRANSMIT BUFFER DESCRIPTOR. In the IDL mode, this BD is identical to the SMC1 transmit BD. In the GCI mode, SMC2 is used to control the C/I channel.



R — Ready

0 = This bit is cleared by the CP after transmission to indicate that the BD is now available to the M68000 core.

1 = This bit is set by the M68000 core to indicate that the data associated with this BD is ready for transmission.

Bits 14–6 — Reserved for future use; should be set to zero by the user.

C/I — Command/Indication Channel Data

Bits 1–0 — These bits should be written with zeros by the M68000 core.

4.7.5 SMC Interrupt Requests

SMC1 and SMC2 send individual interrupt requests to the IMP interrupt controller when one of the respective SMC receive buffers is full or when one of the SMC transmit buffers is empty. Each of the two interrupt requests from each SMC is enabled when its respective SMC channel is enabled in the SPMODE register. Interrupt requests from SMC1 and SMC2 can be masked in the interrupt mask register. See **3.2 INTERRUPT CONTROLLER** for more details.

4

SECTION 5
SIGNAL DESCRIPTION

This section defines the MC68302 pinout. The input and output signals of the MC68302 are organized into functional groups and are described in the following sections. The MC68302 is offered in a 132 pin (13 × 13) pin grid array (PGA), a 132 lead ceramic quad flat package (CQFP), and a 132 lead plastic quad flat package (PQFP).

The MC68302 uses a standard M68000 bus for communication between both on-chip and external peripherals. This bus is a single, continuous bus existing both on and off-chip of the MC68302. Any access made internal to the device is visible externally. Any access made external is visible internally. Thus, when the M68000 core accesses the dual-port RAM, the bus signals are driven externally. Likewise, when an external device accesses an area of external system memory, the chip-select logic can be used to generate the chip-select signal and $\overline{\text{DTACK}}$.

5.1 FUNCTIONAL GROUPS

The input and output signals of the MC68302 are organized into functional groups as shown in Table 5-1 and Figure 5-1.

Table 5-1. Signal Definitions (Sheet 1 of 2)

Functional Group	Signals	Num.
Clocks	XTAL, EXTAL, CLKO	3
System Control	$\overline{\text{RESET}}$, $\overline{\text{HALT}}$, $\overline{\text{BERR}}$, BUSW, DISCPU	5
Address Bus	A23–A1	23
Data Bus	D15–D0	16
Bus Control	$\overline{\text{AS}}$, R/$\overline{\text{W}}$, $\overline{\text{UDS}}$/A0, $\overline{\text{LDS}}$/$\overline{\text{DS}}$, $\overline{\text{DTACK}}$	5
Bus Control	$\overline{\text{RMC}}$, IAC, $\overline{\text{BCLR}}$	3
Bus Arbitration	$\overline{\text{BR}}$, $\overline{\text{BG}}$, $\overline{\text{BGACK}}$	3
Interrupt Control	$\overline{\text{IPL2}}$–$\overline{\text{IPL0}}$, FC2–FC0, $\overline{\text{AVEC}}$	7
NMSI1/ISDN I/F	RXD, TXD, RCLK, TCLK, $\overline{\text{CD}}$, $\overline{\text{CTS}}$, $\overline{\text{RTS}}$, BRG1	8
NMSI2/PAIO	RXD, TXD, RCLK, TCLK, $\overline{\text{CD}}$, $\overline{\text{CTS}}$, $\overline{\text{RTS}}$, SDS2	8

Table 5-1. Signal Definitions (Sheet 2 of 2)

Functional Group	Signals	Num.
NMSI3/SCP/PAIO	RXD, TXD, RCLK, TCLK, $\overline{CD}$, $\overline{CTS}$, $\overline{RTS}$, PA12	8
IDMA/PAIO	$\overline{DREQ}$, $\overline{DACK}$, $\overline{DONE}$	3
IACK/PBIO	$\overline{IACK7}$, $\overline{IACK6}$, $\overline{IACK1}$	3
Timer/PBIO	TIN2, TIN1, $\overline{TOUT2}$, $\overline{TOUT1}$, $\overline{WDOG}$	5
PBIO	PB11–PB8	4
Chip Select	$\overline{CS3}$–$\overline{CS0}$	4
Testing	$\overline{FRZ}$ (2 Spare)	3
V_{DD}		8
GND		13

All pins except EXTAL, CLKO, and the layer 1 interface pins in IDL mode support TTL levels. EXTAL, when used as an input clock, needs a CMOS level. CLKO supplies a CMOS level output. The IDL interface is specified as a CMOS electrical interface.

All outputs (except CLKO and the GCI pins) drive 130 pF. CLKO is designed to drive 50 pF. The GCI output pins drive 150 pF.

5

5.2 POWER PINS

The IMP has 21 power supply pins. Careful attention has been paid to reducing IMP noise, potential cross-talk, and RF radiation from the output drivers. Inputs may be +5 V when V_{DD} is 0 V without damaging the device.

V_{DD} (8) — There are 8 power pins.

GND (13) — There are 13 ground pins.

5.3 CLOCKS

EXTAL — External Clock/Crystal Input

This input provides two clock generation options. EXTAL may be used (with XTAL) to connect an external crystal to the on-chip oscillator and clock generator. If an external clock is used, the clock source should be connected to EXTAL, and XTAL should be left unconnected. The oscillator uses an internal frequency equal to the external crystal frequency. The frequency of EXTAL may range from 8 MHz to 16.67 MHz. When an external clock is used, it must provide a CMOS level at this input frequency.

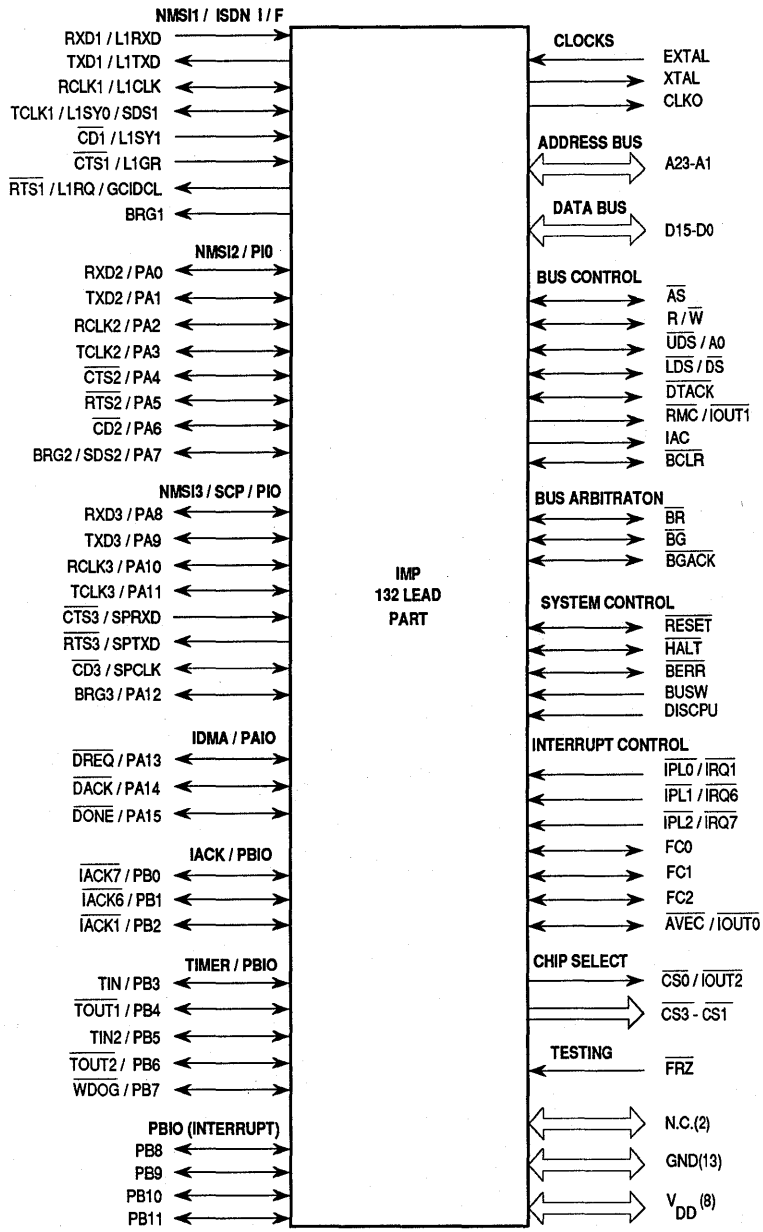


Figure 5-1. Functional Signal Groups

XTAL — Crystal Output

This output connects the on-chip oscillator output to an external crystal. If an external clock is used, XTAL should be left unconnected. See **3.7 ON-CHIP CLOCK GENERATOR** for more details.

CLKO — Clock Out

This output clock signal is derived from the on-chip clock oscillator. This clock signal is internally connected to the clock input of the M68000 core, the communication processor and system integration block. All M68000 bus timings are referenced to the CLKO signal. CLKO supports both CMOS and TTL output levels.

5.4 SYSTEM CONTROL

$\overline{\text{RESET}}$

This bidirectional, open-drain signal line, acting as an input and asserted along with the $\overline{\text{HALT}}$ pin, starts an initialization sequence called a total system reset that resets the entire MC68302. $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ should remain asserted for at least 100 ms at poweron reset, and at least 10 clocks otherwise. The on-chip system RAM is not initialized during reset except for several locations initialized by the CP.

An internally generated reset, from the M68000 RESET instruction, causes the $\overline{\text{RESET}}$ line to become an output. In this case, the M68000 core is not reset; however, the communication processor is fully reset, and the system integration block is almost fully reset (refer to Tables 2-6 and 2-9 for a list of the unaffected registers). The user may also use the $\overline{\text{RESET}}$ signal in this case to reset all external devices.

During reset, the address, data, and bus control pins are all three-stated, except for $\overline{\text{CS3}}$–$\overline{\text{CS0}}$, which are high, and IAC, which is low. The $\overline{\text{BG}}$ pin output is the same as that on the $\overline{\text{BR}}$ input. The general-purpose I/O pins are configured as inputs, except for $\overline{\text{WDOG}}$, which is an open-drain output. The NMSI1 pins are all inputs, except for $\overline{\text{RTS1}}$ and TXD1, which output a high value. $\overline{\text{RTS3}}$, NC1, and NC3 are also high. BRG1 is CLKO/3.

$\overline{\text{HALT}}$ — Halt

When this bidirectional, open-drain signal is driven by an external device, it will cause the IMP bus master (M68000 core, SDMA or IDMA) to stop at the completion of the current bus cycle. If the processor has stopped executing instructions due to a double-fault condition, this line is driven by the processor to indicate to external devices that the processor has stopped.

This signal is asserted with the $\overline{\text{RESET}}$ signal to cause a total MC68302 system reset.

$\overline{\text{BERR}}$ — Bus Error

This bidirectional, open-drain signal informs the bus master (M68000 core, SDMA, IDMA, or external bus master) that there is a problem with the cycle currently being executed. This signal can be asserted by the on-chip hardware watchdog (bus timeout because of no $\overline{\text{DTACK}}$), by the chip-select logic (address conflict or write-protect violation) or by external circuitry.

BUSW — Bus Width Select

This input defines the M68000 processor mode (MC68000 or MC68008) and the data bus width (16 bits or 8 bits, respectively). BUSW may only be changed upon a total system reset.

Low = 8-bit data bus, MC68008 core processor
High = 16-bit data bus, MC68000 core processor

DISCPU — Disable CPU (M68000 core)

The MC68302 can be configured to work solely with an external CPU. In this mode the on-chip M68000 core CPU should be disabled by asserting the DISCPU pin high during a total system reset ($\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ asserted). DISCPU may only be changed upon a total system reset.

The DISCPU pin, for instance, allows use of several IMPs to provide more than three SCC channels without the need for bus isolation techniques. Only one of the IMP M68000 cores is active, and services the other IMPs as peripherals (with their respective cores disabled). Refer to **3.8.4 Disable CPU Logic (M68000)** for more details.

$\overline{\text{FRZ}}$ — Freeze Activity

The $\overline{\text{FRZ}}$ pin is used to freeze the activity of selected peripherals. This is useful for system debugging purposes. Refer to **3.8 SYSTEM CONTROL** for more details. $\overline{\text{FRZ}}$ should be negated during total system reset.

5.5 ADDRESS BUS PINS A23–A1

A23–A1 form a 24-bit address bus when combined with $\overline{\text{UDS}}$/A0. The address bus is a bidirectional, three-state bus capable of addressing 16M bytes of data (including the IMP internal address space). It provides the address for bus operation during all cycles except CPU space cycles. In CPU space cycles, the CPU reads a peripheral device vector number or indicates a breakpoint instruction.

These lines are outputs when the IMP (M68000 core, SDMA or IDMA) is the bus master and are inputs otherwise.

5.6 DATA BUS PINS D15–D0

This 16-bit, bidirectional, three-state bus is the general-purpose data path. It can transmit and accept data in either word or byte lengths. For all 16-bit IMP accesses, byte 0, the high-order byte of a word, is available on D15–D8, conforming to the standard M68000 format.

When working with an 8-bit bus (BUSW is low), the data is transferred through the low-order byte (7–0). The high-order byte is not used.

5.7 BUS CONTROL PINS

$\overline{AS}$ — Address Strobe

This bidirectional signal indicates that there is a valid address on the address bus. This line is an output when the IMP (M68000 core, SDMA or IDMA) is the bus master, and is an input otherwise.

R/$\overline{W}$ — Read/Write

This bidirectional signal defines the data bus transfer as a read or write cycle. It is an output when the IMP is the bus master and is an input otherwise.

$\overline{UDS}$/A0 — Upper Data Strobe/Address 0

This bidirectional line controls the flow of data on the data bus. When using a 16-bit data bus, this pin functions as upper data strobe ($\overline{UDS}$). When using an 8-bit data bus, this pin functions as A0.

This line is an output when the IMP is the bus master, and is an input otherwise.

$\overline{LDS}$/$\overline{DS}$ — Lower Data Strobe/Data Strobe

This bidirectional line controls the flow of data on the data bus. When using a 16-bit data bus, this pin functions as lower data strobe ($\overline{LDS}$). When using an 8-bit data bus, this pin functions as $\overline{DS}$. This line is an output when the IMP (M68000 core, SDMA or IDMA) is the bus master and is an input otherwise.

$\overline{\text{DTACK}}$ — Data Transfer Acknowledge.

This bidirectional signal indicates that the data transfer has been completed. $\overline{\text{DTACK}}$ can be generated internally in the chip-select logic either for an IMP bus master or for an external bus master access to an external address within the chip-select ranges. It will also be generated internally during any access to the on-chip dual-port RAM or internal registers. If $\overline{\text{DTACK}}$ is generated internally, then it is an output. It is an input when the IMP accesses an external device not within the range of the chip-select logic, or when programmed to be generated externally.

$\overline{\text{RMC}}$/$\overline{\text{IOUT1}}$ — Read-Modify-Write Cycle Indication/Interrupt Output 1

This signal functions as $\overline{\text{RMC}}$ in normal operation. $\overline{\text{RMC}}$ is an output signal that is asserted when a read-modify-write cycle is executed. It indicates that the cycle is indivisible.

When the M68000 core is disabled, this pin operates as $\overline{\text{IOUT1}}$. $\overline{\text{IOUT2}}$–$\overline{\text{IOUT0}}$ provide the interrupt request output signals from the IMP interrupt controller to an external CPU when the M68000 core is disabled.

IAC — Internal Access

This output indicates that the current bus cycle accesses an on-chip location. This includes the on-chip 4K byte block of internal RAM and registers (both real and reserved locations), and the system configuration registers ($0F0–$0FF). The above-mentioned bus cycle may originate from the M68000 core, the IDMA, or an external bus master. Note that, if the SDMA accesses the internal dual-port RAM, it does so without arbitration on the M68000 bus; therefore, the IAC pin is not asserted in this case. The timing of IAC is identical to that of the $\overline{\text{CS3}}$–$\overline{\text{CS0}}$ pins.

IAC can be used to disable an external address/data buffer when the on-chip dual-port RAM and registers are accessed, thus preventing bus contention. Such a buffer is optional and is only required in larger systems. An external address/data buffer with its output enable ($\overline{\text{E}}$) and direction control (dir) may be placed between the two bus segments as shown in Figure 5-2. The IAC signal saves the propagation delay and logic required to OR all the various system chip-select lines together, to determine when to enable the external buffers.

$\overline{\text{BCLR}}$ — Bus Clear

This open-drain output indicates that the M68000 core or the serial DMA (SDMA) requests the external bus master to release the bus. The core may be configured to assert this signal when it has a pending interrupt to

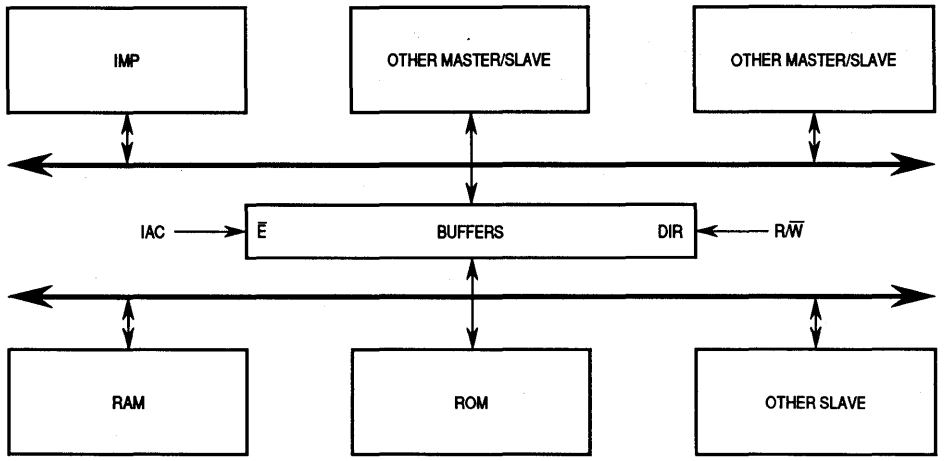


Figure 5-2. External Address/Data Buffer

execute. The SDMA asserts this signal when one of the SCCs is requesting DMA service.

When the M68000 core is disabled, this signal is an input to the independent DMA (IDMA), and is interpreted as a bus release request. It remains an output from the SDMA in this mode.

5.8 BUS ARBITRATION PINS

$\overline{BR}$ — Bus Request

This input signal indicates to the on-chip bus arbiter that an external device desires to become the bus master. This signal is an open-drain output request signal from the IDMA and SDMA when the internal M68000 core is disabled.

$\overline{BG}$ — Bus Grant

This output signal indicates to all external bus master devices that the processor will release bus control at the end of the current bus cycle to an external bus master. This signal is an input to the IDMA and SDMA when the internal M68000 core is disabled.

$\overline{BGACK}$ — Bus Grant Acknowledge

This bidirectional signal indicates that some other device besides the M68000 core has become the bus master. This signal is an input when an external device or the M68000 core owns the bus. This signal is an output when

the IDMA or SDMA has become the master of the bus. If the SDMA steals a cycle from the IDMA, the $\overline{\text{BGACK}}$ pin will remain asserted continuously.

5.9 INTERRUPT CONTROL PINS

These inputs have dual functionality:

$\overline{\text{IPL0}}/\overline{\text{IRQ1}}$
$\overline{\text{IPL1}}/\overline{\text{IRQ6}}$
$\overline{\text{IPL2}}/\overline{\text{IRQ7}}$ — Interrupt Priority Level 2–0/Interrupt Request 1,6,7

These inputs have dual functionality:

As $\overline{\text{IPL2}}$–$\overline{\text{IPL0}}$ (normal mode), these input pins indicate the encoded priority level of the external device requesting an interrupt. Level 7 is the highest (nonmaskable) priority; whereas, level 0 indicates that no interrupt is requested. The least significant bit is $\overline{\text{IPL0}}$, and the most significant bit is $\overline{\text{IPL2}}$. These lines must remain stable until the M68000 core signals an interrupt acknowledge through FC2–FC0 and A19–A16, to ensure that the interrupt is properly recognized.

As $\overline{\text{IRQ1}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ7}}$ (dedicated mode), these inputs indicate to the MC68302 that an external device is requesting an interrupt. Level 7 is the highest level and cannot be masked. Level 1 is the lowest level. Each one of these inputs can be programmed to be either level-sensitive or edge-sensitive.

FC2–FC0 — Function Codes 2–0

These bidirectional signals indicate the state and the cycle type currently being executed. The information indicated by the function code outputs is valid whenever $\overline{\text{AS}}$ is active.

These lines are outputs when the IMP (M68000 core, SDMA or IDMA) is the bus master and are inputs otherwise.

$\overline{\text{AVEC}}/\overline{\text{IOUT0}}$ — Autovector/Interrupt Output 0

In normal operation, this signal functions as the input $\overline{\text{AVEC}}$. $\overline{\text{AVEC}}$, when asserted during an interrupt acknowledge cycle, indicates that the M68000 core should use automatic vectoring for an interrupt. This pin operates like $\overline{\text{VPA}}$ on the MC68000, but is used for automatic vectoring only.

When the M68000 core is disabled, this pin operates as $\overline{\text{IOUT0}}$. $\overline{\text{IOUT2}}$–$\overline{\text{IOUT0}}$ provide the interrupt request output signals from the IMP interrupt controller to an external CPU when the M68000 core is disabled.

5.10 MC68302 BUS INTERFACE SIGNAL SUMMARY

Table 5-2 is a summary of all bus signals discussed in the previous paragraphs. It shows the direction of each pin for the following bus masters: M68000 core, IDMA, SDMA, and external. Each bus master can access either internal dual-port RAM and registers, or an external device or memory. When an external bus master accesses the internal dual-port RAM or registers, the access may be synchronous or asynchronous.

When the M68000 core is disabled, $\overline{BR}$ and $\overline{BG}$ change their direction, and $\overline{BCLR}$ becomes bidirectional.

Table 5-2(a). Bus Signal Summary — Core and External Master

	Master ▸	M68000 Core		External Master	
	Access to ▸	Internal Memory Space	External Memory Space	Internal Memory Space	External Memory Space
Signal Name	Pin Direction				
A23–A1, FC2–FC0 AS, UDS LDS, R/W RMC	I/O I/O	O O	O O	I I	I I
BCLR	I/O Open Drain	O	O	O	O
IAC	O	O	O	O	O
D15–D0 Read Write	I/O I/O	O O	I O	O I	I I
DTACK	I/O	O	**	O	**
BR	I/O	I	I	I	I
BG	I/O	O	O	O	O
BGACK	I/O	I	I	I	I
HALT	I/O Open Drain	I/O	I/O	I	I
RESET	I/O Open Drain	I/O	I/O	I	I
BERR	I/O Open Drain	I/O***	I/O***	I/O***	I/O***
IPL2–IPL0	I	I	I	I	I
AVEC	I	I	I	I	I
IOUT2–IOUT0	O	O	O	O	O

**If DTACK is generated automatically (internally) by the chip-select logic, then it is an output. Otherwise, it is an input.

***BERR is an open-drain output, and may be asserted by the IMP when the hardware watchdog is used, or when the chip-select logic detects address conflict or write protect violation. BERR may be asserted by external logic in all cases.

Table 5-2(b). Bus Signal Summary — IDMA and SDMA

Signal Name	Master ▸ / Access to ▸ / Pin Direction	IDMA		SDMA	
		Internal Memory Space	External Memory Space	Internal Memory Space	External Memory Space
A23–A1, FC2–FC0	I/O	O	O	N/A	O
AS, UDS	I/O	O	O	N/A	O
LDS, R/W					
RMC					
BCLR	I/O Open Drain	I†	I†	N/A	O
IAC	O	O	O	N/A	O
D15–D0 Read	I/O	O	I	N/A	I
Write	I/O	O	O	N/A	O
DTACK	I/O	O	**	N/A	**
BR	I/O	O††	O††	N/A	O††
BG	I/O	I††	I††	N/A	I††
BGACK	I/O	O	O	N/A	O
HALT	I/O Open Drain	I	I	N/A	I
RESET	I/O Open Drain	I	I	N/A	I
BERR	I/O Open Drain	I	I	N/A	I

**If DTACK is generated automatically (internally) by the chip-select logic, it is an output; otherwise, it is an input.

†Applies to disable CPU mode only. The internal signal IBCLR is used otherwise.

††Applies to disable CPU mode only.

5

5.11 PHYSICAL LAYER SERIAL INTERFACE PINS

The physical layer serial interface has 24 pins, and all but one of them have multiple functionality. The pins can be used in a variety of configurations in ISDN or non-ISDN environments. Table 5-3 shows the functionality of each group of pins and their internal connection to the three SCC and one SCP controllers. The physical layer serial interface can be configured for non-multiplexed operation (NMSI), or multiplexed operation that includes IDL, GCI and PCM highway modes. IDL and GCI are ISDN interfaces. When working in one of the multiplexed modes, the NMSI1/ISDN physical interface can be connected to all three SCC controllers.

Table 5-3. Serial Interface Pin Functions

1st Function	Connected To	2nd Function	Connected To
NMSI1(8)	SCC1 Controller	ISDN Interface	SCC1/SCC2/SCC3
NMSI2 (8)	SCC2 Controller	PIO — Port A	Parallel I/O
NMSI3 (5)	SCC3 Controller	PIO — Port A	Parallel I/O
(3)	SCC3 Controller	SCP	SCP Controller

NOTE: Each one of the parallel I/O pins can be configured individually.

5.12 TYPICAL SERIAL INTERFACE PIN CONFIGURATONS

Table 5-4 shows typical configurations of the physical layer interface pins for an ISDN environment. Table 5-5 shows potential configurations of the physical layer interface pins for a non-ISDN environment. The IDMA, IACK and Timer pins can be used in all applications either as dedicated functions or as PIO pins.

Table 5-4. Typical ISDN Configurations

Pins	Connected To	Used As
NMSI1 or ISDN I/F	SCC1 and SCC3	SCC1 Used as ISDN D-ch SCC3 Used as ISDN B2-ch
NMSI2	SCC2	SCC2 is Connected to Terminal
NMSI3	PA12–8 SCP	PIO (Extra Modem Signals and SCP Select Signals) Status/Control Exchange

NOTES:
1. ISDN environment with SCP port for status/control exchange and with existing terminal (for rate adaption).
2. D-ch is used for signaling.
3. B1-ch is used for voice (external CODEC required).
4. B2-ch is used for data transfer.

Table 5-5. Typical Generic Configurations

Pins	Connected To	Used As
NMSI1 or ISDN I/F	SCC1	Terminal With Modem
NMSI2	SCC2	Terminal With Modem
NMSI3	SCC3	Terminal Without Modem
	SCP	Status/Control Exchange

NOTE: Generic environment with three SCC ports (any protocol) and the SCP port. SCC3 does not use modem control signals.

5.13 NMSI1 OR ISDN INTERFACE PINS

These eight pins can be used either as NMSI1 in nonmultiplexed serial interface (NMSI) mode, or as an ISDN physical layer interface in IDL, GCI, and PCM highway modes. The input buffers have Schmitt triggers.

Table 5-6 shows the functionality of each pin in NMSI, GCI, IDL, and PCM highway modes.

Table 5-6. Mode Pin Functions

Signal Name	NMSI1		GCI		IDL		PCM	
RXD1/L1RXD	I	RXD1	I	L1RXD	I	L1RXD	I	L1RXD
TXD1/L1TXD	O	TXD1	O	L1TXD	O	L1TXD	O	L1TXD
RCLK1/L1CLK	I/O	RCLK1	I	L1CLK	I	L1CLK	I	L1CLK
TCLK1/L1SY0	I/O	TCLK1	O	SDS1	O	SDS1	I	L1SY0
$\overline{\text{CD1}}$/L1SY1	I	$\overline{\text{CD1}}$	I	L1SYNC	I	L1SYNC	I	L1SY1
$\overline{\text{CTS1}}$/L1GR	I	$\overline{\text{CTS1}}$	I	L1GR	I	L1GR		
$\overline{\text{RTS1}}$/L1RQ	O	$\overline{\text{RTS1}}$	O	GCIDCL	O	L1RQ		
BRG1	O	BRG1	O	BRG1	O	BRG1	O	BRG1

NOTES:
1. In IDL and GCI mode, SDS2 is output on the PA7 pin.
2. CD1 may be used as an external sync in NMSI mode.

5

RXD1/L1RXD — Receive Data/Layer-1 Receive Data
This input is used as the NMSI1 receive data in NMSI mode and as the receive data input in IDL, GCI, and PCM modes.

TXD1/L1TXD — Transmit Data/Layer-1 Transmit Data
This output is used as NMSI1 transmit data in NMSI mode and as the transmit data output in IDL, GCI, and PCM modes. TXD1 may be configured as an open-drain output in NMSI mode. L1TXD in IDL and PCM mode is a three-state output. In GCI mode, it is an open-drain output.

RCLK1/L1CLK — Receive Clock/Layer-1 Clock
This pin is used as an NMSI1 bidirectional receive clock in NMSI mode or as an input clock in IDL, GCI, and PCM modes. In NMSI mode, this signal is an input when SCC1 is working with an external clock and is an output when SCC1 is working with its baud rate generator.

TCLK1/L1SY0/SDS1 — Transmit Clock/PCM Sync/Serial Data Strobe 1
This pin is used as an NMSI1 bidirectional transmit clock in NMSI mode, as a sync signal in PCM mode, or as the SDS1 output in IDL/GCI modes.

In NMSI mode, this signal is an input when SCC1 is working with an external clock and is an output when SCC1 is working with its baud rate generator.

In PCM mode, L1SY1–L1SY0 are encoded signals used to create channels that can be independently routed to the SCCs.

Table 5-7. PCM Mode Signals

L1SY1	L1SY0	Data (RXD1,TXD1) is Routed to SCC
0	0	TXD1 is Three-Stated, RXD1 is Ignored
0	1	CH-1
1	0	CH-2
1	1	CH-3

NOTE: CH-1, 2, and 3 are connected to the SCCs as determined in the SIMODE register.

In IDL/GCI modes, the SDS2–SDS1 outputs may be used to route the B1 and/or B2 channels to devices that do not support the IDL or GCI buses. This is configured in the serial interface mode (SIMODE) and serial interface mask (SIMASK) registers.

$\overline{CD1}$/L1SY1 — Carrier Detect/Layer-1 Sync

This input is used as the NMSI1 carrier detect ($\overline{CD}$) pin in NMSI mode, as a PCM sync signal in PCM mode, and as an L1SYNC signal in IDL/GCI modes.

If the $\overline{CD1}$ pin has changed for more than one receive clock cycle, the IMP asserts the appropriate bit in the SCC1 event register. If the SCC1 channel is programmed not to support $\overline{CD1}$ automatically (in the SCC1 mode register), then this pin may be used as an external interrupt source. The current value of $\overline{CD1}$ may be read in the SCCS1 register. $\overline{CD1}$ may also be used as an external sync in NMSI mode.

$\overline{CTS1}$/L1GR — Clear to Send/Layer-1 Grant

This input is the NMSI1 $\overline{CTS}$ signal in the NMSI mode or the grant signal in the IDL/GCI mode. If this pin is not used as a grant signal in GCI mode, it should be connected to V_{DD}.

If the $\overline{CTS1}$ pin has changed for more than one transmit clock cycle, the IMP asserts the appropriate bit in the SCC1 event register and optionally aborts the transmission of that frame.

If SCC1 is programmed not to support $\overline{CTS1}$ (in the SCC1 mode register), then this pin may be used as an external interrupt source. The current value of the $\overline{CTS1}$ pin may be read in the SCCS1 register.

$\overline{RTS1}$/L1RQ/GCIDCL — Request to Send/Layer-1 request/GCI Clock Out
This output is the NMSI1 $\overline{RTS}$ signal in NMSI mode, the IDL request signal in IDL mode, or the GCI data clock output in GCI mode.

$\overline{RTS1}$ is asserted when SCC1 (in NMSI mode) has data or pad (flags or syncs) to transmit.

In GCI mode this pin is used to output the GCI data clock.

BRG1 — Baud Rate Generator 1
This output is always the baud rate generator clock of SCC1. (This pin used to be NC2.)

5.14 NMSI2 PORT OR PORT A PINS

These eight pins can be used either as the NMSI2 port or as a general-purpose parallel I/O port. Each one of these pins can be configured individually to be general-purpose I/O pins or a dedicated function in NMSI2. When they are used as NMSI2 pins they function exactly as the NMSI1 pins in NMSI mode.

The PA7 signal in dedicated mode becomes serial data strobe 2 (SDS2) in IDL and GCI modes. In IDL/GCI modes, the SDS2–SDS1 outputs may be used to route the B1 and/or B2 channels to devices that do not support the IDL or GCI buses. This is configured in the SIMODE and SIMASK registers. If SCC2 is in NMSI mode, this pin operates as BRG2, the output of the SCC2 baud rate generator, unless SDS2 is enabled to be asserted during the B1 or B2 channels of ISDN (bits SDC2–SDC1 of SIMODE). SDS2/BRG2 may be temporarily disabled by configuring it as a general-purpose output pin. The input buffers have Schmitt triggers. TCLK2 acts as the SCC2 baud rate generator output if SCC2 is in one of the multiplexed modes.

RXD2/PA0
TXD2/PA1
RCLK2/PA2
TCLK2/PA3
$\overline{CTS2}$/PA4
$\overline{RTS2}$/PA5
$\overline{CD2}$/PA6
SDS2/PA7/BRG2

Table 5-8. Baud Rate Generator Outputs

Source	NMSI	GCI	IDL	PCM
SCC1	BRG1	BRG1	BRG1	BRG1
SCC2	BRG2	TCLK2	TCLK2	TCLK2
SCC3	BRG3	TCLK3	TCLK3	TCLK3

5.15 NMSI3 PORT OR PORT A PINS OR SCP PINS

These eight pins can be used either as the NMSI3 port or as the NMSI3 port (less three modem lines) and the SCP port. If the SCP is enabled (EN bit in SPMODE register is set), then the three lines are connected to the SCP port. Otherwise, they are connected to the SCC3 port.

Each of the port A I/O pins can be configured individually to be general-purpose I/O pins or a dedicated function in NMSI3. When they are used as the NMSI3 pins, they function exactly as the NMSI1 pins (see the previous description). The input buffers have Schmitt triggers. TCLK3 acts as the SCC3 baud rate generator output if SCC3 is in one of the multiplexed modes.

RXD3/PA8
TXD3/PA9
RCLK3/PA10
TCLK3/PA11

SPRXD/$\overline{\text{CTS3}}$ — SCP Receive Serial Data/NMSI3 Clear-to-Send Pin
This signal functions as the SCP receive data input or may be used as the NMSI3 $\overline{\text{CTS}}$ input pin.

SPTXD/$\overline{\text{RTS3}}$ — SCP Transmit Serial Data/NMSI3 Request-to-Send Pin
This output is the SCP transmit data output or may be used as the NMSI3 $\overline{\text{RTS}}$ pin.

SPCLK/$\overline{\text{CD3}}$ — SCP Clock/NMSI3 $\overline{\text{CD}}$ Pin
This bidirectional signal is used as the SCP clock output or the NMSI3 $\overline{\text{CD3}}$ input pin.

PA12/BRG3
This pin functions as bit 12 of port A or may be used as the SCC3 baud rate generator output clock when SCC3 is operating in NMSI mode.

5.16 IDMA PINS OR PORT A PINS

Each of these three pins can be used either as dedicated pins for the IDMA signals or as general-purpose parallel I/O port A pins. Note that even if one or more of the IDMA pins are used as general-purpose I/O pins, the IDMA can still be used. For example, if $\overline{\text{DONE}}$ is not needed by the IDMA, it can be configured as a general-purpose I/O pin. If the IDMA is used for memory-to-memory transfers only, then all three pins can be used as general-purpose I/O pins. The input buffer of $\overline{\text{DACK}}$ has a Schmitt trigger.

$\overline{\text{DREQ}}$/PA13 — DMA Request
This input is asserted by a peripheral device to request an operand transfer between that peripheral device and memory. In the cycle steal request generation mode, this input is edge-sensitive. In burst mode, it is level-sensitive.

$\overline{\text{DACK}}$/PA14 — DMA Acknowledge
This output, asserted by the IDMA, signals to the peripheral that an operand is being transferred in response to a previous transfer request.

$\overline{\text{DONE}}$/PA15 — DONE
This bidirectional, open-drain signal is asserted by the IDMA or by a peripheral device during any IDMA bus cycle, to indicate that the data being transferred is the last item in a block. The IDMA asserts this signal as an output during a bus cycle when the byte count register is decremented to zero. Otherwise, this pin is an input to the IDMA to terminate IDMA operation.

5.17 IACK OR PIO PORT B PINS

Each one of these three pins can be used either as a dedicated interrupt acknowledge signal or as a general-purpose parallel I/O port. Note that the IMP interrupt controller does not require the IACK pins to support the three dedicated mode interrupt levels. The input buffers have Schmitt triggers.

$\overline{\text{IACK7}}$/PB0
$\overline{\text{IACK6}}$/PB1
$\overline{\text{IACK1}}$/PB2 — Interrupt Acknowledge/Port B I/O
As $\overline{\text{IACK1}}$, $\overline{\text{IACK6}}$, and $\overline{\text{IACK7}}$, these active low output signals indicate to the external device that the MC68302 is executing an interrupt acknowledge cycle. The external device must then place its vector number on the lower byte of the data bus or use $\overline{\text{AVEC}}$ for autovectoring (unless internal vector generation is used).

5.18 TIMER PINS

Each of these five pins can be used either as a dedicated timer function or as a general-purpose port B I/O port pin. Note that the timers do not require the use of external pins. The input buffers have Schmitt triggers.

TIN1/PB3 — Timer 1 Input
This input is used as a timer clock source for timer 1 or as a trigger for the timer 1 capture register. TIN1 may also be used as the external clock source for any or all three SCC baud rate generators.

$\overline{\text{TOUT1}}$/PB4 — Timer 1 Output
This output is used as an active-low pulse timeout or an event overflow output (toggle) from timer 1.

TIN2/PB5 — Timer 2 Input
This input can be used as a timer clock source for timer 2 or as a trigger for the timer 2 capture register.

$\overline{\text{TOUT2}}$/PB6 — Timer 2 Output
This output is used as an active-low pulse timeout or as an event overflow output (toggle) from timer 2.

$\overline{\text{WDOG}}$/PB7 — Watchdog Output
This active-low, open-drain output indicates expiration of the watchdog timer. $\overline{\text{WDOG}}$ is asserted for a period of 16 clock (CLKO) cycles and may be used to reset the MC68302. It is not asserted by the on-chip hardware watchdog (see the $\overline{\text{BERR}}$ signal description).

5.19 PARALLEL I/O PINS WITH INTERRUPT CAPABILITY

PB11–PB8 — Port B Parallel I/O pins
These four pins may be configured as a general-purpose parallel I/O ports with interrupt capability. Each of the pins can be configured either as an input or an output. When configured as an input, each pin can generate a separate, maskable interrupt on a high-to-low transition. PB8 may also be used to request a refresh cycle from the DRAM refresh controller, rather than as an I/O pin. The input buffers have Schmitt triggers.

5.20 CHIP-SELECT PINS

$\overline{CS0}/\overline{IOUT2}$ — Chip-Select 0/Interrupt Output 2

In normal operation, this pin functions as $\overline{CS0}$. $\overline{CS0}$ is one of the four active-low output pins that function as chip selects for external devices or memory. It does not activate on accesses to the internal RAM or registers (including the BAR or SCR registers).

When the M68000 core is disabled, this pin operates as $\overline{IOUT2}$. $\overline{IOUT2}$–$\overline{IOUT0}$ provide the interrupt request output signals from the IMP interrupt controller to an external CPU when the M68000 core is disabled.

$\overline{CS3}$–$\overline{CS1}$ — Chip Selects (3–1)

These three active-low output pins function as chip selects for external devices or memory. $\overline{CS3}$–$\overline{CS0}$ do not activate on accesses to the internal RAM or registers (including the BAR or SCR registers).

5.21 NO-CONNECT PINS

NC1 and NC3 output high values and are reserved for future use.

5.22 WHEN TO USE PULLUP RESISTORS

Pins that are input-only or output-only do not require external pullups. The bidirectional bus control signals require pullups since they are three-stated by the MC68302 when they are not being driven. Open-drain signals always require pullups.

Unused inputs should not be left floating. If they are input-only, they may be tied directly to V_{CC} or ground; otherwise, a pullup or pulldown resistor should be used. Unused outputs may be left unconnected. Unused I/O pins may be configured as outputs after reset and left unconnected.

If the MC68302 is to be held in reset for extended periods of time in an application (other than what occurs in normal power-on reset sequences) due to a special application requirement (such as V_{DD} dropping below required specifications, etc.), then three-stated signals and inputs should be pulled up or down. This decreases stress on the device transistors and saves power.

SECTION 6
ELECTRICAL CHARACTERISTICS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock (CLKO pin) and possibly to one or more other signals.

6.1 MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68302 MC68302I	T_A	 0 to 70 0 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{DD}).

6.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance for PGA	θ_{JA}	33	°C/W
Thermal Resistance for CQFP	θ_{JA}	46	°C/W

$T_J = T_A + (P_D \cdot \theta_{JA})$
$P_D = (V_{DD} \cdot I_{DD}) + P_{I/O}$
where:
$P_{I/O}$ is the power dissipation on pins.
For $T_A = 70°C$ and $P_D = 0.5$ W @ 12.5 MHz
$T_J = 88°C$

6

6.3 POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{DD} \times V_{DD}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < 0.3 \cdot P_{INT}$ and can be neglected.

If $P_{I/O}$ is neglected, an approximate relationship between P_D and T_J is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

6

6.4 POWER DISSIPATION

Characteristic	Symbol	Min	Max	Unit
Power Dissipation (Typical at 16.67 MHz) (See Note 1)	P_D	53	64	mA
Power Dissipation (Typical at 8 MHz) (See Note 1)	P_D	26	31	mA
Low Power Mode Dissipation (Typical at 16.67 MHz) (See Note 3)	LP_D	—	36	mA
Lowest Power Mode Dissipation (Typical at 16.67 MHz) (See Note 4)	LP_D	—	32	mA
Lowest Power Mode Dissipation (Typical at 50 kHz) (See Note 2)	LP_D	—	1	mA

NOTES:
1. The values shown are typical. The typical value varies as shown, based on how many IMP on-chip peripherals are enabled and the rate at which they are clocked.
2. The stated frequency must be externally applied to EXTAL only after the IMP has been placed in the lowest power mode with LPREC = 1. The M68000 core is not specified to operate at this frequency, but the rest of the IMP is. In this configuration, the user does not divide the clock internally using the LPCD4–LPCD0 bits in the system control register.
3. LPREC = 0. Divider = 2.
4. LPREC = 1. Divider = 1024.

6.5 DC ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	Min	Max	Unit
Input High Voltage (Except EXTAL)	V_{IH}	2.0	V_{DD}	V
Input Low Voltage (Except EXTAL)	V_{IL}	$V_{SS}-0.3$	0.8	V
Input High Voltage (EXTAL)	V_{CIH}	4.0	V_{DD}	V
Input Low Voltage (EXTAL)	V_{CIL}	$V_{SS}-0.3$	0.6	V
Input Leakage Current	I_{IN}	—	20	μA
Input Capacitance All Pins	C_{IN}	—	15	pF
Three-State Leakage Current (2.4/0.5 V)	I_{TSI}	—	20	μA
Open Drain Leakage Current (2.4 V)	I_{OD}	—	20	μA
Output High Voltage (I_{OH}=400 μA)	V_{OH}	$V_{DD}-1.0$	—	V
Output Low Voltage (I_{OL}=3.2 mA) A1–A23, PB0–PB11, FC0–FC3, $\overline{CS0}$–$\overline{CS3}$ IAC, $\overline{AVEC}$, $\overline{BG}$, RCLK1, RCLK2, RCLK3, TCLK1, TCLK2, TCLK3, $\overline{RTS1}$, $\overline{RTS2}$, $\overline{RTS3}$, $\overline{SDS2}$, PA12, RXD2, RXD3, $\overline{CTS2}$, $\overline{CD2}$, $\overline{CD3}$ $\overline{DREQ}$ (I_{OL}=5.3 mA) $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{BERR}$, $\overline{BGACK}$, $\overline{BCLR}$, $\overline{DTACK}$, $\overline{DACK}$, $\overline{RMC}$, $\overline{RMC}$, D0–D15, $\overline{RESET}$ (I_{OL}=7.0 mA) TXD1, TXD2, TXD3 (I_{OL}=8.9 mA) $\overline{BR}$, $\overline{DONE}$, $\overline{HALT}$, ($\overline{BR}$ as output) (I_{OL}=3.2 mA) CLKO	V_{OL}	— — — — —	V 0.5 0.5 0.5 0.5 0.4	V
Output Drive CLKO Output Drive ISDN I/F (GCI Mode) Output Drive All Other Pins	O_{CLK} O_{GCI} O_{ALL}	— — —	50 150 130	pF pF pF

6

6.6 DC ELECTRICAL CHARACTERISTICS — NMSI1 IN IDL MODE

Characteristic	Symbol	Min	Nom	Max	Unit	Condition
Power	V_{DD}	4.5	5.0	5.5	V	
Common	V_{SS}	0	0	0	V	
Temperature	T	0	25	70	°C	Operating Range
Input Pin Characteristics: L1CLK, L1SY1, L1RXD, L1GR						
Input Low Level Voltage	V_{IL}	−10%		+20%	V	(% of V_{DD})
Input High Level Voltage	V_{IH}	$V_{DD}-20\%$		$V_{DD}+10\%$	V	
Input Low Level Current	I_{IH}	—		±10	μA	$V_{in}=V_{SS}$
Input High Level Current	I_{IH}	—		±10	μA	$V_{in}=V_{DD}$
Output Pin Characteristics: L1TXD, SDS1–SDS2, L1RQ						
Output Low Level Voltage	V_{OL}	0		0.50	V	I_{OL}=2.0 mA
Output High Level Voltage	V_{OH}	$V_{DD}-0.5$		V_{DD}	V	I_{OH}=2.0 mA

6.7 AC ELECTRICAL SPECIFICATIONS — CLOCK TIMING

(see Figures 6-1, 6-2, 6-3, and 6-4)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
	Frequency of Operation	f	8	16.67	MHz
1	Clock Period (EXTAL)	t_{cyc}	60	125	ns
2, 3	Clock Pulse Width (EXTAL)	t_{CL}, t_{CH}	25	62.5	ns
4, 5	Clock Rise and Fall Times (EXTAL)	t_{Cr}, t_{Cf}	—	5	ns
5a	EXTAL to CLKO Delay (See Notes 1 and 2)	t_{CD}	2	11	ns

NOTE:
1. CLKO loading is 50 pF max.
2. CLKO skew from the rising and falling edges of EXTAL will not differ from each other by more than 1 ns, if the EXTAL rise time equals the EXTAL fall time.

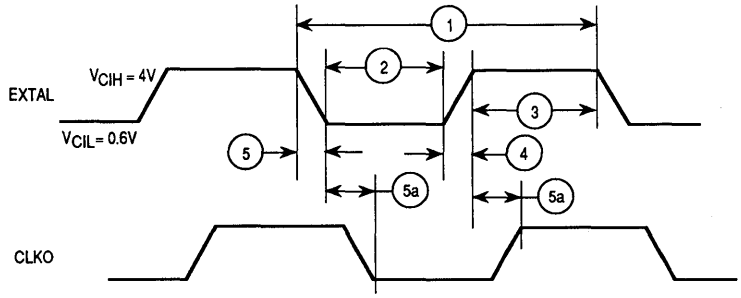


Figure 6-1. Clock Timing Diagram

6.8 AC ELECTRICAL SPECIFICATIONS — IMP BUS MASTER CYCLES

(see Figure 6-2, 6-3, and 6-4)

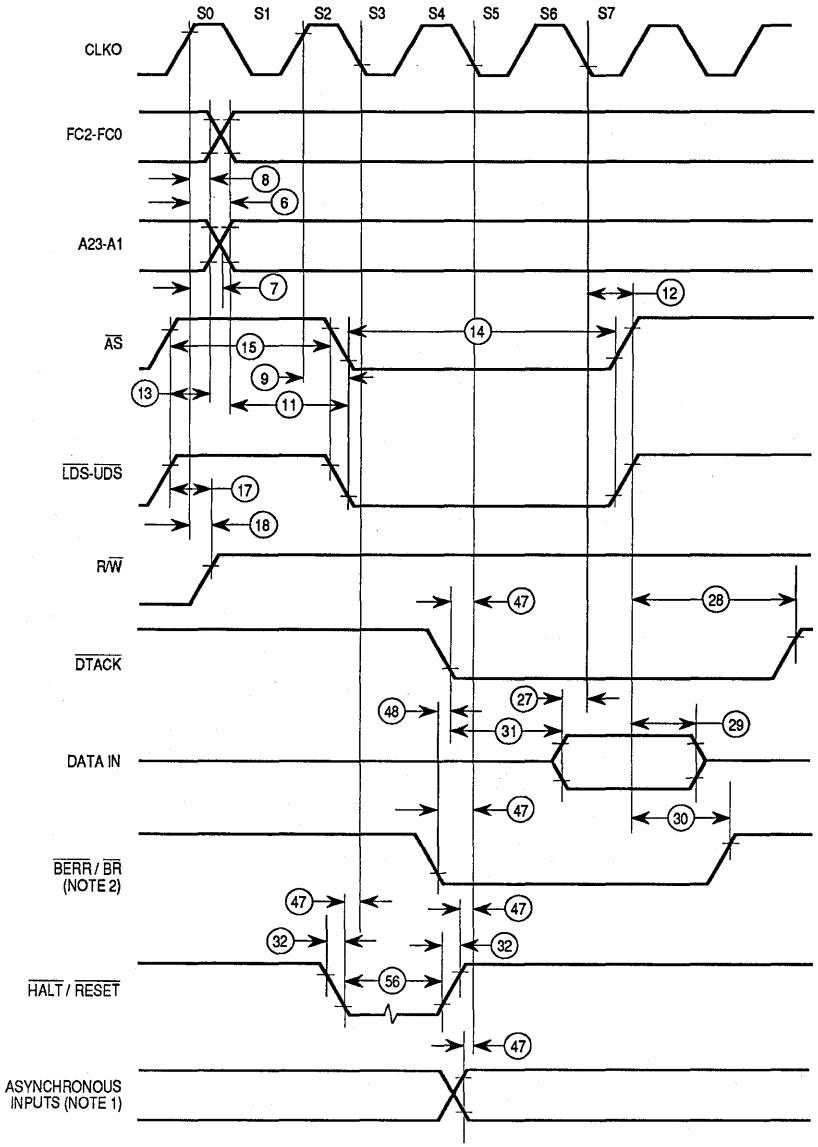
Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
6	Clock High to FC, Address Valid	$t_{CHFCADV}$	—	45	ns
7	Clock High to Address, Data Bus High Impedance (Maximum)	t_{CHADZ}	—	50	ns
8	Clock High to Address, FC Invalid (Minimum)	t_{CHAFI}	0	—	ns
9	Clock High to $\overline{AS}$, $\overline{DS}$ Asserted (see Note 1)	t_{CHSL}	3	30	ns
11	Address, FC Valid to $\overline{AS}$, $\overline{DS}$ Asserted (Read)/$\overline{AS}$ Asserted (Write) (see Note 2)	t_{AFCVSL}	15	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$ Negated (see Note 1)	t_{CLSH}	—	30	ns
13	$\overline{AS}$, $\overline{DS}$ Negated to Address, FC Invalid (see Note 2)	t_{SHAFI}	15	—	ns
14	$\overline{AS}$ (and $\overline{DS}$ Read) Width Asserted (see Note 2)	t_{SL}	120	—	ns
14A	$\overline{DS}$ Width Asserted, Write (see Note 2)	t_{DSL}	60	—	ns
15	$\overline{AS}$, $\overline{DS}$ Width Negated (see Note 2)	t_{SH}	60	—	ns
16	Clock High to Control Bus High Impedance	t_{CHCZ}	—	50	ns
17	$\overline{AS}$, $\overline{DS}$ Negated to R/$\overline{W}$ Invalid (see Note 2)	t_{SHRH}	15	—	ns
18	Clock High to R/$\overline{W}$ High (see Note 1)	t_{CHRH}	—	30	ns
20	Clock High to R/$\overline{W}$ Low (see Note 1)	t_{CHRL}	—	30	ns
20A	$\overline{AS}$ Asserted to R/$\overline{W}$ Low (Write) (see Notes 2 and 6)	t_{ASRV}	—	10	ns
21	Address FC Valid to R/$\overline{W}$ Low (Write) (see Note 2)	t_{AFCVRL}	15	—	ns
22	R/$\overline{W}$ Low to $\overline{DS}$ Asserted (Write) (see Note 2)	t_{RLSL}	30	—	ns
23	Clock Low to Data-Out Valid	t_{CLDO}	—	30	ns
25	$\overline{AS}$, $\overline{DS}$, Negated to Data-Out Invalid (Write) (see Note 2)	t_{SHDOI}	15	—	ns
26	Data-Out Valid to $\overline{DS}$ Asserted (Write) (see Note 2)	t_{DOSL}	15	—	ns
27	Data-In Valid to Clock Low (Setup Time on Read) (see Note 5)	t_{DICL}	7	—	ns
28	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DTACK}$ Negated (Asynchronous Hold) (see Note 2)	t_{SHDAH}	0	110	ns
29	$\overline{AS}$, $\overline{DS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{SHDII}	0	—	ns
30	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{BERR}$ Negated	t_{SHBEH}	0	—	ns
31	$\overline{DTACK}$ Asserted to Data-In Valid (Setup Time) (see Notes 2 and 5)	t_{DALDI}	—	50	ns
32	$\overline{HALT}$ and $\overline{RESET}$ Input Transition Time	t_{RHr}, t_{RHf}	—	150	ns
33	Clock High to $\overline{BG}$ Asserted	t_{CHGL}	—	30	ns
34	Clock High to $\overline{BG}$ Negated	t_{CHGH}	—	30	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted	t_{BRLGL}	2.5	4.5	clks
36	$\overline{BR}$ Negated to $\overline{BG}$ Negated (see Note 7)	t_{BRHGH}	1.5	2.5	clks
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GALGH}	2.5	4.5	clks
37A	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated (see Note 8)	t_{GALBRH}	10	1.5	ns/clks
38	$\overline{BG}$ Asserted to Control, Address, Data Bus High Impedance ($\overline{AS}$ Negated)	t_{GLZ}	—	50	ns
39	$\overline{BG}$ Width Negated	t_{GH}	1.5	—	clks
44	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{AVEC}$ Negated	t_{SHVPH}	0	50	ns
46	$\overline{BGACK}$ Width Low	t_{GAL}	1.5	—	clks

— Continued

6.8 AC ELECTRICAL SPECIFICATIONS — IMP BUS MASTER CYCLES (Continued) (see Figure 6-2, 6-3, and 6-4)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
47	Asynchronous Input Setup Time (see Note 5)	t_{ASI}	10	—	ns
48	$\overline{BERR}$ Asserted to $\overline{DTACK}$ Asserted (see Notes 2 and 3)	t_{BELDAL}	10	—	ns
53	Data-Out Hold from Clock High	t_{CHDOI}	0	—	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RLDBD}	0	—	ns
56	$\overline{HALT}$/$\overline{RESET}$ Pulse Width (see Note 4)	t_{HRPW}	10	—	clks
57	$\overline{BGACK}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven	t_{GASD}	1.5	—	clks
57A	$\overline{BGACK}$ Negated to FC	t_{GAFD}	1	—	clks
58	$\overline{BR}$ Negated to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ Driven (see Note 7)	t_{RHSD}	1.5	—	clks
58A	$\overline{BR}$ Negated to FC (see Note 7)	t_{RHFD}	1	—	clks
60	Clock High to $\overline{BCLR}$ Asserted	t_{CHBCL}	—	30	ns
61	Clock High to $\overline{BCLR}$ Negated (See Note 10)	t_{CHBCH}	—	30	ns
62	Clock Low (S0 Falling Edge during read) to $\overline{RMC}$ Asserted	t_{CLRML}	—	30	ns
63	Clock High (S7 Rising Edge during write) to $\overline{RMC}$ Negated	t_{CHRMH}	—	30	ns
64	$\overline{RMC}$ Negated to $\overline{BG}$ Asserted (see Note 9)	t_{RMHGL}	—	30	ns

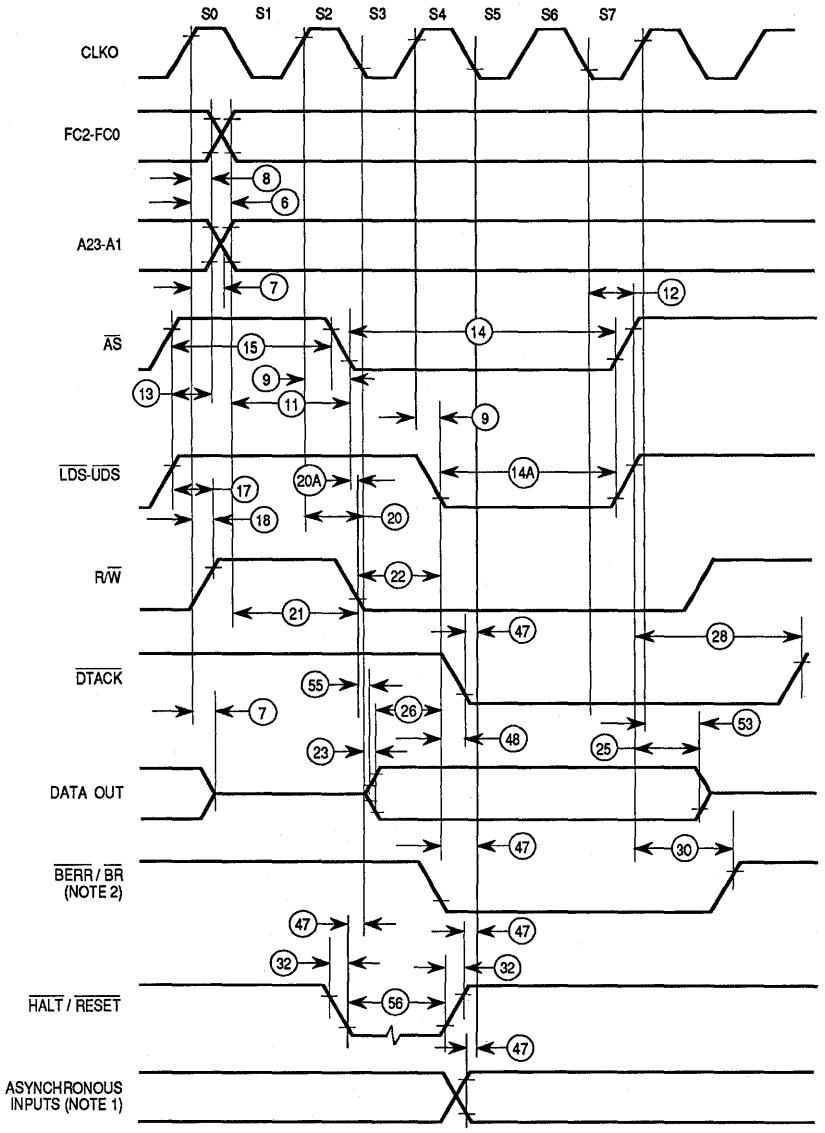
NOTES:
1. For loading capacitance of less than or equal to 50 picofarads, subtract 4 nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock period.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be ignored. In the absence of $\overline{DTACK}$, $\overline{BERR}$ is a synchronous input using the asynchronous input setup time (#47).
4. For powerup, the MC68302 must be held in the reset state for 100 milliseconds to allow stabilization of on-chip circuit. After the system is powered up #56 refers to the minimum pulse width required to reset the processor.
5. If the asynchronous input setup (#47) requirement is satisfied for $\overline{DTACK}$, the $\overline{DTACK}$ asserted to data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock low setup time (#27) for the following clock cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20%), subtract 5 nanoseconds from the values given in these columns.
7. The MC68302 will negate $\overline{BG}$ and begin driving the bus if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. The minimum value must be met to guarantee proper operation. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.
9. This specification is valid only when the RMCST bit is set in the SCR register.
10. Occurs on S0 of SDMA read/write access when the SDMA becomes bus master.



NOTES:

1. Setup time for the asynchronous inputs $\overline{IPL2}$-$\overline{IPL0}$ guarantees their recognition at the next falling edge of the clock.
2. $\overline{BR}$ need fall at this time only to insure being recognized at the end of the bus cycle.
3. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall is linear between 0.8 volt and 2.0 volts.

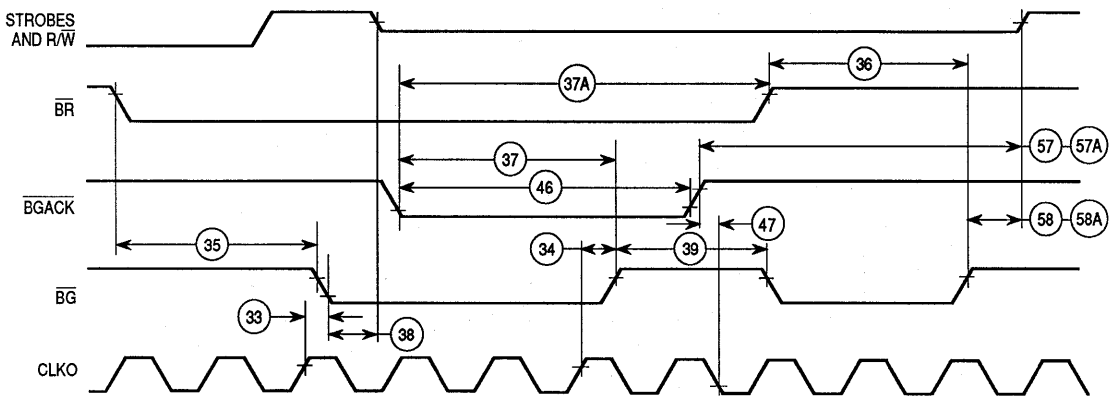
Figure 6-2. Read-Cycle Timing Diagram



NOTES:

1. Timing measurements are referenced to and from a low voltage of 0.8 volt and a high voltage of 2.0 volts, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall is linear between 0.8 volt and 2.0 volts.
2. Because of loading variations, R/$\overline{W}$ may be valid after $\overline{AS}$ even though both are initiated by the rising edge of S2 (specification #20A).

Figure 6-3. Write-Cycle Timing Diagram



NOTE: Setup time to the clock (#47) for the asynchronous inputs BERR, BGACK, BR, DTACK, and IPL2-IPL0 guarantees their recognition at the next falling edge of the clock.

Figure 6-4. Bus Arbitration Timing Diagram

6.9 AC ELECTRICAL SPECIFICATIONS — DMA

(see Figure 6-5)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
80	$\overline{DREQ}$ Asynchronous Setup Time (see Note 1)	t_{REQASI}	15	—	ns
81	$\overline{DREQ}$ Width Low (see Note 2)	t_{REQL}	2	—	clk
82	$\overline{DREQ}$ Low to $\overline{BR}$ Low (see Notes 3 and 4)	$t_{REQLBRL}$	—	2	clk
83	Clock High to $\overline{BR}$ Low (see Notes 3 and 4)	t_{CHBRL}	—	30	ns
84	Clock High to $\overline{BR}$ High Impedance (see Notes 3 and 4)	t_{CHBRZ}	—	30	ns
85	$\overline{BGACK}$ Low to $\overline{BR}$ High Impedance (see Notes 3 and 4)	t_{BKLBRZ}	30	—	ns
86	Clock High to $\overline{BGACK}$ Low	t_{CHBKL}	—	30	ns
87	$\overline{AS}$ and $\overline{BGACK}$ High (the latest one) to $\overline{BGACK}$ Low (when $\overline{BG}$ is Asserted)	t_{ABHBKL}	1.5	2.5 +30	clk ns
88	$\overline{BG}$ Low to $\overline{BGACK}$ Low (No Other Bus Master) (see Notes 3 and 4)	t_{BGLBKL}	1.5	2.5 +30	clk ns
89	$\overline{BR}$ High Impedance to $\overline{BG}$ High (see Notes 3 and 4)	t_{BRHBGH}	0	—	ns
90	Clock on Which $\overline{BGACK}$ Low to Clock on Which $\overline{AS}$ Low	$t_{CLBKLAL}$	2	2	clk
91	Clock High to $\overline{BGACK}$ High	t_{CHBKH}	—	30	ns
92	Clock Low to $\overline{BGACK}$ High Impedance	t_{CLBKZ}	—	15	ns
93	Clock High to $\overline{DACK}$ Low	t_{CHACKL}	—	30	ns
94	Clock Low to $\overline{DACK}$ High	t_{CLACKH}	—	30	ns
95	Clock High to $\overline{DONE}$ Low (Output)	t_{CHDNL}	—	30	ns
96	Clock Low to $\overline{DONE}$ High Impedance	t_{CLDNZ}	—	30	ns
97	$\overline{DONE}$ Input Low to Clock High (Asynchronous Setup)	t_{DNLTCH}	15	—	ns

NOTES:

1. $\overline{DREQ}$ is sampled on the falling edge of CLK in cycle steal and burst modes.
2. If #80 is satified for $\overline{DREQ}$, #81 may be ignored.
3. $\overline{BR}$ will not be asserted while $\overline{AS}$, $\overline{HALT}$, or $\overline{BERR}$ is asserted.
4. Specifications are for DISABLE CPU mode only.
5. $\overline{DREQ}$, $\overline{DACK}$, and $\overline{DONE}$ do not apply to the SDMA channels.
6. IDMA and SDMA read and write cycle timing is the same as that for the M68000 core.

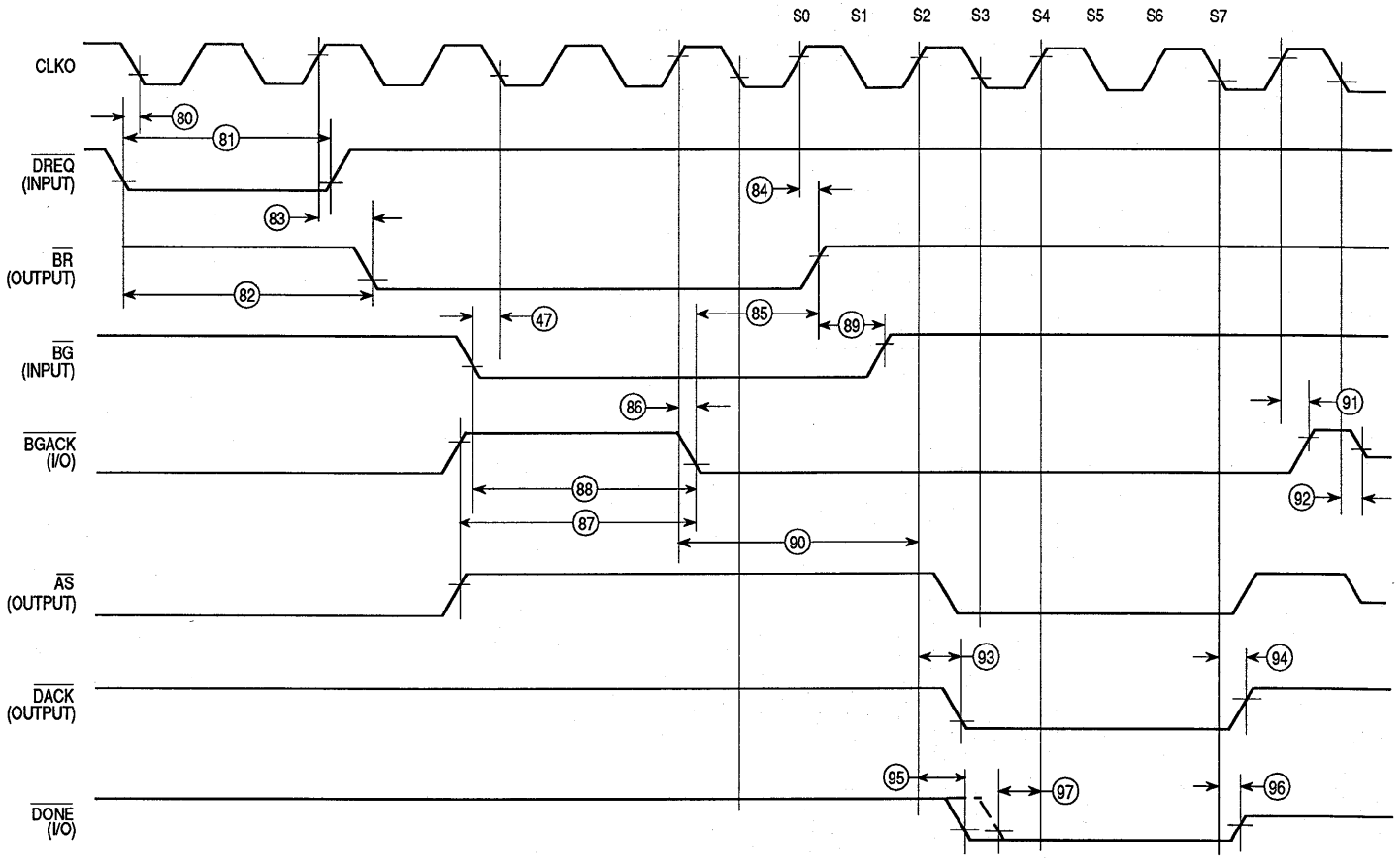


Figure 6-5. DMA Timing Diagram

6.10 AC ELECTRICAL SPECIFICATIONS — EXTERNAL MASTER INTERNAL ASYNCHRONOUS READ/WRITE CYCLES

(see Figures 6-6 and 6-7)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
100	R/$\overline{W}$ Valid to $\overline{DS}$ Low	t_{RWVDSL}	0	—	ns
101	$\overline{DS}$ Low to Data In Valid	t_{DSLDIV}	—	30	ns
102	$\overline{DTACK}$ Low to Data In Hold Time	t_{DKLDH}	0	—	ns
103	$\overline{AS}$ Valid to $\overline{DS}$ Low	t_{ASVDSL}	0	—	ns
104	$\overline{DTACK}$ Low to $\overline{DS}$ High	t_{DKLDSH}	0	—	ns
105	$\overline{DS}$ High to $\overline{DTACK}$ High	t_{DSHDKH}	—	45	ns
106	$\overline{DS}$ Inactive to $\overline{AS}$ Inactive	t_{DSIASI}	0	—	ns
107	$\overline{DS}$ High to R/$\overline{W}$ High	t_{DSHRWH}	0	—	ns
108	$\overline{DS}$ High to Data High Impedance	t_{DSHDZ}	—	45	ns
108A	$\overline{DS}$ High to Data Out Hold Time	t_{DSHDH}	0	—	ns
109	$\overline{DS}$ High to Data In Hold Time (see Note)	t_{DSHDOH}	0	—	ns
109A	Data Out Valid to $\overline{DTACK}$ Low	t_{DOVDKL}	15	—	ns

NOTE: If $\overline{AS}$ is negated before $\overline{DS}$, the data bus could be three-stated (spec 126) before $\overline{DS}$ is negated.

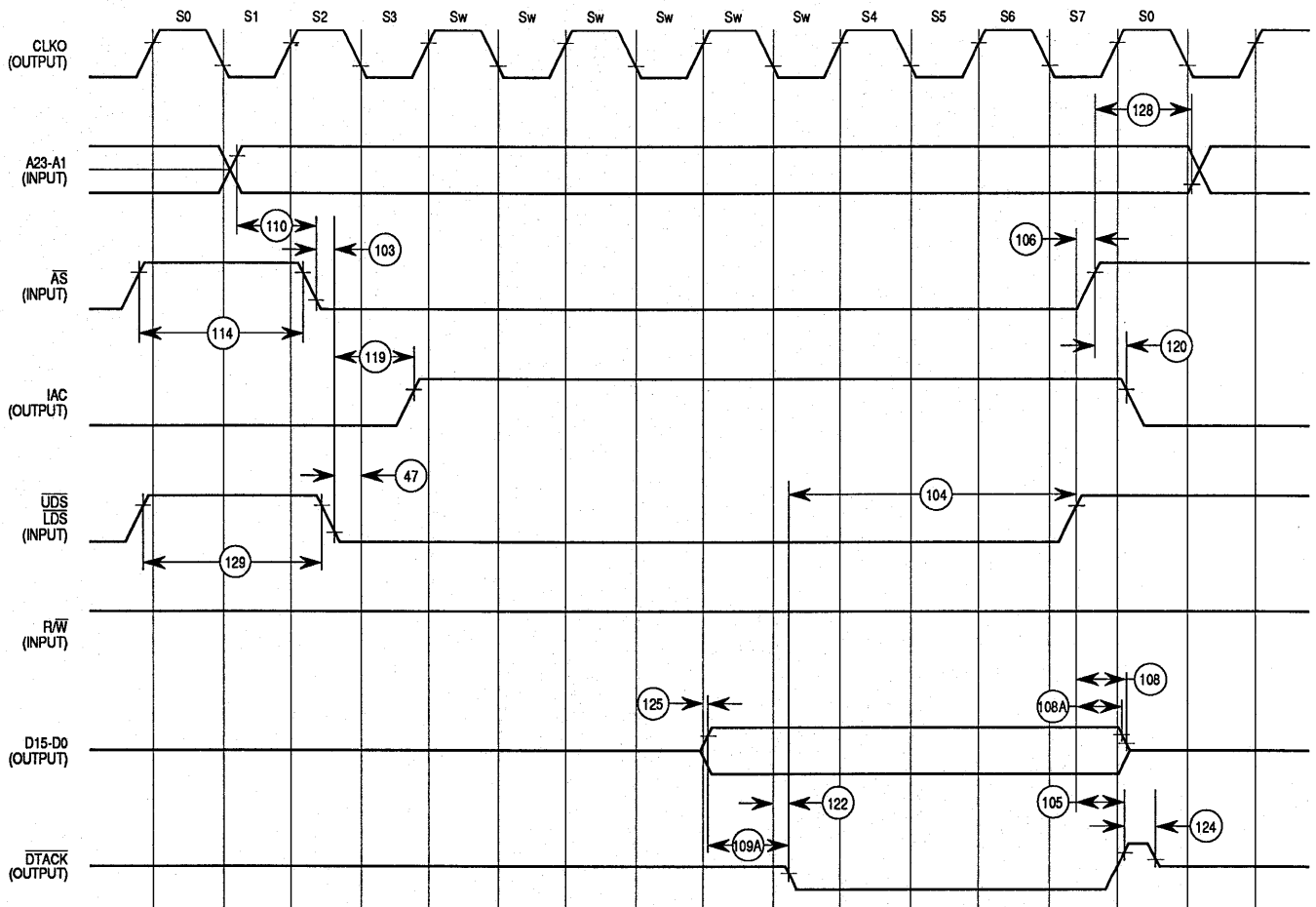


Figure 6-6. External Master Internal Asynchronous Read Cycle Timing Diagram

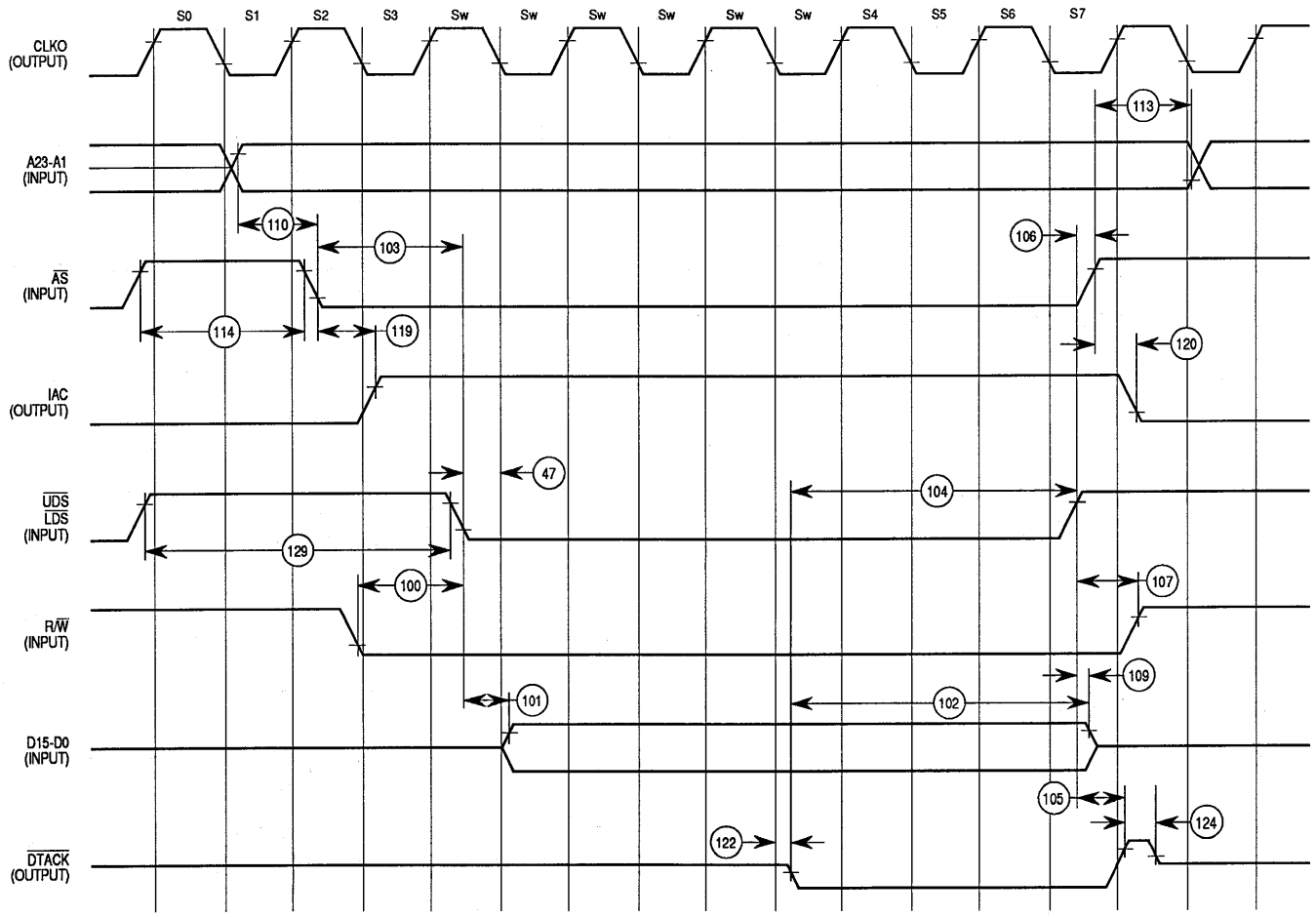


Figure 6-7. External Master Internal Asynchronous Write Cycle Timing Diagram

6.11 AC ELECTRICAL SPECIFICATIONS — EXTERNAL MASTER INTERNAL SYNCHRONOUS READ/WRITE CYCLES

(see Figures 6-8, 6-9, and 6-10)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
110	Address Valid to $\overline{AS}$ Low	t_{AVASL}	15	—	ns
111	$\overline{AS}$ Low to Clock High	t_{ASLCH}	30	—	ns
112	Clock Low to $\overline{AS}$ High	t_{CLASH}	—	45	ns
113	$\overline{AS}$ High to Address Hold Time on Write	t_{ASHAH}	0	—	ns
114	$\overline{AS}$ Inactive Time	t_{ASH}	1	—	clk
115	$\overline{UDS}/\overline{LDS}$ Low to Clock High	t_{SLCH}	40	—	ns
116	Clock Low to $\overline{UDS}/\overline{LDS}$ High	t_{CLSH}	—	45	ns
117	R/$\overline{W}$ Valid to Clock High	t_{RWVCH}	30	—	ns
118	Clock High to R/$\overline{W}$ High	t_{CHRWH}	—	45	ns
119	$\overline{AS}$ Low to IAC High	t_{ASLIAH}	—	40	ns
120	$\overline{AS}$ High to IAC Low	t_{ASHIAL}	—	40	ns
121	$\overline{AS}$ Low to $\overline{DTACK}$ Low (0 Wait State)	t_{ASLDTL}	—	45	ns
122	Clock Low to $\overline{DTACK}$ Low (1 Wait State)	t_{CLDTL}	—	30	ns
123	$\overline{AS}$ High to $\overline{DTACK}$ High	t_{ASHDTH}	—	45	ns
124	$\overline{DTACK}$ High to $\overline{DTACK}$ High Impedance	t_{DTHDTZ}	—	15	ns
125	Clock High to Data Out Valid	t_{CHDOV}	—	30	ns
126	$\overline{AS}$ High to Data High Impedance	t_{ASHDZ}	—	45	ns
127	$\overline{AS}$ High to Data Out Hold Time	t_{ASHDOI}	0	—	ns
128	$\overline{AS}$ High to Address Hold Time on Read	t_{ASHAI}	0	—	ns
129	$\overline{UDS}/\overline{LDS}$ Inactive Time	t_{SH}	1	—	clk
130	Data In Valid to Clock Low	t_{CLDIV}	30	—	ns
131	Clock Low to Data In Hold Time	t_{CLDIH}	15	—	ns

NOTE: Specifications are valid only when SAM = 1 in the SCR.

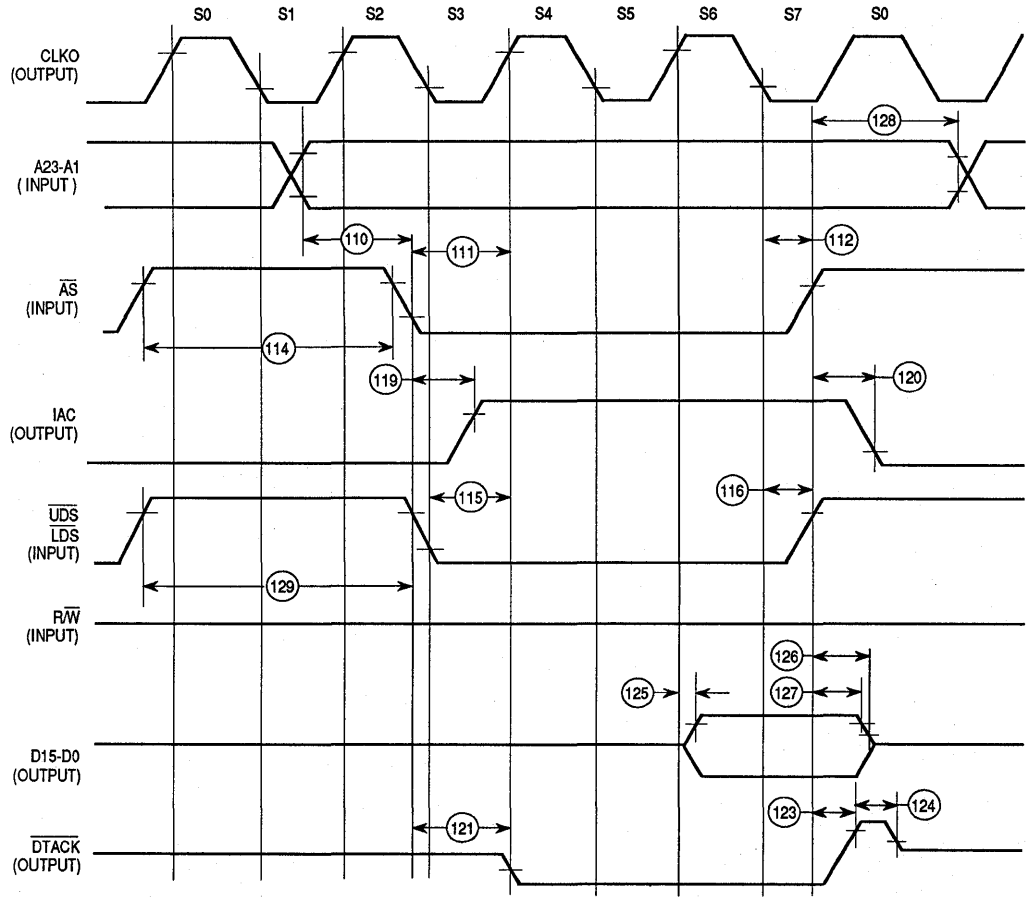


Figure 6-8. External Master Internal Synchronous Read Cycle Timing Diagram

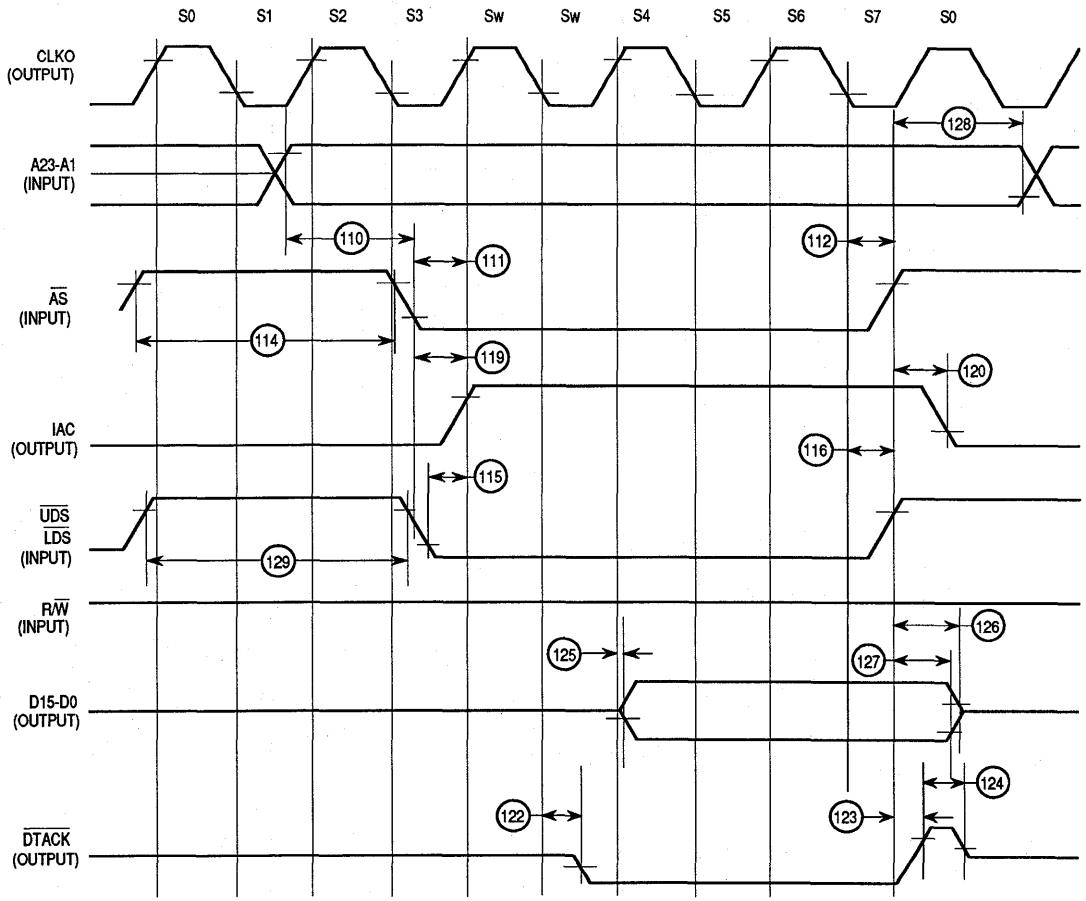


Figure 6-9. External Master Internal Synchronous Read Cycle Timing Diagram (One Wait State)

6

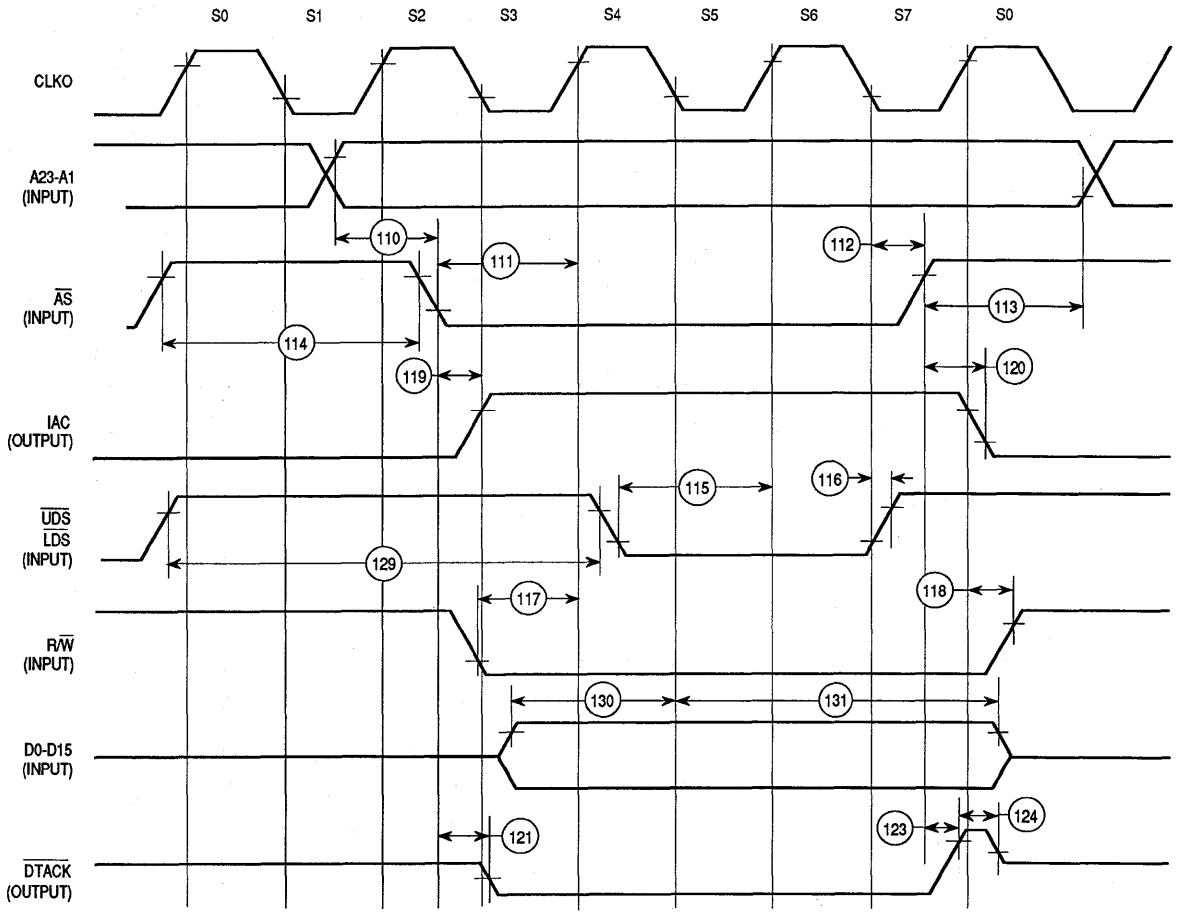


Figure 6-10. External Master Internal Synchronous Write Cycle Timing Diagram

6.12 AC ELECTRICAL SPECIFICATIONS — INTERNAL MASTER INTERNAL READ/WRITE CYCLES

(see Figure 6-11)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
140	Clock High to IAC High	t_{CHIAH}	—	40	ns
141	Clock Low to IAC Low	t_{CLIAL}	—	40	ns
142	Clock High to $\overline{DTACK}$ Low (0 Wait State)	t_{CHDTL}	—	45	ns
143	Clock Low to $\overline{DTACK}$ High	t_{CLDTH}	—	40	ns
144	Clock High to Data Out Valid	t_{CHDOV}	—	30	ns
145	$\overline{AS}$ High to Data Out Hold Time	t_{ASHDOH}	0	—	ns

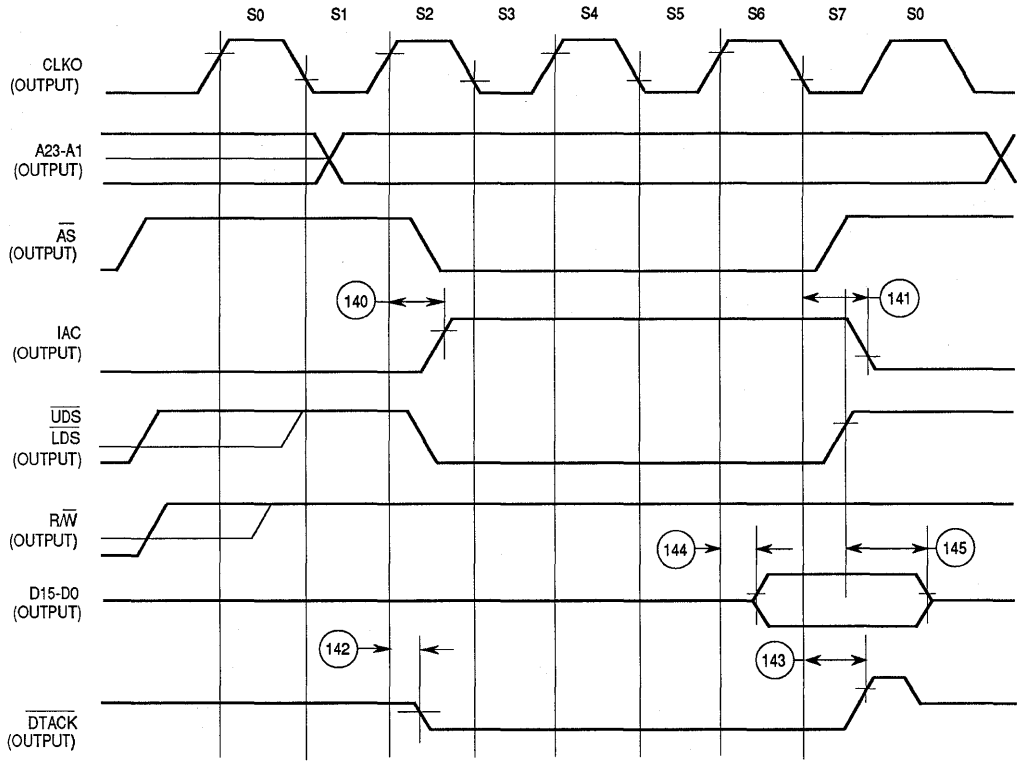


Figure 6-11. Internal Master Internal Read Cycle Timing Diagram

6.13 AC ELECTRICAL SPECIFICATIONS — CHIP-SELECT TIMING INTERNAL MASTER

(see Figure 6-12)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
150	Clock High to $\overline{CS}$, $\overline{IACK}$ Low (see Note 2)	$t_{CHCSIAKL}$	—	40	ns
151	Clock Low to $\overline{CS}$, $\overline{IACK}$ High (see Note 2)	$t_{CLCSIAKH}$	—	40	ns
152	$\overline{CS}$ Width Negated	t_{CSH}	60	—	ns
153	Clock High to $\overline{DTACK}$ Low (0 Wait State)	t_{CHDTKL}	—	45	ns
154	Clock Low to $\overline{DTACK}$ Low (1–6 Wait States)	t_{CLDTKL}	—	30	ns
155	Clock Low to $\overline{DTACK}$ High	t_{CLDTKH}	—	40	ns
156	Clock High to $\overline{BERR}$ Low (see Note 1)	t_{CHBERL}	—	40	ns
157	Clock Low to $\overline{BERR}$ High Impedance (see Note 1)	t_{CLBERH}	—	40	ns
158	$\overline{DTACK}$ High to $\overline{DTACK}$ High Impedance	$t_{DTKHDTKZ}$	—	15	ns
171	Input Data Hold Time from S6 Low	t_{IDHCL}	5	—	ns
172	$\overline{CS}$ Negated to Data Out Invalid (Write)	t_{CSNDOI}	10	—	ns
173	Address, FC Valid to $\overline{CS}$ Asserted	t_{AFVCSA}	15	—	ns
174	$\overline{CS}$ Negated to Address, FC Invalid	t_{CSNAFI}	15	—	ns
175	$\overline{CS}$ Low Time (0 Wait States)	t_{CSLT}	120	—	ns
176	$\overline{CS}$ Negated to R/$\overline{W}$ Invalid	t_{CSNRWI}	10	—	ns
177	$\overline{CS}$ Asserted to R/$\overline{W}$ Low (Write)	t_{CSARWL}	—	10	ns
178	$\overline{CS}$ Negated to Data In Invalid (Hold Time on Read)	t_{CSNDII}	0	—	ns

NOTE:

1. This specification is valid only when the ADCE or WPVE bits in the SCR are set.
2. For loading capacitance less than or equal to 50 pF, subtract 4 ns from the maximum value given.
3. Specs 172–178 do not have diagrams. However, similar diagrams for $\overline{AS}$ are shown as 25, 11, 13, 14, 17, 20A, and 29, respectively.

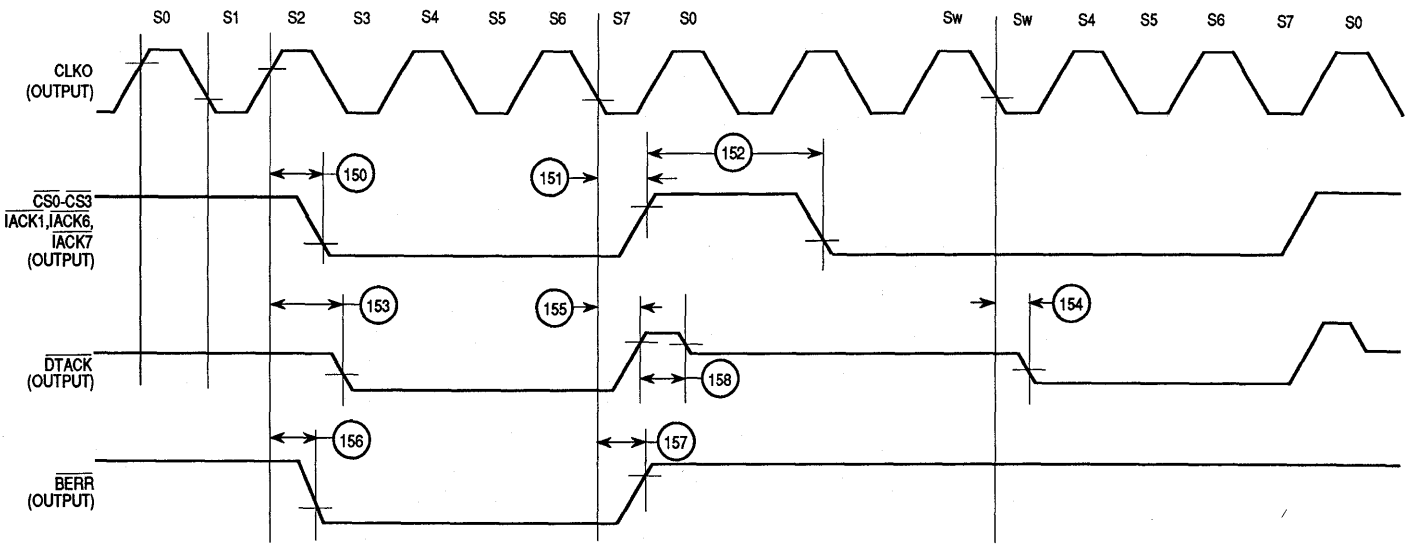


Figure 6-12. Internal Master Chip-Select Timing Diagram

6.14 AC ELECTRICAL SPECIFICATIONS — CHIP-SELECT TIMING EXTERNAL MASTER

(see Figure 6-13)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
154	Clock Low to $\overline{DTACK}$ Low (1–6 Wait States)	t_{CLDTKL}	—	30	ns
160	$\overline{AS}$ Low to $\overline{CS}$ Low	t_{ASLCSL}	—	30	ns
161	$\overline{AS}$ High to $\overline{CS}$ High	t_{ASHCSH}	—	30	ns
162	Address Valid to $\overline{AS}$ Low	t_{AVASL}	15	—	ns
163	R/$\overline{W}$ Valid to $\overline{AS}$ Low (see Note 1)	t_{RWVASL}	15	—	ns
164	$\overline{AS}$ Negated to Address Hold Time	t_{ASHAI}	0	—	ns
165	$\overline{AS}$ Low to $\overline{DTACK}$ Low (0 Wait State)	$t_{ASLDTKL}$	—	45	ns
167	$\overline{AS}$ High to $\overline{DTACK}$ High	$t_{ASHDTKH}$	—	30	ns
168	$\overline{AS}$ Low to $\overline{BERR}$ Low (see Note 2)	$t_{ASLBERL}$	—	30	ns
169	$\overline{AS}$ High to $\overline{BERR}$ High (see Notes 2 and 3)	$t_{ASHBERH}$	—	30	ns

NOTES:
1. The minimum value must be met to guarantee write protection operation.
2. This specification is valid when the ADCE or WPVE bits in the SCR are set.
3. Also applies after a timeout of the hardware watchdog.

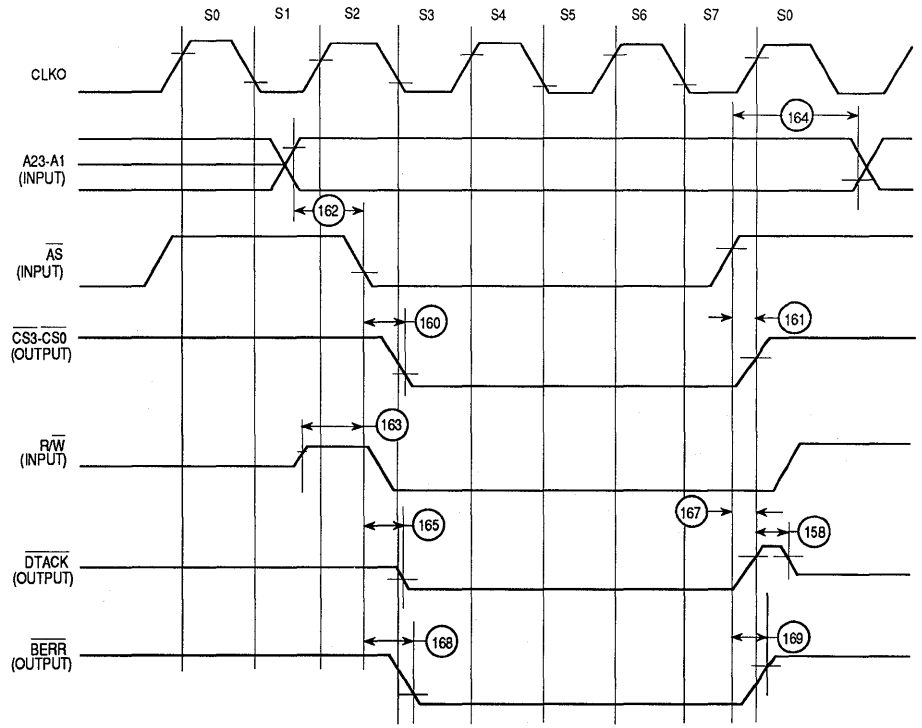


Figure 6-13. External Master Chip-Select Timing Diagram

6.15 AC ELECTRICAL SPECIFICATIONS — PARALLEL I/O

(see Figure 6-14)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
180	Input Data Setup Time (to Clock Low)	t_{DSU}	20	—	ns
181	Input Data Hold Time (from Clock Low)	t_{DH}	10	—	ns
182	Clock High to Data out Valid (CPU Writes Data, Control, or Direction)	t_{CHDOV}	—	35	ns

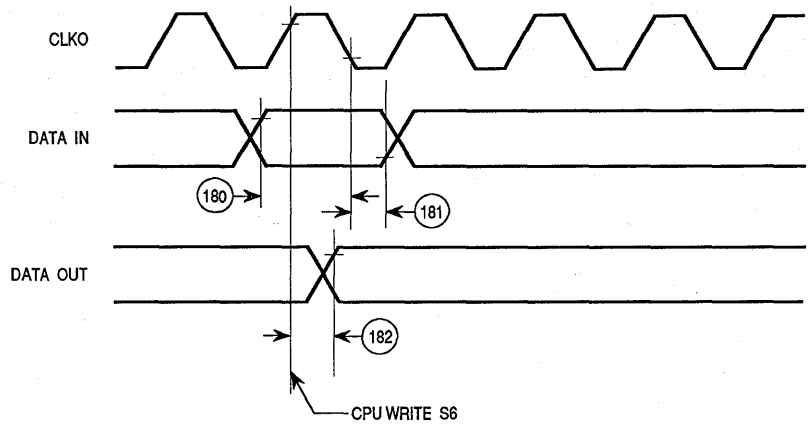


Figure 6-14. Parallel I/O Data In/Data Out Timing Diagram

6.16 AC ELECTRICAL SPECIFICATIONS — INTERRUPTS

(see Figure 6-15)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
190	Interrupt Pulse Width Low $\overline{IRQ}$ (Edge Triggered Mode)	t_{IPW}	50	—	ns
191	Minimum Time Between Active Edges	t_{AEMT}	3	—	clk

NOTE: Setup time for the asynchronous inputs IPL2–IPL0 and $\overline{AVEC}$ guarantees their recognition at the next falling edge of the clock.

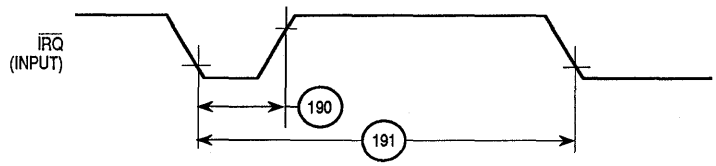


Figure 6-15. Interrupts Timing Diagram

6.17 AC ELECTRICAL SPECIFICATIONS — TIMERS

(see Figure 6-16)

Num.	Characteristic	Symbol	16.67 MHz		Unit
			Min	Max	
200	Timer Input Capture Pulse Width	t_{TPW}	50	—	ns
201	TIN Clock Low Pulse Width	t_{TICLT}	50	—	ns
202	TIN Clock High Pulse Width	t_{TICHT}	1.5	—	clk
203	TIN Clock Cycle Time	t_{cyc}	3	—	clk
204	Clock High to TOUT Valid	t_{CHTOV}	—	35	ns
205	$\overline{FRZ}$ Input Setup Time (to Clock High) (see Note)	t_{FRZSU}	20	—	ns
206	$\overline{FRZ}$ Input Hold Time (from Clock High)	t_{FRZHT}	10	—	ns

NOTE: $\overline{FRZ}$ should be negated during total system reset.

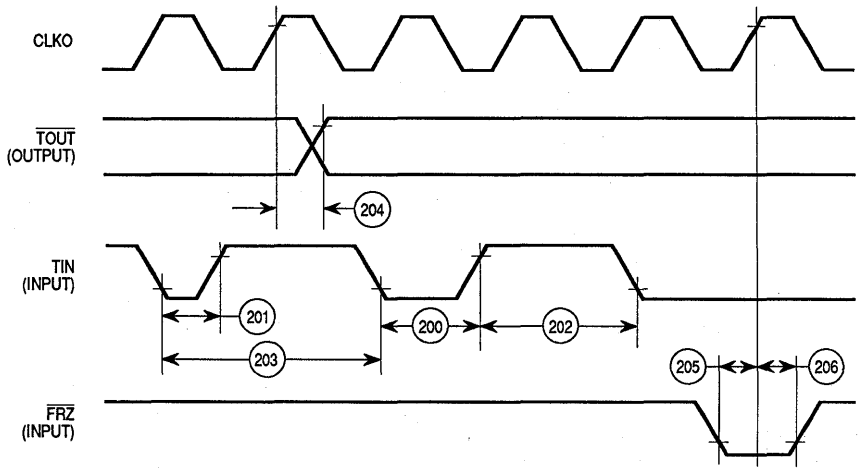


Figure 6-16. Timers Timing Diagram

6.18 AC ELECTRICAL SPECIFICATIONS — SERIAL COMMUNICATION PORT

(see Figure 6-17)

Num.	Characteristic	16.67 MHz		Unit
		Min	Max	
250	SPCLK Clock Output Period	4	64	clks
251	SPCLK Clock Output Rise/Fall Time	—	15	ns
252	Delay from SPCLK to Transmit (see Note 1)	0	40	ns
253	SCP Receive Setup Time (see Note 1)	40	—	ns
254	SCP Receive Hold Time (see Note 1)	10	—	ns

NOTES:
1. This also applies when SPCLK is inverted by CI in the SPMODE register.
2. The enable signals for the slaves may be implemented by the parallel I/O pins.

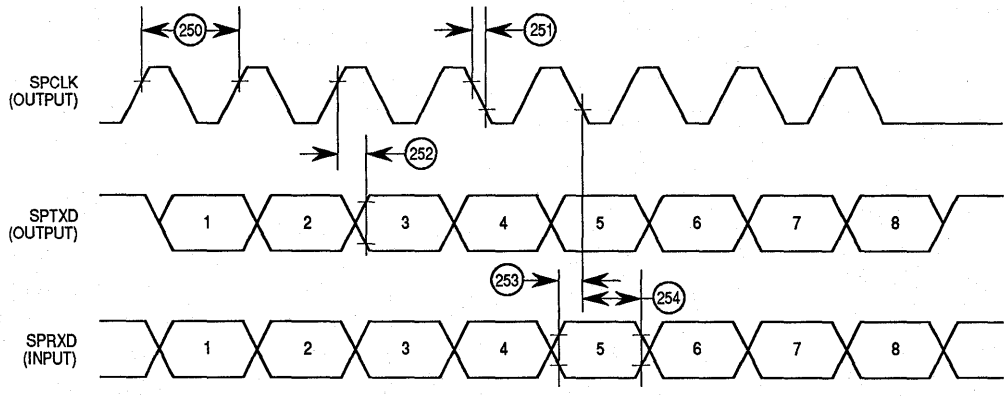


Figure 6-17. Serial Communication Port Timing Diagram

6

6.19 AC ELECTRICAL SPECIFICATIONS — IDL TIMING (All timing measurements, unless, otherwise specified, are referenced to the L1CLK at 50% point of V_{DD}) (see Figure 6-18)

Num.	Characteristic	16.67 MHz		Unit
		Min	Max	
260	L1CLK (IDL Clock) Frequency (see Note 1)	—	6.66	MHz
261	L1CLK Width Low	55	—	ns
262	L1CLK Width High	55	—	ns
263	L1TXD, L1RQ, SDS1–SDS2 Rising/Falling Time	—	20	ns
264	L1SY1 (sync) Setup Time (to L1CLK Falling Edge)	30	—	ns
265	L1SY1 (sync) Hold Time (from L1CLK Falling Edge)	50	—	ns
266	L1SY1 (sync) Inactive Before 4th L1CLK	0	—	ns
267	L1TxD Active Delay (from L1CLK Rising Edge)	0	90	ns
268	L1TxD to High Impedance (from L1CLK Rising Edge) (see Note 2)	0	50	ns
269	L1RxD Setup Time (to L1CLK Falling Edge)	50	—	ns
270	L1RxD Hold Time (from L1CLK Falling Edge)	50	—	ns
271	Time Between Successive IDL syncs	20	—	L1CLK
272	L1RQ Valid before Falling Edge of L1SY1	1	—	L1CLK
273	L1GR Setup Time (to L1SY1 Falling Edge)	50	—	ns
274	L1GR Hold Time (from L1SY1 Falling Edge)	50	—	ns
275	SDS1–SDS2 Active Delay from L1CLK Rising Edge	10	90	ns
276	SDS1–SDS2 Inactive Delay from L1CLK Falling Edge	10	90	ns

NOTES:
1. The ratio CLK/L1CLK must be greater than 2.5/1.
2. High impedance is measured at the 30% and 70% of V_{DD} points, with the line at $V_{DD}/2$ through 10K in parallel with 130 pF.

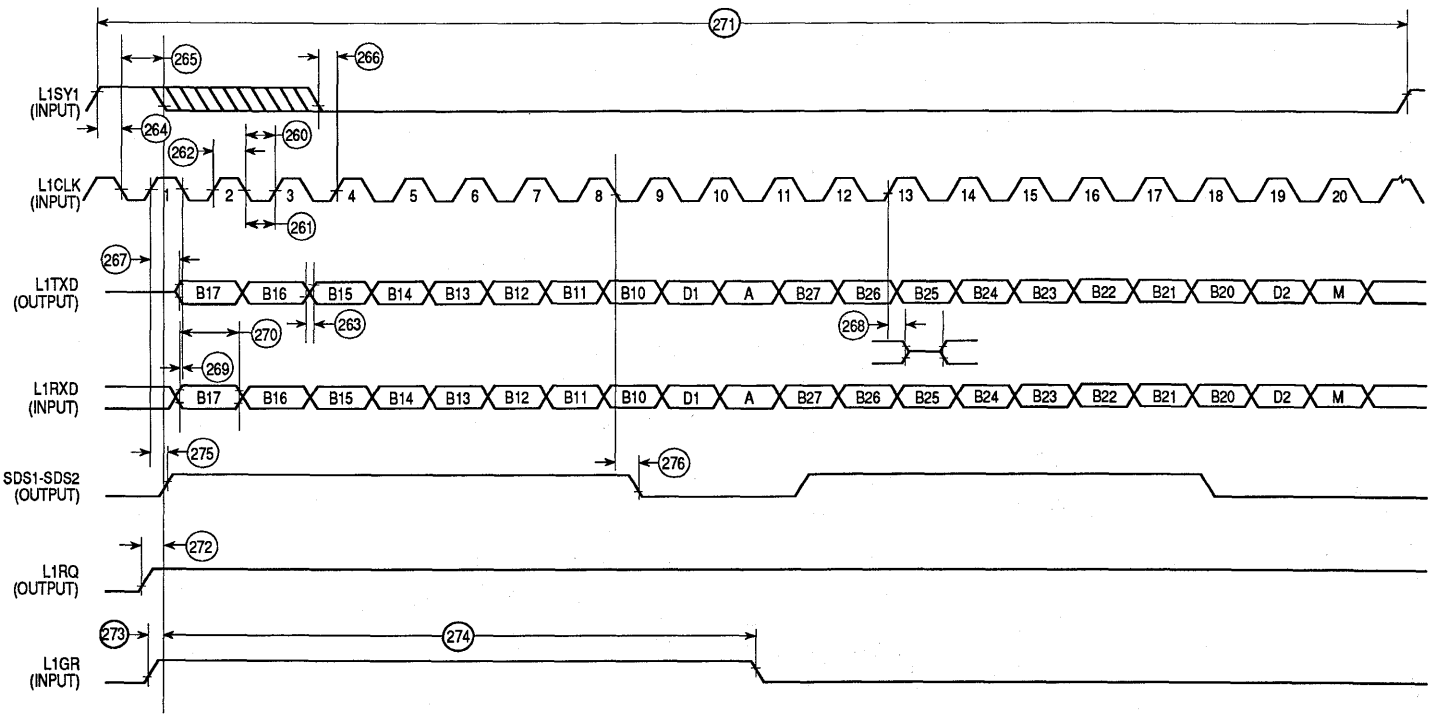


Figure 6-18. IDL Timing Diagram

6.20 AC ELECTRICAL SPECIFICATIONS — GCI TIMING

GCI supports the NORMAL mode and the GCI channel 0 (GCN0) in MUX mode.
Normal mode uses 512 kHz clock rate (256K bit rate).
MUX mode uses 256 × n − 3088 kbs (clock rate is data rate × 2).
The ratio CLK/L1CLK must be greater than 2.5/1.
(see Figure 6-19)

Num.	Characteristic	16.67 MHz		Unit
		Min	Max	
	L1CLK GCI Clock Frequency (Normal Mode) (see Note 1)	—	512	kHz
280	L1CLK Clock Period Normal Mode (see Note 1)	1800	2100	ns
281	L1CLK Width Low/High Normal Mode	840	1450	ns
282	L1CLK Rise/Fall Time Normal Mode (see Note 4)	—	—	ns
	L1CLK (GCI Clock) Period (MUX Mode) (see Note 1)	—	6.668	MHz
280	L1CLK Clock Period MUX Mode (see Note 1)	150	—	ns
281	L1CLK Width Low/High MUX Mode	55	—	ns
282	L1CLK Rise/Fall Time MUX Mode (see Note 4)	—	—	ns
283	L1SY1 Sync Setup Time to L1CLK Falling Edge	30	—	ns
284	L1SY1 Sync Hold Time from L1CLK Falling Edge	50	—	ns
285	L1TxD Active Delay (from L1CLK Rising Edge) (see Note 2)	0	100	ns
286	L1TxD Active Delay (from L1SY1 Rising Edge) (see Note 2)	0	100	ns
287	L1RxD Setup Time to L1CLK Rising Edge	20	—	ns
288	L1RxD Hold Time from L1CLK Rising Edge	50	—	ns
289	Time Between Successive L1SY1 in Normal Mode SCIT Mode	64 192	— —	L1CLK L1CLK
290	SDS1–SDS2 Active Delay from L1CLK Rising Edge (see Note 3)	10	90	ns
291	SDS1–SDS2 Active Delay from L1SY1 Rising Edge (see Note 3)	10	90	ns
292	SDS1–SDS2 Inactive Delay from L1CLK Falling Edge	10	90	ns
293	GCIDCL (GCI Data Clock) Active Delay	0	50	ns

NOTES:
1. The ratio CLK/L1CLK must be greater than 2.5/1.
2. Condition $C_L = 150$ pF
   L1TxD becomes valid after the L1CLK rising edge or L1SY1, whichever is later.
3. SDS1–SDS2 become valid after the L1CLK rising edge or L1SY1, whichever is later.
4. Schmitt trigger used on input buffer.

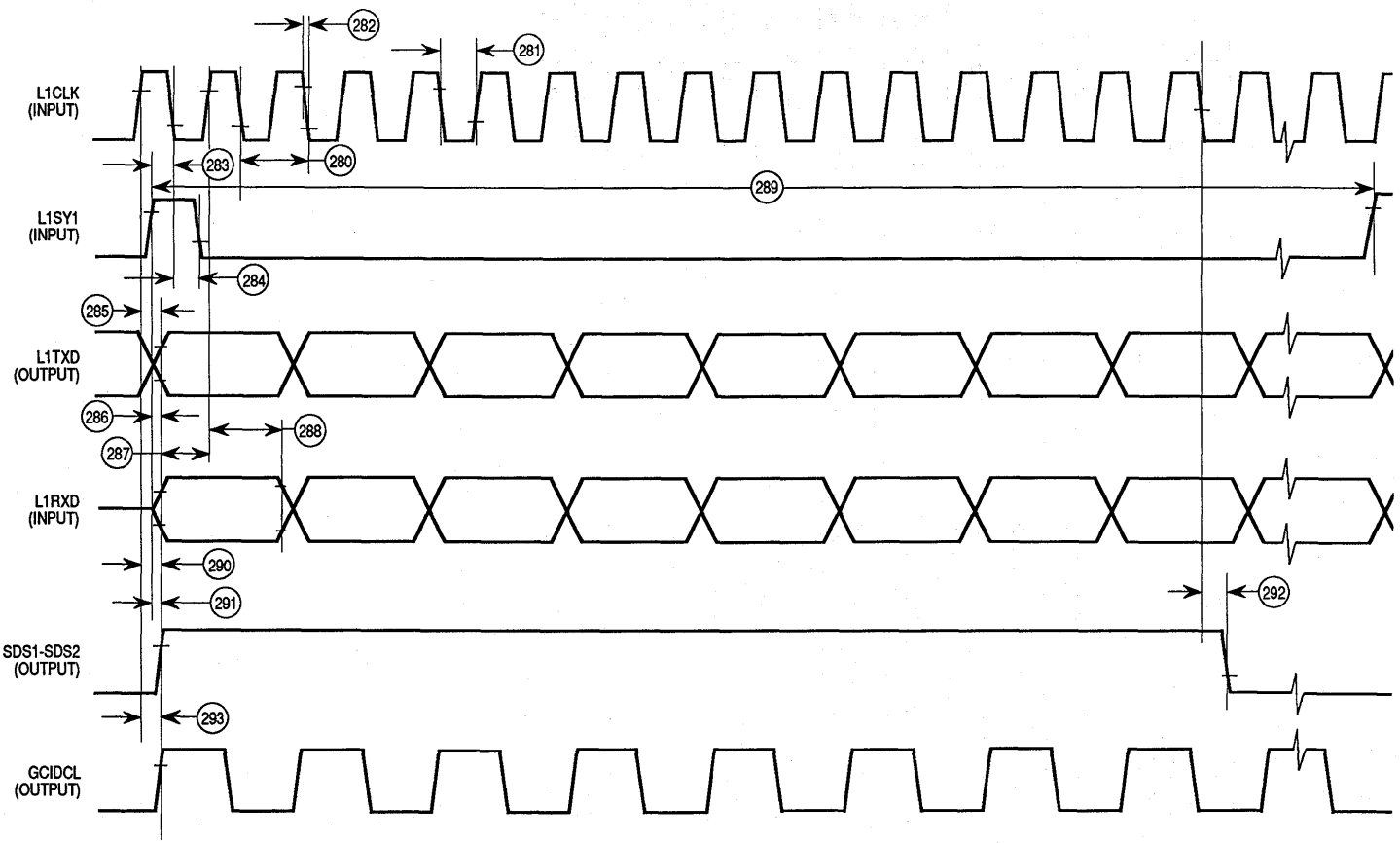


Figure 6-19. GCI Timing Diagram

6.21 AC ELECTRICAL SPECIFICATIONS — PCM TIMING

There are two syncs types:
Short Frame — Sync signals are one clock cycle prior to the data
Long Frame — Sync signals are N-bits that envelope the data, N>0
(see Figure 6-20)

Num.	Characteristic	16.67 MHz		Unit
		Min	Max	
300	L1CLK (PCM Clock) Frequency (see Note 1)	—	6.66	MHz
301	L1CLK Width Low/High	55	—	ns
302	L1SY0–L1SY1 Setup Time to L1CLK Falling Edge	20	—	ns
303	L1SY0–L1SY1 Hold Time from L1CLK Falling Edge	40	—	ns
304	L1SY0–L1SY1 Width Low	1	—	L1CLK
305	Time Between Successive Sync Signals (Short Frame)	8	—	L1CLK
306	L1TxD Data Valid after L1CLK Rising Edge (see Note 2)	0	100	ns
307	L1TxD to High Impedance (from L1CLK Rising Edge)	0	70	ns
308	L1RxD Setup Time (to L1CLK Falling Edge) (see Note 3)	20	—	ns
309	L1RxD Hold Time (from L1CLK Falling Edge) (see Note 3)	50	—	ns
310	L1TxD Data Valid After Syncs Rising Edge (Long) (see Note 2)	0	100	ns
311	L1TxD to High Impedance (from L1SY0–L1SY1 Falling Edge) (Long)	0	70	ns

NOTES:
1. The ratio CLK/L1CLK must be greater than 2.5/1.
2. L1TxD becomes valid after the L1CLK rising edge or the sync enable, whichever is later, if long frames are used.
3. Specification valid for both sync methods.

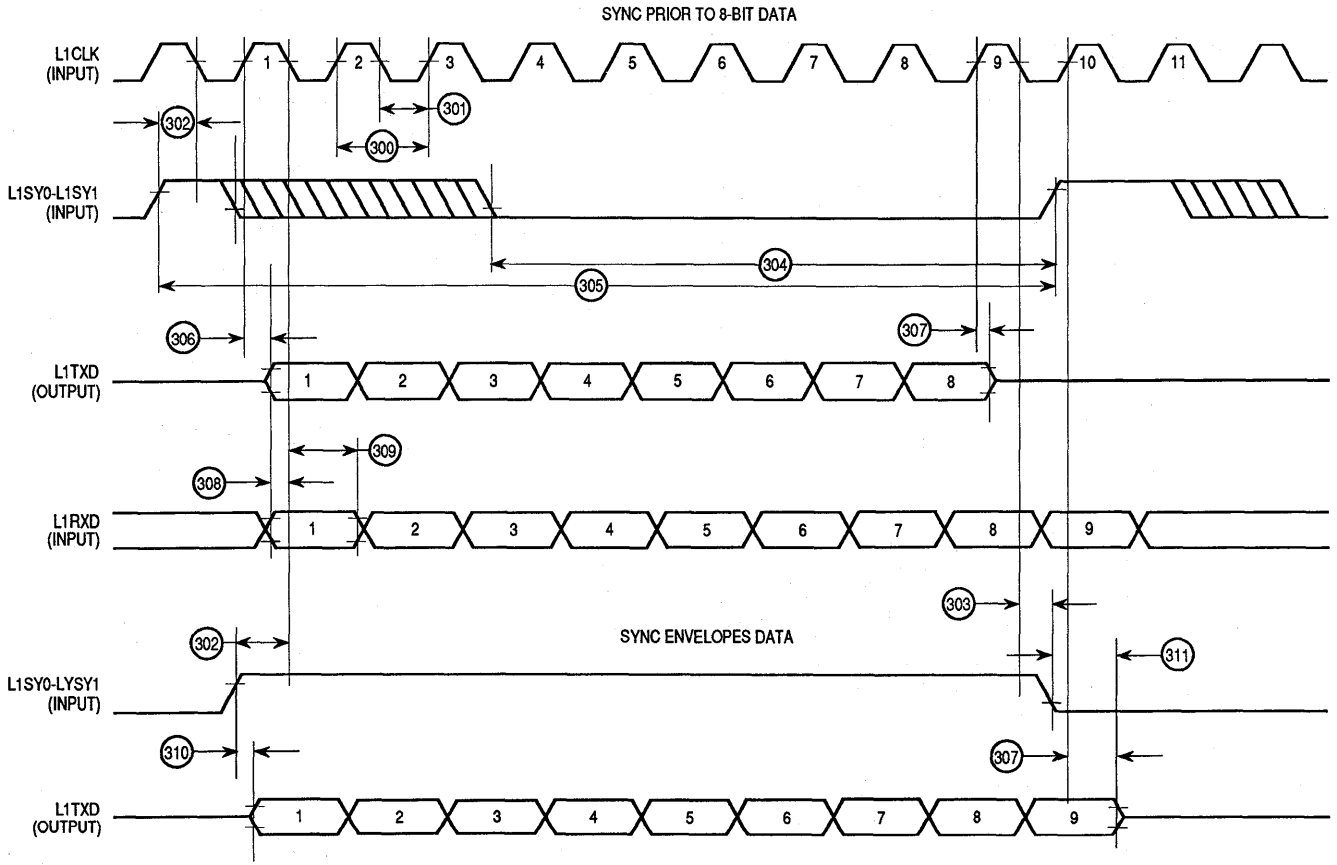


Figure 6-20. PCM Timing Diagram

6.22 AC ELECTRICAL SPECIFICATIONS — NMSI TIMING

The NMSI mode uses two clocks, one for receive and one for transmit. Both clocks can be internal or external. When the clock is internal, it is generated by the internal baud rate generator and it is output on L1RXD or L1TXD. All the timing is related to the external clock pin. The timing is specified for NMSI1. It is also valid for NMSI2 and NMSI3.
(see Figure 6-21)

Num.	Characteristic	Internal Clock		External Clock		Unit
		Min	Max	Min	Max	
315	RCLK1 and TCLK1 Frequency (see Note 1)	—	5.12	—	6.668	MHz
316	RCLK1 and TCLK1 Low/High	70	—	55	—	ns
317	RCLK1 and TCLK1 Rise/Fall Time (see Note 3)	—	—	—	—	ns
318	TxD1 Active Delay from TCLK1 Falling Edge	0	40	0	70	ns
319	$\overline{\text{RTS1}}$ Active/Inactive Delay from TCLK1 Falling Edge	0	40	0	100	ns
320	$\overline{\text{CTS1}}$ Setup Time to TCLK1 Rising Edge	50	—	10	—	ns
321	RXD1 Setup Time to RCLK1 Rising Edge	50	—	10	—	ns
322	RXD1 Hold Time from RCLK1 Rising Edge (see Note 2)	10	—	50	—	ns
323	$\overline{\text{CD1}}$ Setup Time to RCLK1 Rising Edge	50	—	10	—	ns

NOTE:
1. The ratio CLK/TCLK1 and CLK/RCLK1 must be greater than 2.5/1 for external clock.
   For internal clock the ratio must be greater than 3/1 (the input clock to the baud rate generator may be either CLK or TIN1), in both cases the maximum frequency is limited to 16.67 MHz.
   In asynchronous mode (UART), the bit rate is 1/16 of the clock rate.
2. Also applies to $\overline{\text{CD}}$ hold time when $\overline{\text{CD}}$ is used as an external sync in BISYNC or totally transparent mode.
3. Schmitt triggers used on input buffers.

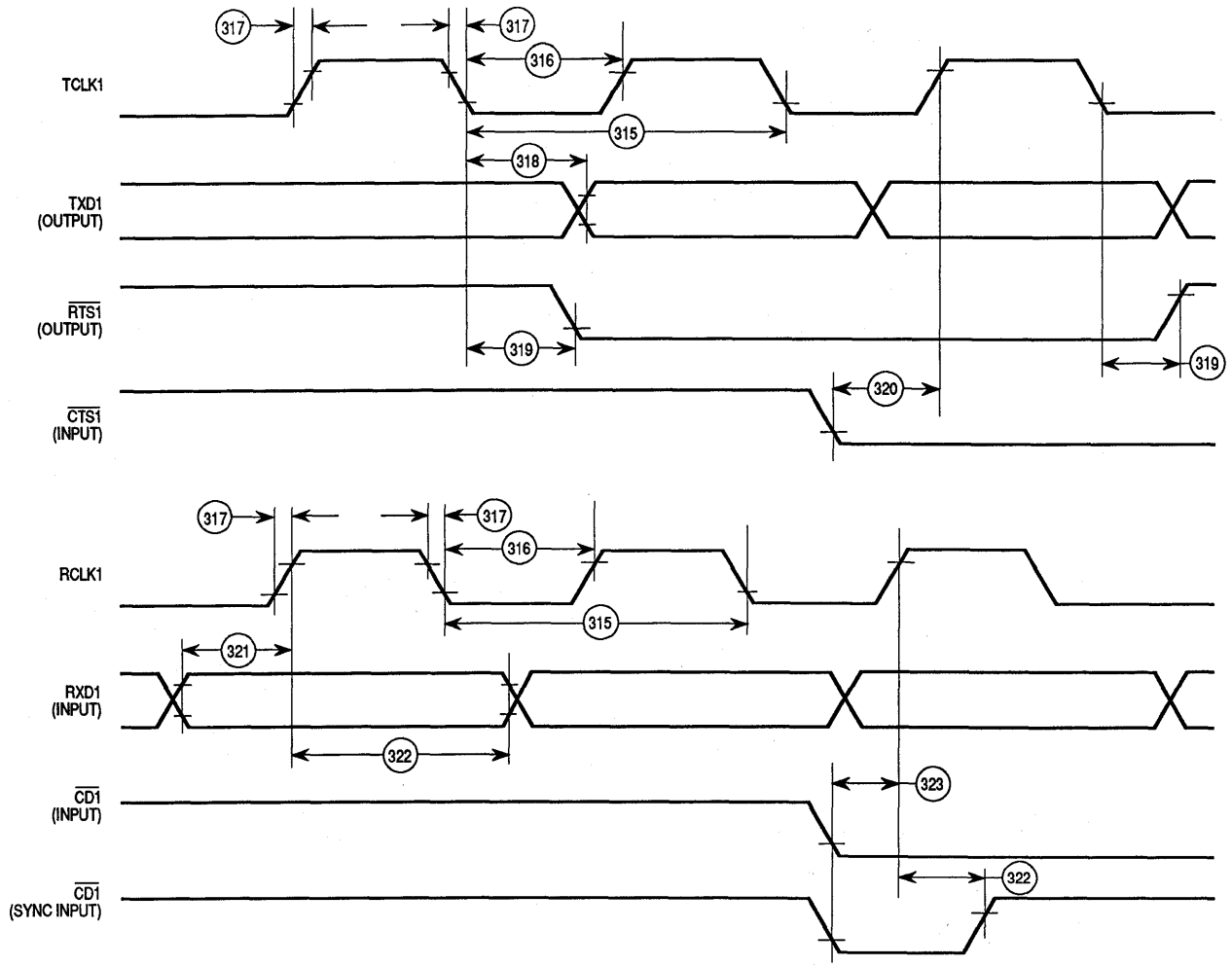
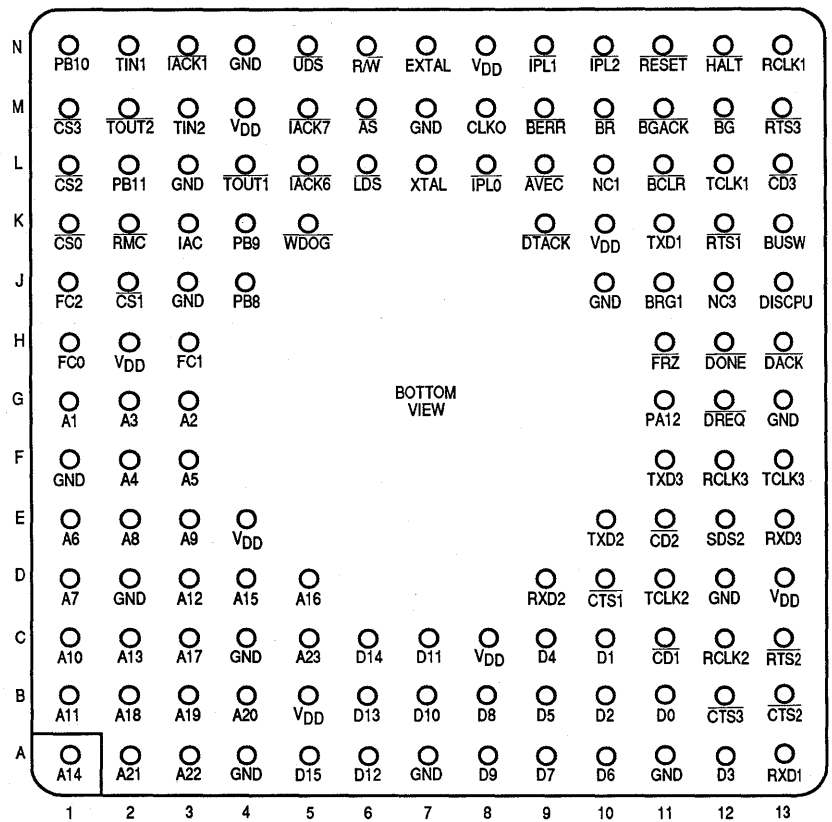


Figure 6-21. NMSI Timing Diagram

SECTION 7
MECHANICAL DATA AND ORDERING INFORMATION

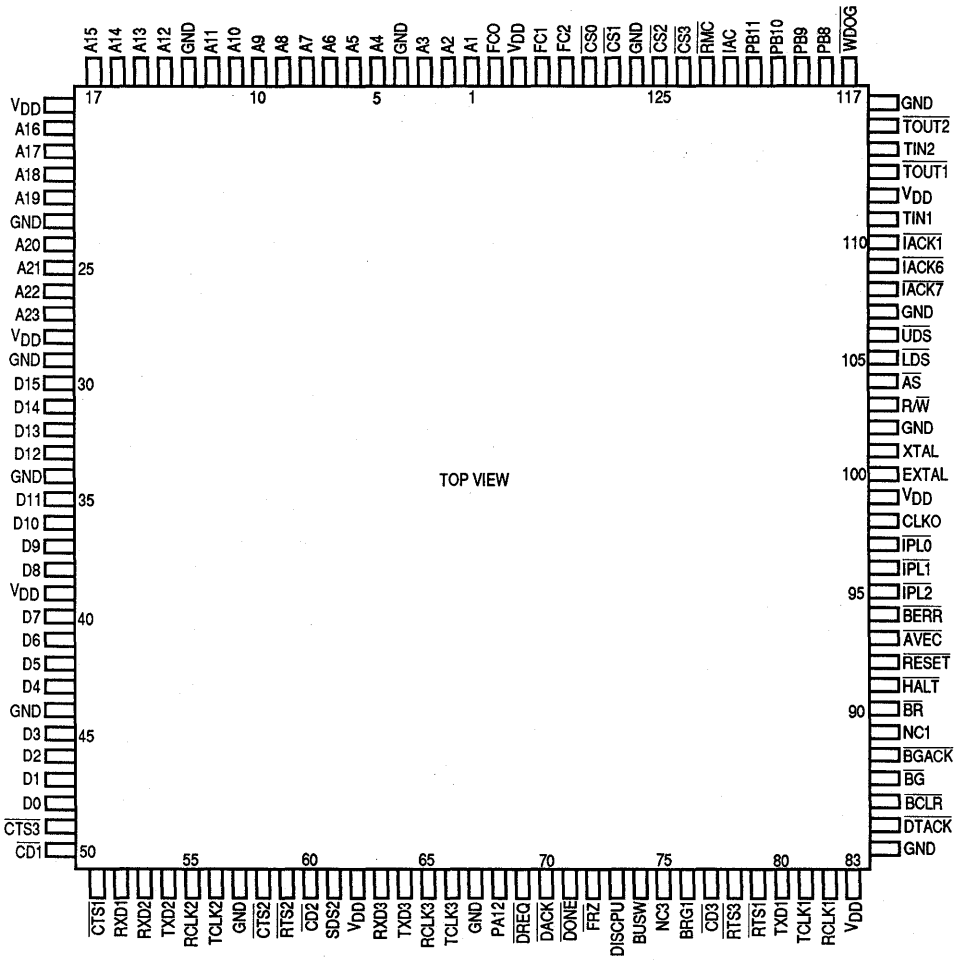
7.1 PIN ASSIGNMENTS

7.1.1 Pin Grid Array



7

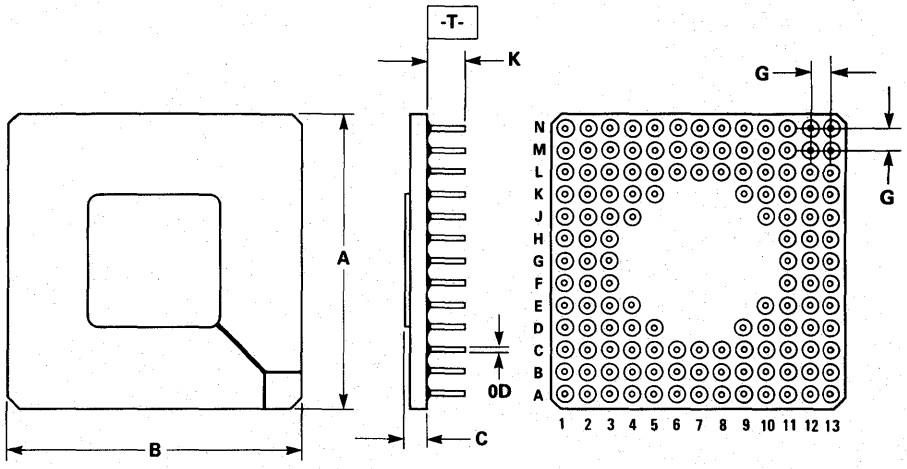
7.1.2 Ceramic Surface Mount (CQFP)



7

7.2 PACKAGE DIMENSIONS

RC SUFFIX
PIN GRID ARRAY
CASE 789B-01



NOTES:
1. A AND B ARE DATUMS AND T IS A DATUM SURFACE.
2. POSITIONAL TOLERANCE FOR LEADS (132 PL).

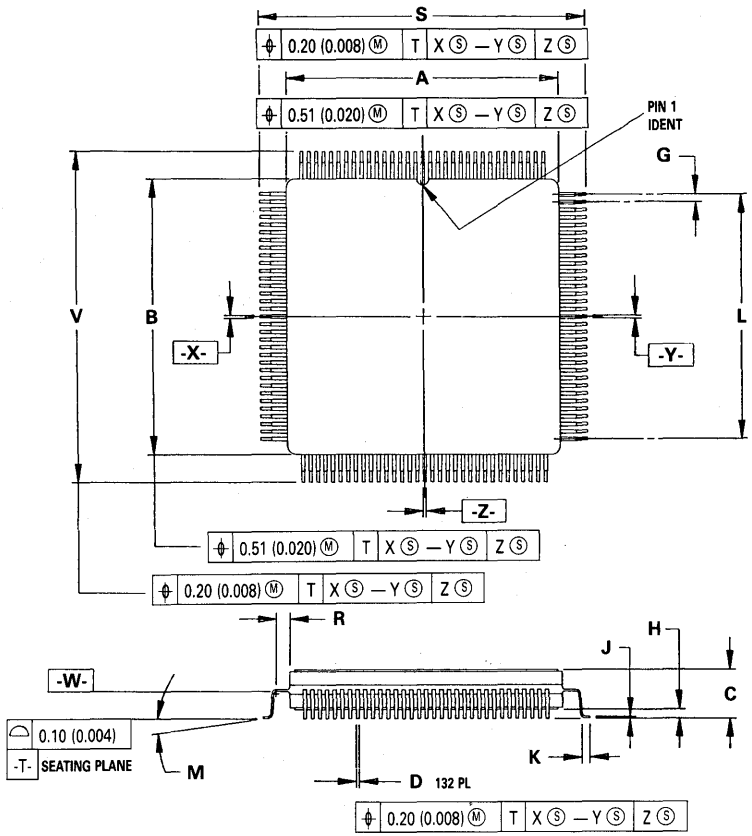
⌖	ϕ 0.13 (0.005) Ⓜ	T	A Ⓢ	B Ⓢ

3. DIMENSIONING AND TOLERANCING PER Y14.5M, 1982.
4. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	34.04	35.05	1.340	1.380
B	34.04	35.05	1.340	1.380
C	2.54	3.81	0.100	0.150
D	0.43	0.55	0.017	0.022
G	2.54 BSC		0.100 BSC	
K	4.32	4.95	0.170	0.195

7

FE SUFFIX
CERAMIC SURFACE MOUNT
CASE 831-01



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	21.85	22.86	0.860	0.900
B	21.85	22.86	0.860	0.900
C	3.94	4.31	0.155	0.170
D	0.204	0.292	0.0080	0.0115
G	0.64 BSC		0.025 BSC	
H	0.64	0.88	0.025	0.035
J	0.13	0.20	0.005	0.008
K	0.51	0.76	0.020	0.030
L	20.32 REF		0.800 REF	
M	0°	8°	0°	8°
R	0.64	—	0.025	—
S	27.31	27.55	1.075	1.085
V	27.31	27.55	1.075	1.085

NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIM A AND B DEFINE MAXIMUM CERAMIC BODY DIMENSIONS INCLUDING GLASS PROTRUSION AND MISMATCH OF CERAMIC BODY TOP AND BOTTOM.
4. DATUM PLANE -W- IS LOCATED AT THE UNDERSIDE OF LEADS WHERE LEADS EXIT PACKAGE BODY.
5. DATUMS X–Y AND Z TO BE DETERMINED WHERE CENTER LEADS EXIT PACKAGE BODY AT DATUM -W-.
6. DIM S AND V TO BE DETERMINED AT SEATING PLANE, DATUM -T-.
7. DIM A AND B TO BE DETERMINED AT DATUM PLANE -W-.

7.3 ORDERING INFORMATION

Package Type	Frequency (MHz)	Temperature	Order Number
Pin Grid Array (RC Suffix)	16.67 16.67	0°C to 70°C 0°C to 85°C	MC68302RC16 MC68302IRC16
Ceramic Surface Mount (FE Suffix)	16.67 16.67	0°C to 70°C 0°C to 85°C	MC68302FE16 MC68302IFE16

7

APPENDIX A
SCC PERFORMANCE

The MC68302 at 16.67 MHz was designed to support unrestricted operation of multiple (three) serial communications controllers (SCCs) servicing differing communications protocols with cost-effective usage of silicon at data rates of 256 kbps for HDLC-framed or transparent data handling and 128 kbps for BISYNC, DDCMP, V.110, or asynchronous framed data. The resultant design is well able to support these design goals.

Since the MC68302 serial channels will likely service serial channels time-division multiplexed into higher bandwidth channels, such as the U.S. and Japanese T1 and European CEPT primary rate channels, the physical clocking of the serial channels can be accomplished at the higher speeds required by these channels — up to a maximum of 40% (a 1:2.5 ratio) of the system clock frequency. This gives up to a 6.67-MHz serial clock for a 16.67-MHz IMP system clock. At this same 16.67-MHz system-clock speed, the MC68302 can therefore handle the 1.544-MHz and 2.048-MHz clocking frequencies of T1 and CEPT lines as well as the 4.096-Mbps signaling rates of the common ISDN interchip local buses, such as IDL and GCI (also known as IOM2).

Thus, the MC68302 is well equipped to handle, for instance, three 256-kbps channels multiplexed on a T1 or CEPT primary rate channel. The limitation for a given channel becomes the number of contiguous 8-bit subchannels that are supported. With T1/CEPT, for example, the limitation is four 64-kbps subchannels giving 256 kbps per SCC.

Where an application requires even higher bandwidth channels, such as the 384-kbps H0 channels, the 1.536-Mbps H11 channel or higher, the following "restricted" operation should be observed:

1. Receive buffers must be equal to, or longer than, the maximum frame length — i.e., a single buffer per frame.
2. Bus latency of the SDMA must be less than 20 system clocks. (The $\overline{\text{BCLR}}$ signal may be helpful here.)
3. Bus cycles, including arbitration, must be less than or equal to 10 clock cycles, which implies no more than two wait states in system memory accesses.

If the above restrictions are adhered to, the following table shows sample performance obtainable with the device. These results apply to the revision B devices (mask 2B14M). Performance of revision A devices (mask 1B14M) may be slightly less.

The frequency ratio stated represents the system (EXTAL) frequency to serial bit rate frequency. A user exceeding this bit rate will begin to experience SCC underruns and/or overruns. Some users may wish to tolerate an occasional underrun/overrun to slightly increase performance.

For example, a ratio of 1:10 shows that a system clock of 16.67 MHz can support a serial rate of 1.67 Mbps. Typically, this 1.67-Mbps rate would be achieved with 1 bit every 1.67-MHz serial clock. However, it may also be achieved with a faster serial clock (subject to the clocking limits of the SCC mentioned previously) as long as the bit rate over a 9-bit (17-bit period for HDLC or transparent) period averages out to 1.67 Mbps.

High-Speed Channels		Low-Speed Channels		Comments/Restrictions
Number	Frequency Ratio	Number	Frequency Ratio	
1 HDLC	1:8	—	—	Buffers in Dual-Port RAM
1 HDLC	1:9	—	—	
1 HDLC	1:10	2	1:156	Any Protocol (see Note 2).
2 HDLC	1:19	—	—	
3 HDLC	1:28	—	—	
3 UART	1:2.5*16	—	—	UART Clock is 16×Bit Rate
3 BISYNC	1:51	—	—	Normal Half-Duplex Assumed
3 DDCMP	1:72	—	—	

NOTES:
1. All performance data is preliminary.
2. With a 16.67-MHz clock and HDLC operating at 1.544 Mbps, the two low-speed channels may run at 128 kbps.
3. All results assume the DRAM refresh controller is not operating; otherwise, performance is slightly reduced.
4. Unless specifically stated, all results assume full-duplex operation. Results for half-duplex operation will be 1.5 to 2×better.
5. In revision B devices, promiscuous (totally transparent) performance is the same as that for HDLC and can be substituted for all uses of HDLC above. In revision A devices, a single transparent channel can operate with a ratio of 1:33.

A

Since operation at high data rates is characteristic of HDLC-framed channels rather than BISYNC-, DDCMP-, or async-framed channels, the user can also use the MC68302 in conjunction with either the Motorola MC68605 1984 CCITT X.25 LAPB controller, the MC68606 CCITT Q.921 Multilink LAPD controller, or the MC145488 dual data link controller. These devices fully support operation at T1/CEPT rates (and above) and can operate with their serial

clocks "gated" onto subchannels of such an interface. These devices are full M68000 bus masters. The MC68605 and MC68606 perform the full datalink layer protocol as well as support various transparent modes within HDLC-framed operation. The MC145488 provides HDLC-framed and totally transparent operation on two full-duplex channels.

A

APPENDIX B
DEVELOPMENT TOOLS AND SUPPORT

B.1 SOFTWARE OVERVIEW

The software development package is offered as a set of independent modules which provide the following features:

- IMP Chip Evaluation
  By running the modules on the development board described in **B.4 ADS302 DEVELOPMENT SYSTEM**, it is possible to examine and evaluate the MC68302. Symbolic, user-friendly menus allow control of all parts of the IMP.
- IMP Chip Drivers
  Written in C, the source code of the IMP drivers is available as a document or on electronic media.
- Protocol Implementations
  Modules implementing common ISO/OSI layer 2 and 3 protocols are available under license. These are in the form of binary relocatable object code as well as source code written in C.
- Portability
  All of the modules may be easily ported to different MC68302 implementations and combined with user-developed code. Well-defined software interface documentation is available for all provided modules.

B.2 SOFTWARE MODULES

Since the IMP is a memory-mapped device based on a full M68000 core, existing compilers, assemblers, and linkers designed for the M68000 Family may be successfully used. The memory address space for the IMP is relocatable (through the base address register) to any 4K-byte contiguous space in the M68000 processor memory map, allowing the IMP registers to be mapped to any location in system RAM.

Chip driver routines written in C illustrate initialization of the IMP, interrupt handling, and the management of data transmission and reception on all channels.

Modules of software are also available for evaluation/debugging purposes and the execution of common protocols. All modules communicate with each other using message passing. This technique simplifies the interfaces between the different modules and enables them to interface with other user-added modules with relative ease. Detailed descriptions of the software interfaces are available.

Each module operates completely independent of its environment. Independence from the hardware environment is achieved by the fact that each module is individually configurable and relocatable anywhere in system memory. Calls are used for requesting resources or services from the host operating system (memory management and message passing), which creates independence from the firmware. Modules may be used with any combination of other modules, including those added by the user.

Modules for the layer-2 protocols, LAPD and LAPB, and layer-3 protocol, X.25, are initially provided. This library will be expanded to include X.3 (PAD), Q.931, V.110, V.120, and support for the BISYNC and DDCMP upper layers.

The user-interface module provides a menu-driven user interface to the IMP and each functional protocol module. This module may be used for chip evaluation or as a tool for debugging user-developed applications. Menu options will allow a user to examine the appropriate module's memory structures (or the register set and on-chip dual-port RAM of the IMP) or to issue specific commands. The commands may result in specific IMP commands or in the execution of protocol-defined primitives in one of the protocol modules.

All modules (protocol, driver, and user-interface modules) are available in both object code on the ADS302 board or C source code.

B.3 IN-CIRCUIT EMULATION SUPPORT

Full in-circuit emulation support is available but is not discussed in this manual.

B.4 ADS302 DEVELOPMENT SYSTEM

The application development system (ADS) 302 board is a standalone board developed by Motorola that includes software modules, a board-level real-time kernel, and a monitor/debugger. It is a flexible yet lower cost alternative to full in-circuit emulation. It also provides an immediate platform in which to begin testing software.

The ADS302 board consists of the MC68302, memory (512K bytes of RAM expandable to 1M byte, 256K bytes of EPROM, and EEPROM) and an MC68681 DUART (to allow all MC68302 serial ports to be left available to the user). Both a serial (RS-232) and a parallel external interface are provided for connecting a dumb terminal and/or a host computer, respectively (see Figure B-1).

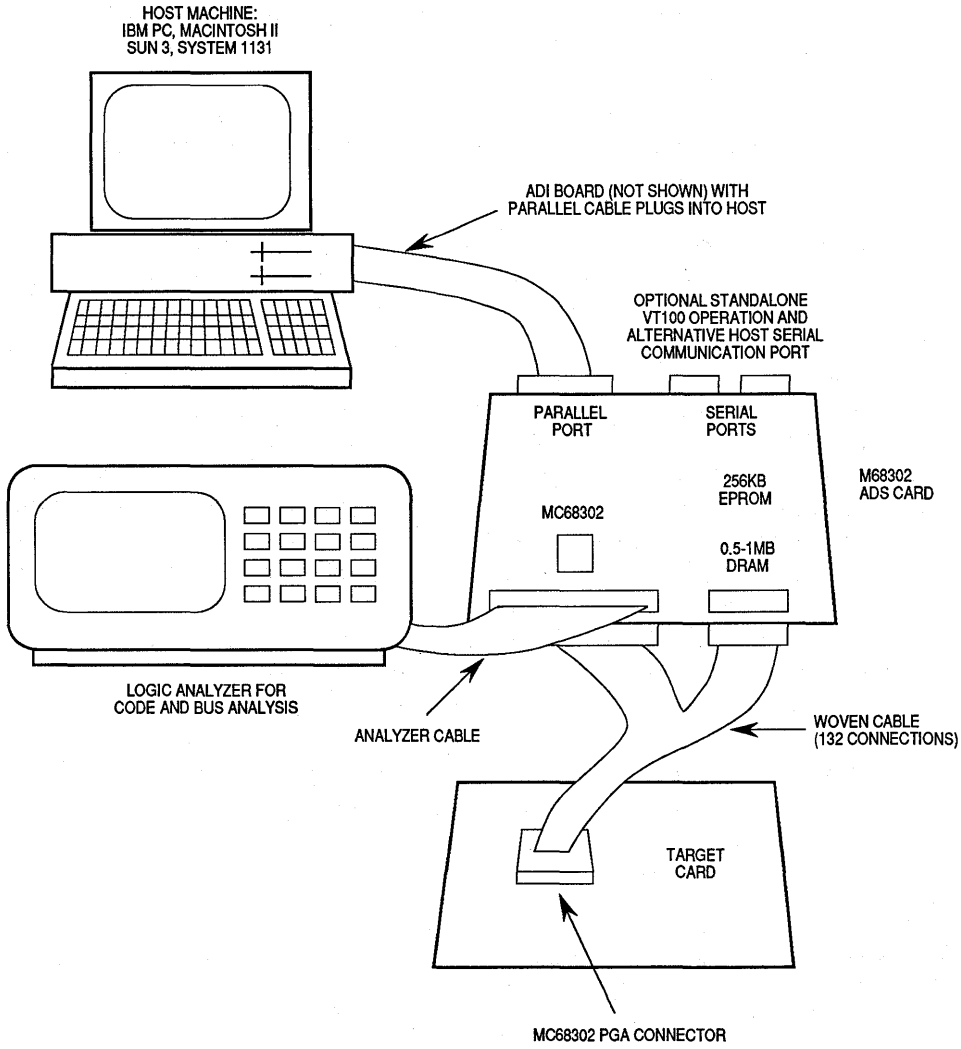


Figure B-1. ADS302 Development System Board

B

To assist with target-board development, the ADS302 board may be connected to the target board using a specially supplied, impedance-matched target cable that carries all IMP signals to a 132 PGA connector. (Adaptors from PGA to CQFP and PQFP are available from third parties.) Prototype boards may therefore make use of signals available from the ADS302 development board. Target board execution may be carried out with the MC68302 executing from the ADS302 board memory, target board memory, or a combination of both.

The ADS302 board also contains connections for use by a logic analyzer, allowing all MC68302 pins to be examined. Connectors between the ADS302 board and common logic analyzers are available from third parties. A logic analyzer output trigger signal may also be connected to the MC68302 freeze ($\overline{FRZ}$) pin to halt on-chip peripherals, or to the interrupt controller to generate a nonmaskable interrupt.

To connect the ADS302 board to a host computer, a parallel interface cable and host interface (ADI) board are supplied. Hosts include the Macintosh™ II, SUN-3™, Motorola 1131, and the IBM™ PC. Available software executing on the host, allows uploading and downloading of files and data between the host and the ADS302 board, and control of the user interface module through a menu-driven interface. Source-level debugging support through this interface and the serial interface is also provided by third-party vendors.

To serve as a convenient platform for software development, the ADS302 board is supplied together with a real-time kernel and the 302bug monitor/debugger. The real-time kernel (EDX) offers basic operating system services such as multitasking and memory management. The monitor/debugger provides operations of memory dump and set (with optional assembly/disassembly of M68000 instructions), single instruction execution, breakpoints, and downloads over the serial and parallel interfaces.

Additionally, the ADS302 board contains the software modules (protocol, driver, and user-interface modules) in EPROM.

B

Macintosh is a trademark of Apple Computer, Inc.
SUN-3 is a trademark of Sun Microsystems, Inc.
IBM is a trademark of International Business Machines.

APPENDIX C
RISC MICROCODE FROM RAM

The MC68302 RISC processor has an option to execute microcode from the 576-byte user RAM in the on-chip dual-port RAM. In this mode, the 576-byte user RAM cannot be accessed by the M68000 core or other M68000 bus master.

Once the microcode has been loaded into the dual-port RAM by the M68000 core or other bus master, the RAM from microcode option is enabled in the 32-bit reserved register at location $0F8. If bit 16 of this register is set, the RISC processor will execute microcode from RAM once the CP is reset in the command register. At this time, the RISC processor can freely address both the dual-port RAM and its own private ROM.

Microcode for a new application or protocol is developed only under special arrangement and coordination with Motorola.

Some RAM microcode routines are also available for purchase. The following is an overview of available routines. Contact a Motorola sales office for detailed specifications of the microcode routines.

C.1 SS7 PROTOCOL SUPPORT

The HDLC routines are enhanced and extended to provide special Signalling System #7 support. In addition to the HDLC features, the following key features are included:

- Automatic Discard of Short (less than 5 octets) Signal Units
- Automatic Fill-In Signal Unit (FISU) Transmission and Reception
- Automatic Link Status Signal Unit (LSSU) Retransmission
- Octet Counting Mode Support

C

C.2 CENTRONIX™ PARALLEL INTERFACE SUPPORT

The RISC processor implements a Centronix parallel interface using the PA7–PA0, PA12–PA8, and the PA15 parallel I/O pins. All SCC channels and protocols, including DRAM refresh, are supported concurrently with this feature, but SCC2, if used, must be configured in a multiplexed mode, and several of its buffer descriptors are used by the Centronix interface. Once a buffer is configured in memory, the protocol handles the movement of all data from the buffer to the interface.

Centronix is a trademark of Centronix.

C

A24641-3 PRINTED IN USA 3/91 GTE 14019 5000 MPU YGAAAA