



Welcome to [E-XFL.COM](https://www.e-xfl.com)

### Understanding [Embedded - Microprocessors](#)

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

### Applications of [Embedded - Microprocessors](#)

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

#### Details

Product Status	Obsolete
Core Processor	M68000
Number of Cores/Bus Width	1 Core, 8/16-Bit
Speed	16MHz
Co-Processors/DSP	Communications; RISC CPM
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	3.3V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68302pv16vc">https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68302pv16vc</a>

Paragraph Number	Title	Page Number
4.4.5	Serial Interface Registers.....	4-19
4.4.5.1	Serial Interface Mode Register (SIMODE) .....	4-19
4.4.5.2	Serial Interface Mask Register (SIMASK) .....	4-22
4.5	Serial Communication Controllers (SCCs).....	4-22
4.5.1	SCC Features .....	4-24
4.5.2	SCC Configuration Register (SCON) .....	4-24
4.5.2.1	Asynchronous Baud Rate Generator Examples .....	4-26
4.5.2.2	Synchronous Baud Rate Generator Examples .....	4-27
4.5.3	SCC Mode Register (SCM).....	4-27
4.5.4	SCC Data Synchronization Register (DSR).....	4-31
4.5.5	Buffer Descriptors Table .....	4-32
4.5.6	SCC Parameter RAM Memory Map.....	4-34
4.5.6.1	Data Buffer Function Code Register (TFCR, RFCR) .....	4-35
4.5.6.2	Maximum Receive Buffer Length Register (MRBLR) .....	4-36
4.5.6.3	Receiver Buffer Descriptor Number (RBD#) .....	4-36
4.5.6.4	Transmit Buffer Descriptor Number (TBD#).....	4-36
4.5.6.5	Other General Parameters.....	4-37
4.5.7	SCC Initialization.....	4-37
4.5.8	Interrupt Mechanism .....	4-38
4.5.8.1	SCC Event Register (SCCE) .....	4-38
4.5.8.2	SCC Mask Register (SCCM) .....	4-39
4.5.8.3	SCC Status Register (SCCs) .....	4-39
4.5.8.4	Bus Error on SDMA Access.....	4-40
4.5.9	SCC Transparent Mode .....	4-41
4.5.10	Disabling the SCCs.....	4-42
4.5.11	UART Controller.....	4-43
4.5.11.1	Normal Asynchronous Mode.....	4-45
4.5.11.2	Asynchronous DDCMP MODE .....	4-46
4.5.11.3	UART Memory Map .....	4-46
4.5.11.4	UART Programming Model.....	4-48
4.5.11.5	UART Command Set .....	4-49
4.5.11.6	UART Address Recognition .....	4-50
4.5.11.7	UART Control Characters and Flow Control.....	4-51
4.5.11.8	Send Break .....	4-53
4.5.11.9	Send Preamble (IDLE) .....	4-53
4.5.11.10	Wakeup Timer.....	4-53
4.5.11.11	UART Error-Handling Procedure .....	4-54
4.5.11.12	Fractional Stop Bits.....	4-55
4.5.11.13	UART Mode Register.....	4-56
4.5.11.14	UART Receive Buffer Descriptor (Rx BD) .....	4-57
4.5.11.15	UART Transmit Buffer Descriptor (Tx BD).....	4-61
4.5.11.16	UART Event Register.....	4-63
4.5.11.17	UART MASK Register.....	4-65
4.5.11.18	S-Records Programming Example .....	4-65
4.5.12	HDLC Controller.....	4-66

bus cycle after the completion of the current instruction.

3. The interrupt controller recognizes the interrupt acknowledge cycle and places the interrupt vector for that interrupt request onto the M68000 bus.
4. The M68000 reads the vector, reads the address of the interrupt handler in the exception vector table, and then begins execution at that address.

Steps 2 and 4 are the responsibility of the M68000 core on the IMP; whereas, steps 1 and 3 are the responsibility of the interrupt controller on the IMP.

The M68000 core is not modified on the IMP; thus, steps 2 and 4 operate exactly as they would on the MC68000. In step 2, the M68000 status register (SR) is available to mask interrupts globally or to determine which priority levels can currently generate interrupts (see 2.5 Interrupt Processing for more details). Also in step 2, the interrupt acknowledge cycle is executed.

The interrupt acknowledge cycle carries out a standard M68000 bus read cycle, except that FC2–FC0 are encoded as 111, A3–A1 are encoded with the interrupt priority level (1–7, with 7 (i.e., 111) being the highest), and A19–A16 are driven high.  $\overline{UDS}$  and  $\overline{LDS}$  are both driven low.

In step 4, the M68000 reads the vector number, multiplies it by 4 to get the vector address, fetches a 4-byte program address from that vector address (see Table 2-5), and then jumps to that 4-byte address. That 4-byte address is the location of the first instruction in the interrupt handler.

Steps 1 and 3 are the responsibility of the interrupt controller on the IMP. In steps 1 and 3, a number of configuration options are available. For instance, in step 1, there are two modes for handling external interrupts: normal and dedicated. In step 3, there are several different ways of generating vectors. These and other interrupt controller options are introduced in the following paragraphs.

### 3.2.1.2 Interrupt Controller Overview

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 a register in that peripheral.

In addition to the INRQ interrupts, the interrupt controller can also receive external requests (EXRQ). EXRQ interrupts are input to the IMP according to normal or dedicated mode. In the normal mode, EXRQ interrupts 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  $\overline{IPL2}$ – $\overline{IPL0}$  as in the original MC68000. Up to seven levels of interrupt priority may be encoded. Level 4 is reserved for IMP INRQ interrupts and may not be generated by an external device.

### 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 should also be cleared by the user program.) To clear a bit in the ISR, the user writes a one to that bit. The user can only clear bits in this register, and bits that are written with zeros will not be affected. The ISR is cleared at 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.

If the INRQ error vector is taken, no bit in the ISR is set. Bit 0 of the ISR is always zero.

15	14	13	12	11	10	9	8
PB11	PB10	SCC1	SDMA	IDMA	SCC2	TIMER1	SCC3

7	6	5	4	3	2	1	0
PB9	TIMER2	SCP	TIMER3	SMC1	SMC2	PB8	0

### 3.2.6 Interrupt Handler Examples

The following examples illustrate proper interrupt handling on the IMP. Nesting of level 4 interrupts (a technique described earlier) is not implemented in the following examples.

#### Example 1—Timer 3 (Software Watchdog Timer) Interrupt Handler

1. Vector to interrupt handler.
2. (Handle Event)
3. Clear the TIMER3 bit in the ISR.
4. Execute RTE instruction.

#### Example 2— SCC1 Interrupt Handler

1. Vector to interrupt handler.

2. Immediately read the SCC1 event (SCCE1) register into a temporary location.
3. Decide which events in the SCCE1 will be handled in this handler and clear those bits in the SCCE1 as soon as possible.

(Handle events in the SCC1 Rx or Tx BD tables.)

At the end:

4. Clear the SCC1 bit in the ISR.
5. Execute RTE instruction. If any unmasked bits in SCCE1 remain at this time (either uncleared by the software or set by the IMP during the execution of this handler), this interrupt source will be made pending again immediately following the RTE instruction.

In example 1, the hardware clears the TIMER3 bit in the IPR during the interrupt acknowledge cycle. This is an example of a handler for an interrupt source without multiple events. In example 2, the IPR bit remains set as long as one or more unmasked event bits remain in the SCCE1 register. This is an example of a handler for an interrupt source with multiple events.

Note that, in both cases, it is not necessary to clear the IPR bit; however, in both cases, it is necessary to clear the ISR bit to allow future interrupts from this source.

### 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. An example block diagram of PA0 is given in Figure 3-5

nously 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.

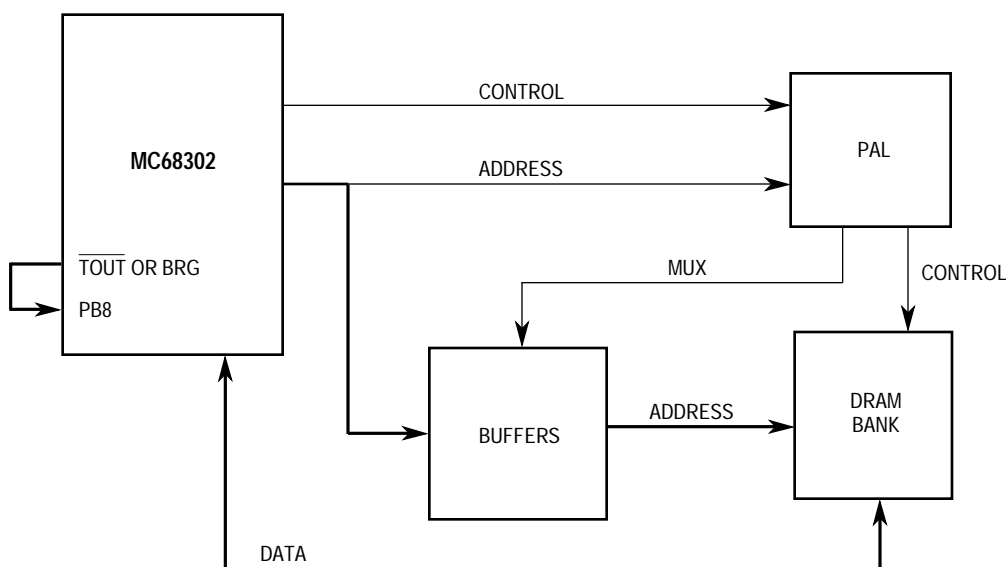
The 576-byte parameter RAM area includes pointers, counters, and registers used with the serial ports. This area is accessed by the CP during communications processing. Any individual locations not required in a given application may be used as general-purpose RAM.

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 3-7. 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.

to the DRAM bank. The PAL generates the RAS and CAS lines for the DRAM chips and controls the address multiplexing in the external address buffers. One of the MC68000 chip-select lines can be used as the DRAM bank enable signal, if desired.

The refresh operation is a byte read operation. Thus,  $\overline{UDS}$  or  $\overline{LDS}$  will be asserted from the MC68302, but not both. A refresh to an odd address will assert  $\overline{LDS}$ ; whereas, a refresh to an even address will assert  $\overline{UDS}$ .



**Figure 3-13. DRAM Control Block Diagram**

### 3.10.2 DRAM Refresh Controller Bus Timing

The DRAM refresh controller bus cycles are actually SDMA byte read 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  $\overline{BCLR}$  signal, just like the SDMA. Note that the function code bits may be used to distinguish DRAM refresh cycles from SDMA cycles, if desired.

A bus error on a DRAM refresh controller access causes the  $\overline{BERR}$  channel number at offset BASE + \$67C to be written with a \$0001. This is also the value written if the SCC1 receive SDMA channel experiences a bus error; thus, these two sources cannot be distinguished upon a bus error. The DRAM refresh SDMA channel and SCC1 receive SDMA channel are separate and independent in all other respects.

### 3.10.3 Refresh Request Calculations

A typical 1-Mbyte DRAM needs one refresh cycle every 15.625  $\mu$ s. 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  $\mu$ s. 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.





If  $\overline{\text{RTS}}$  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.

**Table 4-5. Transmit Data Delay (TCLK Periods)**

Protocol Type	From RTS Low	From CTS Low
Asynchronous Protocols (16x clock)	0	48
Synchronous Protocols (1x clock)	1	3.5

**NOTES:**

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

$\overline{\text{RTS}}$  is negated by the SCC one clock after the last bit in the frame. Figure 4-13 shows a diagram of synchronous mode timing from  $\overline{\text{RTS}}$  low. Figure 4-14 shows a diagram of synchronous mode timing delays from  $\overline{\text{CTS}}$  low.

The SCC samples  $\overline{\text{CTS}}$  on the every rising edge of the TCLK. If  $\overline{\text{CTS}}$  is negated when  $\overline{\text{RTS}}$  is asserted, a CTS lost error occurs. If a synchronous protocol is used, the transmit data will be aborted after four additional bits are transmitted. If an asynchronous protocol is used, the transmit data will be aborted after three additional bits are transmitted. See the transmit error section of each protocol for further details and steps to be taken following a CTS lost error.

The SCC latches its first bit of valid receive data on the same clock edge (rising RCLK) that samples  $\overline{\text{CD}}$  as low. The only exception is when the EXSYN bit is set in the SCC mode register for the BISYNC and Transparent protocols.

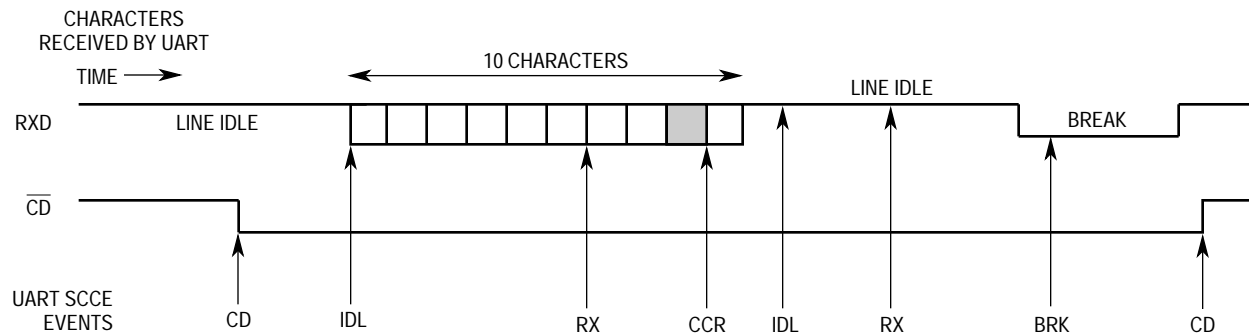
If  $\overline{\text{CD}}$  is negated during frame reception, a CD lost error occurs and the SCC will quit receiving data within four additional bit times. At this point, any residue of bits less than 8 bits (or 16 bits in HDLC or transparent modes) will be discarded and not written to memory. Thus, the last bit written to memory will be within plus or minus four bit times from the point at which  $\overline{\text{CD}}$  was negated.

**NOTE**

The CTS lost error and CD lost error (with  $\overline{\text{CTS}}$  and  $\overline{\text{CD}}$  under automatic control) is not intended to implement a flow control method in the UART protocol. The software operation of the DIAG1–DIAG0 bits should be chosen if UART flow control is desired, with transmission being temporarily suspended by the FRZ bit in the UART event register. CTS lost and CD lost, as defined here, are intended to implement the aborting of transmission and reception as defined in many synchronous protocols.

01 = Loopback mode


In this mode, the transmitter output is internally connected to the receiver input while the receiver and the transmitter operate normally. The value on the RXD pin is ignored. For the NMSI2 and NMSI3 pins, the TXD pin may be programmed to either show the transmitted data or not show the data by programming port A par-

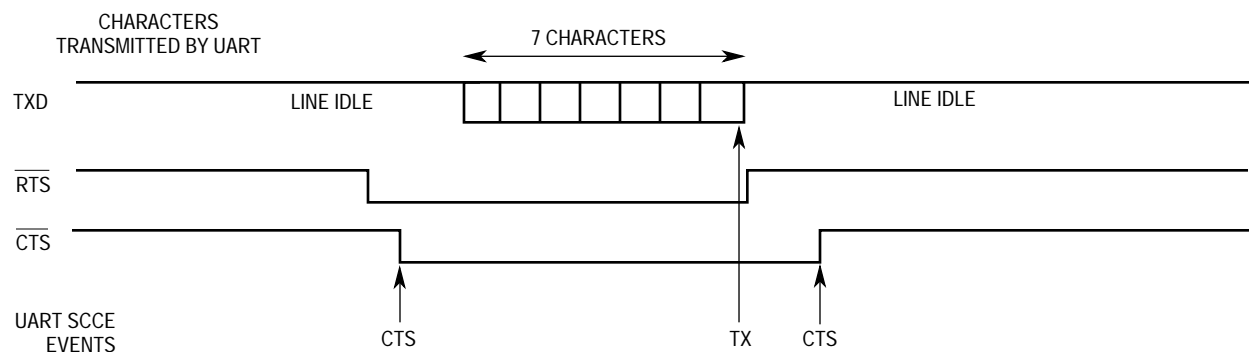


**NOTES:**

1. The first RX event assumes receive buffers are six bytes each.
2. The second IDL event occurs after 9 to 13 ones received in a row.
3. The second RX event position is programmable based on the max\_IDL value.
4. The BRK event occurs after the first break character is received.

**LEGEND:**

 Is a receive control character defined not to be stored in the receive buffer.



NOTE: TX event assumes all seven characters were put into a single buffer.

**Figure 4-23. UART Interrupt Events Example**

7	6	5	4	3	2	1	0
CTS	CD	IDL	BRK	CCR	BSY	TX	RX

**CTS—Clear-To-End Status Changed**

A change in the status of the  $\overline{\text{CTS}}$  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  $\overline{\text{CD}}$  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 receive serial line was detected on the UART channel. The SCC status register may be read to determine the current status.

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 (overflow, 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 or sixteen transmit clocks as determined by the FLG bit in the HDLC mode register.

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). The TBD# is not advanced. 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. The abort sequence on transmit is a zero followed by seven ones (01111111).

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  $\overline{CTS}$  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.

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 asynchronous 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 (CD), clear to send (CTS), and request to send (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 information 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

#### 4.5.14.1 DDCMP Channel Frame Transmission Processing

The DDCMP transmitter is designed to work with almost no intervention from the M68000 core (see Figure 4-35).

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).

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 or before 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.

### $\overline{\text{DREQ}}$ /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

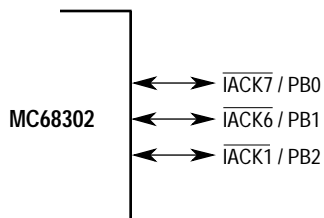
This output, asserted by the IDMA, signals to the peripheral that an operand is being transferred in response to a previous transfer request.

### $\overline{\text{DONE}}$ /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

The IACK or PIO port B pins are shown in Figure 5-14.



**Figure 5-14. IACK or PIO Port B Pins**

Each one of these three pins can be used either as an interrupt acknowledge signal or as a general-purpose parallel I/O port. Note that the IMP interrupt controller does not require the use of the IACK pins when it supplies the interrupt vector for the external source. 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 AVEC for autovectoring (unless internal vector generation is used).

## 5.18 TIMER PINS

The timer pins are shown in Figure 5-15.

## APPENDIX D

### MC68302 APPLICATIONS

This appendix describes different applications for the MC68302.

#### D.1 MINIMUM SYSTEM CONFIGURATION

The following paragraphs describe a minimum 16-bit MC68302 system. As Figure D-1 shows, this system can be easily built with very few components.

##### D.1.1 System Configuration

The crystal circuit shown in Figure D-1 is a typical configuration. The values used are not required by Motorola. Some deviation of the capacitance or resistance values is allowed. Crystal parameters need not be anything special— $C_0 < 10 \text{ pF}$  and  $R_x = 50 \Omega$  is used. Of course, an oscillator could be used in place of the crystal circuit.

$\overline{\text{AVEC}}$  is pulled high since autovectoring for external interrupts is not needed. If external devices were added (not shown) the MC68302 interrupt controller could handle the interrupt vector generation for up to seven external sources using  $\overline{\text{IRQ7}}$ ,  $\overline{\text{IRQ6}}$ ,  $\overline{\text{IRQ1}}$ , PB11, PB10, PB9, and PB8. The  $\overline{\text{IPL2}}$ – $\overline{\text{IPL0}}$  lines are also pulled high (inactive) since no external interrupts are required.

BUSW is pulled high for 16-bit operation and may not be modified dynamically. Choice of 8-bit operation could be used to eliminate two of the memory chips.

$\overline{\text{BERR}}$  is pulled high since it is an open-drain signal. It will be asserted low by the MC68302 if the hardware watchdog terminates a stalled bus cycle.

$\overline{\text{BR}}$  is tied high since no external bus masters exist in this design.

$\overline{\text{BGACK}}$  is pulled high (inactive). It is asserted low during an IDMA or SDMA bus cycle.

$\overline{\text{FRZ}}$  is tied high since the MC68302 freeze debugging logic is not used in this design.

DISCPU is tied low to allow the M68000 core to function normally. Tying this pin high causes the part to enter the disable CPU mode when the core is disabled and the part is an intelligent peripheral.

All unused lines should be terminated.



- e. Clear all status bits to zero (the default condition).
- f. Set (PRD -> Empty) = 1.
- g. Move PRD to point to the next BD (If the wrap bit is set, point to the first BD).

If the interrupt was due to a buffer being transmitted, perform the following confirm process:

2. While (CTD -> Ready) = 0 and (CTD -> data length) > 0, do the following:
  - /\* Look at frames that have been transmitted, but not confirmed \*/
  - a. Check for errors in the BD.
  - b. Clear all status bits to zero (the default condition).
  - c. Clear out the data length field so that it is zero to start with. This procedure allows new data be placed in this transmit buffer by the transmit algorithm.
  - d. Move CTD to point to the next BD. (If the wrap bit is set, point to the first BD.)

Finally:

3. Clear the SCC bit in the in-service register (ISR) of the interrupt controller. This is standard procedure.
4. Return from interrupt.

### D.3.7 Final Comments

Note that nowhere in the algorithm does RBD# or TBD# need to be read. The empty and ready bits provide all the necessary information.

Whether receiving buffers or confirming buffers, the interrupt routines deal with all the buffers they can before the interrupt routine is exited. This approach is not mandatory, but if not all buffers are dealt with, then some provision needs to be made for getting the work done without waiting for another interrupt.

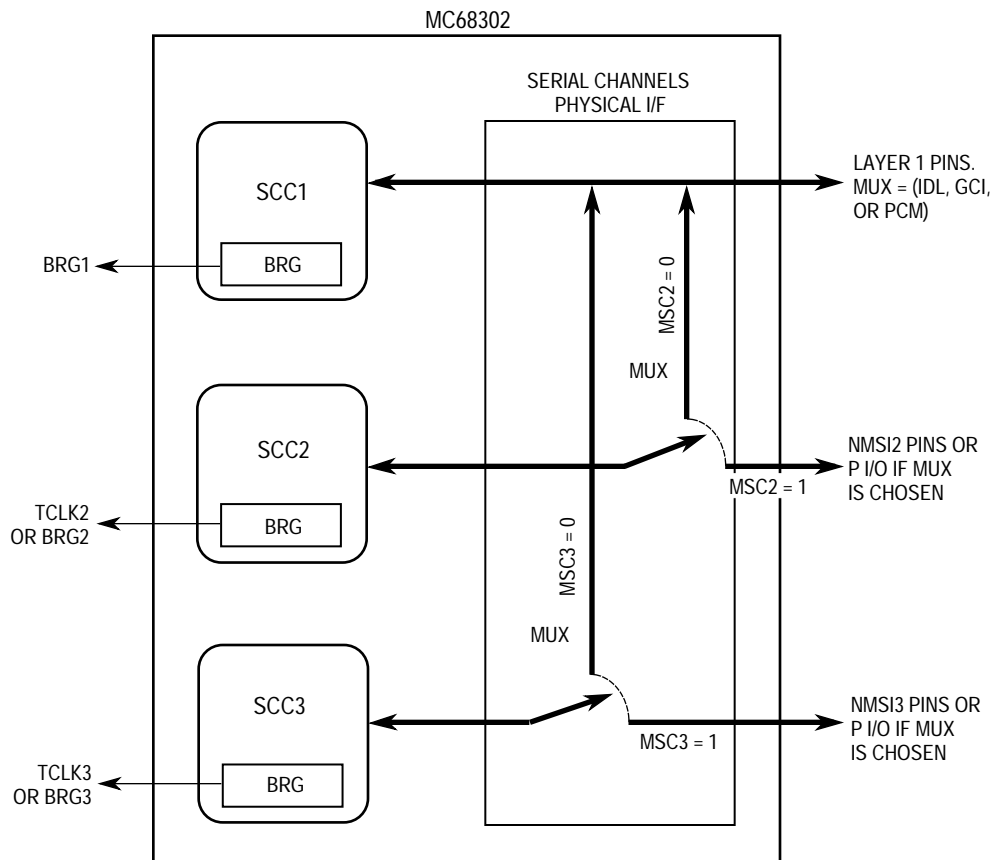
For frame-oriented protocols such as HDLC, the user may wish to only process frames, not buffers. In this case, the algorithm would be the same, but the interrupt mask would be set for frame interrupts only.

### D.3.8 HDLC Code Listing

The following code shows the initialization of the HDLC protocol and the implementation of the buffer processing algorithms just described. HDLC frames are continuously transmitted and received in the SCC loopback mode.

```
***** BUFFER PROCESSING CODE*****
*****EQU TABLE*****
* The following three values are application dependent
BASE      EQU      $0700000      ;This is set according to the val in BAR
INIT      EQU      $0030300      ;Initialization Routine
INT_VEC    EQU      $0031000      ;Interrupt Vector for SCC1
* Commonly used Registers and Parameters
BAR        EQU      $0F2          ;Base Address Register
SCR        EQU      $0F4          ;System Control Register
CKCR       EQU      $0F6          ;Clock Control Register
GIMR       EQU      BASE + $812    ;Global Interrupt Mode Register
IPR        EQU      BASE + $814    ;Interrupt Pending Register
```





**Figure D-22. Multiplexed Modes Example**

PCM highway is any time-division multiplexed serial interface (i.e., data is transferred over time slots). The most common examples of a PCM highway (serial bus) are a T1 line or a CEPT line; however, there are many other possible PCM highway configurations. T1 has 24 8-bit channels and is clocked at 1.544 MHz. CEPT has 32 8-bit channels and is clocked at 2.048 MHz. In both cases, the actual rate of an 8-bit channel is 64 kbps. If more data throughput is needed, multiple 8-bit time slots can be grouped together for faster data transfer. You may wish to choose PCM highway, even if you only have one SCC using the interface (and therefore one long time slot) as will be shown.

The following examples illustrate how the different types of interfaces can be combined.

Example 1. If you need multiple transparent channels on separate physical interfaces, then all of them can be NMSI or all but one as NMSI. Thus, you can choose from the following four combinations:

1. NMSI1, NMSI2, and NMSI3
2. PCM, NMSI2, and NMSI3
3. IDL, NMSI2, and NMSI3
4. GCI, NMSI2, and NMSI3

## I—Interrupt

- 0 = The TXB bit in the event register is not set when this buffer is closed.
- 1 = The TXB bit in the event register is set if this buffer closed without an error. If an error occurred, then TXE is set.

## L—Last in Frame

- 0 = This buffer is not the last buffer in a frame.
- 1 = This buffer is the last buffer in a frame.

## TC—Tx CRC

- 0 = Transmit the closing flag after the last data byte.
- 1 = Transmit the CRC sequence after the last data byte.

## Bits 9-2—Reserved for future use

## UN—Underrun

- 0 = No transmitter underrun occurred.
- 1 = A transmitter underrun condition occurred while transmitting the associated data buffer.

## CT—CTS Lost

- 0 = No CTS or L1GR lost was detected during frame transmission.
- 1 = CTS in NMSI mode or L1GR in IDL/GCI mode was lost during frame transmission.

**E.1.1.5.2 Transmit Buffer Data Length.** This 16-bit value is written by the user to indicate the number of data bytes to be transmitted from the data buffer.

**E.1.1.5.3 Transmit Buffer Pointer.** This 32-bit value is written by the user to indicate the address of the first byte of data in the data buffer.

## E.1.2 Programming the SCC for HDLC

This section gives a generic algorithm for programming an SCC to handle HDLC. The algorithm is intended to show what must be done and in what order to initialize the SCC and prepare the SCC for transmission and reception. The algorithm is not specific and assumes that the IMP and other on-chip peripherals have been initialized as required by the system hardware (timers, chip selects, etc.).

### E.1.2.1 CP INITIALIZATION.

1. Write the port A and port B control registers (PACNT and PBCNT) to configure SCC2 or SCC3 serial interface pins as peripheral pins, if SCC2 or SCC3 is used.
2. Write SIMODE to configure the SCCs physical interface.
3. Write SIMASK if IDL or GCI multiplexed mode was selected in SIMODE.

### E.1.2.2 GENERAL AND HDLC PROTOCOL-SPECIFIC RAM INITIALIZATION.

4. Write RFCR/TFRCR.



- V.110 Rate Adaption 4-117
  - Data Types 2-3
  - DDCMP
    - Asynchronous DDCMP Mode 4-46
    - Carrier Detect Lost 4-109
    - Clear-To-Send Lost 4-109
    - CRC Error 4-109
    - DDCMP Address Recognition 4-108
    - DDCMP Event Register 4-112, 4-115, 4-116
    - DDCMP Frames 4-102
    - DDCMP Mask Register 4-117
    - DDCMP Memory Map 4-105
    - DDCMP Mode Register 4-110
    - DDLE 4-108
    - DENQ 4-108
    - DSOH 4-108
    - DSR 4-105
    - DSYN1 4-107
    - FIFO 4-108
    - Framing Error 4-109
    - Overrun Error 4-109
    - Parity Error 4-110
    - RTS 4-111
    - Rx BD 4-111
    - SCCE 4-116
    - SCCM 4-117
    - SYN1-SYN2 4-110
    - Transmitter Underrun 4-108
    - Tx BD 4-114
  - DDCMP Controller 4-102
  - Disable CPU 5-6
    - AVEC 3-55
    - BCLR 3-55
    - BG 3-54
    - BR 3-54
    - CS0 3-55
    - DTACK 3-55
    - EMWS 3-55
    - IOUT0/IOUT1/IOUT2 3-55
    - Low-Power Modes 3-55
    - RMC 3-55
    - SAM 3-55
    - Vector Generation Enable (VGE) 3-55
  - Disabled 4-39, 4-43
  - Disabling the SCCs 4-42
  - DISCPU 3-54, 5-6
  - DONE 5-20
  - DRAM Refresh 3-66, 3-67, 4-35
    - BERR Channel Number 3-67
    - Buffer Descriptors 3-66
    - Bus Bandwidth 3-66
    - Bus Exception 3-66
    - ERRE 3-68
    - PB8 3-32
    - SDMA 3-67
  - DREQ 5-20
  - DSR 4-32
  - DTACK 2-8, 3-21, 3-34, 3-47, 3-48, 3-53, 3-54, 3-55, 5-6, 5-12, See Dual-Port RAM, See Signals
  - Dual-Port RAM 1-5, 2-14, 3-33
    - BR 3-34
    - DTACK 3-34
    - EMWS 3-34
    - SAM 3-34
- E**
- E 2-11, See Signals
  - EMWS (External Master Wait State) 3-34, 3-53, 3-54, 3-55
  - Enable Receiver 4-31
  - Enable Transmitter 4-31
  - ENTER HUNT MODE Command 4-6, 4-36, 4-37, 4-43, 4-50, 4-72, 4-89, 4-107
  - Envelope Mode 4-17
  - ERRE 3-68, See DRAM Refresh
  - Error Counters 4-55, 4-75, 4-93, 4-110
  - Event Registers 2-19
  - Exception
    - Bus 3-14
    - Bus Error 3-14
    - Halt 3-14
    - PB8 3-66
    - Processing 2-7, 2-11, exception vector is determined 2-8
    - Reset 3-14
    - Retry 3-14
    - Stack Frame 2-10
    - Vectors 2-8
  - EXRQ 3-17
  - EXTAL 3-62, 5-2, 5-4
  - External
    - Bus Master 2-7, 3-58

Pin Grid Array 7-1  
 Port A/B  
     Parallel I/O 3-29  
 Power Consumption 3-60  
 Power Dissipation 6-3  
 Power-Saving Tips 3-60  
 PQFP 7-2  
 Priority Interrupts 3-20  
 Privilege State 2-6, 2-8  
 Protocol Parameters 2-15  
 Pullup Resistors 5-23

## R

RAM  
     Dual-Port 1-5, 3-33  
     Parameter 3-34  
     System 3-34  
 RBD  
 Read-Modify-Write 2-11, 2-19, See  
     Instructions  
 Read-Modify-Write Cycle 6-11, 6-12  
 Receive BDs 4-34  
 Received Control Character Register 4-51  
 Reception Errors 4-54  
 Registers  
     Base Address 2-12  
     Event 2-19  
     Internal 1-6, 2-16  
     Interrupt In-Service (ISR) 3-28  
     Interrupt Mask (IMR) 3-32  
     Interrupt Pending (IPR) 3-26  
     Port A  
         Control (PACNT) 3-29  
         Data Direction (PADDDR) 3-30  
     Port B  
         Control (PBCNT) 3-31  
         Data Direction Register (PBDDR) 3-31  
     Status 2-2, 2-8, 2-11, 3-17, 3-20  
     System Configuration 2-12, 2-14  
     System Control (SCR) 2-12, 2-13  
 RESET 2-19, 3-41, 3-44, 3-62, 5-6, 5-22  
     Instruction 2-7, 2-13, 2-19  
 Reset 4-39  
     SMC Interrupt Requests 4-145  
     SMC Loopback 4-141  
     SMC Memory Structure 4-142

TIMEOUT Command 4-142  
 Total System 2-13  
 TRANSMIT ABORT REQUEST  
     Command 4-142  
 RESET BCS CALCULATION Command 4-89  
 RESTART TRANSMIT Command 4-6, 4-49, 4-72, 4-89, 4-107  
 Reverse DATA 4-94  
 Revision Number 2-15  
 RISC Processor 4-1  
 RMC 2-11, 3-53, 3-55, 3-58, 5-9  
 RTE 3-29  
 RTS 4-18, 4-28

## S

SAM 3-34, 3-54, 3-55  
 SAM See Dual-Port RAM  
 SCC 2-15  
     Asynchronous Baud Rate 4-26  
     Baud Rate Generator 4-25  
     Buffer Descriptor 4-32  
     CD 4-28, 4-40  
     Clock Divider 4-26  
     CTS 4-28, 4-40  
     Disabled 4-39, 4-43  
     DSR 4-32  
     Enable Receiver 4-31  
     ENTER HUNT MODE Command 4-36, 4-37  
     External Loopback 4-30  
     Function Code 4-34, 4-36  
     Idle Status 4-40  
     IPR 4-39  
     Low-Power 4-43  
     MRBLR 4-36  
     Normal Operation 4-28  
     Output Delays 4-30  
     Performance 4-23  
     Promiscuous Operation 4-124  
 RBD  
 Receive BDs 4-34  
 Reset 4-39  
 RTS 4-28  
 SCC Initialization 4-38  
 SCCE 4-39  
 SCCM 4-39