
**Welcome to 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 | CPU32+ |
| Number of Cores/Bus Width | 1 Core, 32-Bit |
| Speed | 25MHz |
| Co-Processors/DSP | Communications; CPM |
| RAM Controllers | DRAM |
| Graphics Acceleration | No |
| Display & Interface Controllers | - |
| Ethernet | 10Mbps (1) |
| SATA | - |
| USB | - |
| Voltage - I/O | 5.0V |
| Operating Temperature | 0°C ~ 70°C (TA) |
| Security Features | - |
| Package / Case | 240-BFQFP |
| Supplier Device Package | 240-FQFP (32x32) |
| Purchase URL | https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68360em25l |

mand registers. The parameter RAM of the SCCs is very similar, and most parameter RAM register names and usage are retained. More importantly, the basic structure of a buffer descriptor (BD) on the QUICC is identical to that of the MC68302, except for a few new bit functions that were added. (In a few cases, a bit in a BD status word had to be shifted.)

- When porting code from the MC68302 CP to the QUICC CPM, the software writer may find that the QUICC has new options to simplify what used to be a more code-intensive process. For specific examples, see the INIT TX AND RX PARAMETERS, GRACEFUL STOP TRANSMIT, and CLOSE BD commands.

## 1.4 QUICC GLUELESS SYSTEM DESIGN

A fundamental design goal of the QUICC was ease of interface to other system components. An example of this goal is a minimal QUICC design using EPROM and DRAM, shown in Figure 1-2. This system interfaces gluelessly to an EPROM and a DRAM SIMM module. It also offers parity support for the DRAM.

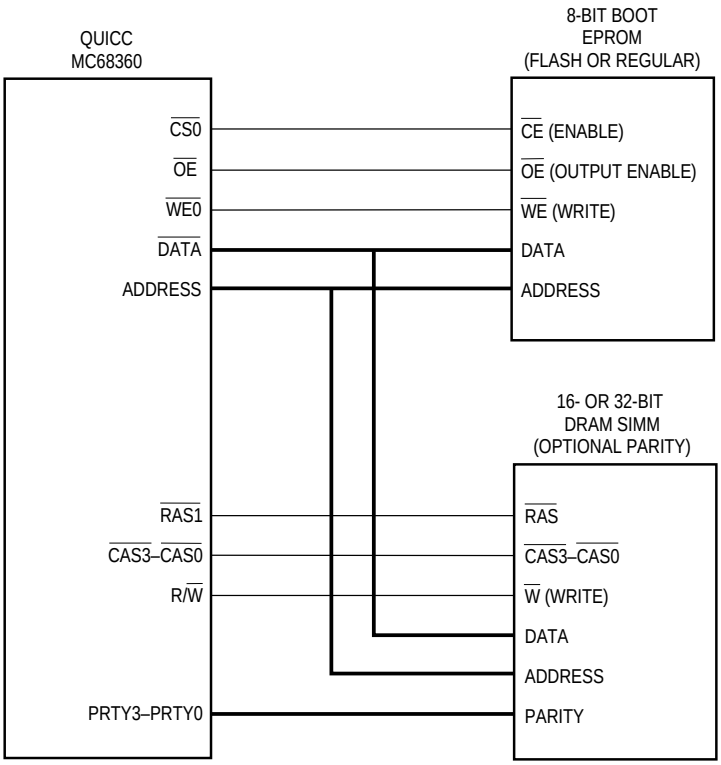



**Figure 1-2. Minimum QUICC System Configuration**

Figure 1-3 shows a larger system configuration. This system offers one EPROM, one flash EPROM, and supports two DRAM SIMMs. Depending on the capacitance on the system bus, external buffers may be required. From a logic standpoint, however, a glueless system is maintained.

### 4.1.1 Bus Control Signals

The QUICC initiates a bus cycle by driving the address, size, function code, and read/write outputs. At the beginning of a bus cycle, SIZ1 and SIZ0 are driven with the FC signals. SIZ1 and SIZ0 indicate the number of bytes remaining to be transferred during an operand cycle (consisting of one or more bus cycles). Table 4-3 lists the encoding of SIZ1 and SIZ0. These signals are valid while $\overline{AS}$ is asserted.

The R/$\overline{W}$ signal determines the direction of the transfer during a bus cycle. Driven at the beginning of a bus cycle, R/$\overline{W}$ is valid while $\overline{AS}$ is asserted. R/$\overline{W}$ only transitions when a write cycle is preceded by a read cycle or vice versa. The signal may remain low for consecutive write cycles.

The $\overline{RMC}$ signal is asserted at the beginning of the first bus cycle of a read-modify-write operation and remains asserted until completion of the final bus cycle of the operation.

### 4.1.2 Function Codes (FC3–FC0)

The FCx signals are outputs that indicate one of 16 address spaces to which the address applies. Fifteen of these spaces are designated as either a normal or DMA cycle, user or supervisor, and program or data spaces. One other address space is designated as CPU space to allow the CPU32+ to acquire specific control information not normally associated with read or write bus cycles. The FCx signals are valid while $\overline{AS}$ is asserted.

Function codes (see Table 4-1) can be considered as extensions of the 32-bit address that can provide up to eight different 4-Gbyte address spaces. Function codes are automatically generated by the CPU32+ to select address spaces for data and program at both user and supervisor privilege levels, and a CPU address space for processor functions. User programs access only their own program and data areas to increase protection of system integrity and can be restricted from accessing other information. The S-bit in the CPU32+ status register is set for supervisor accesses and cleared for user accesses to provide differentiation. Refer to 4.4 CPU Space Cycles for more information.

**Table 4-1. Address Space Encoding**

| Function Code Bits | | | | Address Spaces |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | |
| 0 | 0 | 0 | 0 | Reserved (Motorola) |
| 0 | 0 | 0 | 1 | User Data Space |
| 0 | 0 | 1 | 0 | User Program Space |
| 0 | 0 | 1 | 1 | Reserved (User) |
| 0 | 1 | 0 | 0 | Reserved (Motorola) |
| 0 | 1 | 0 | 1 | Supervisor Data Space |
| 0 | 1 | 1 | 0 | Supervisor Program Space |
| 0 | 1 | 1 | 1 | Supervisor CPU Space |
| 1 | x | x | x | DMA space |

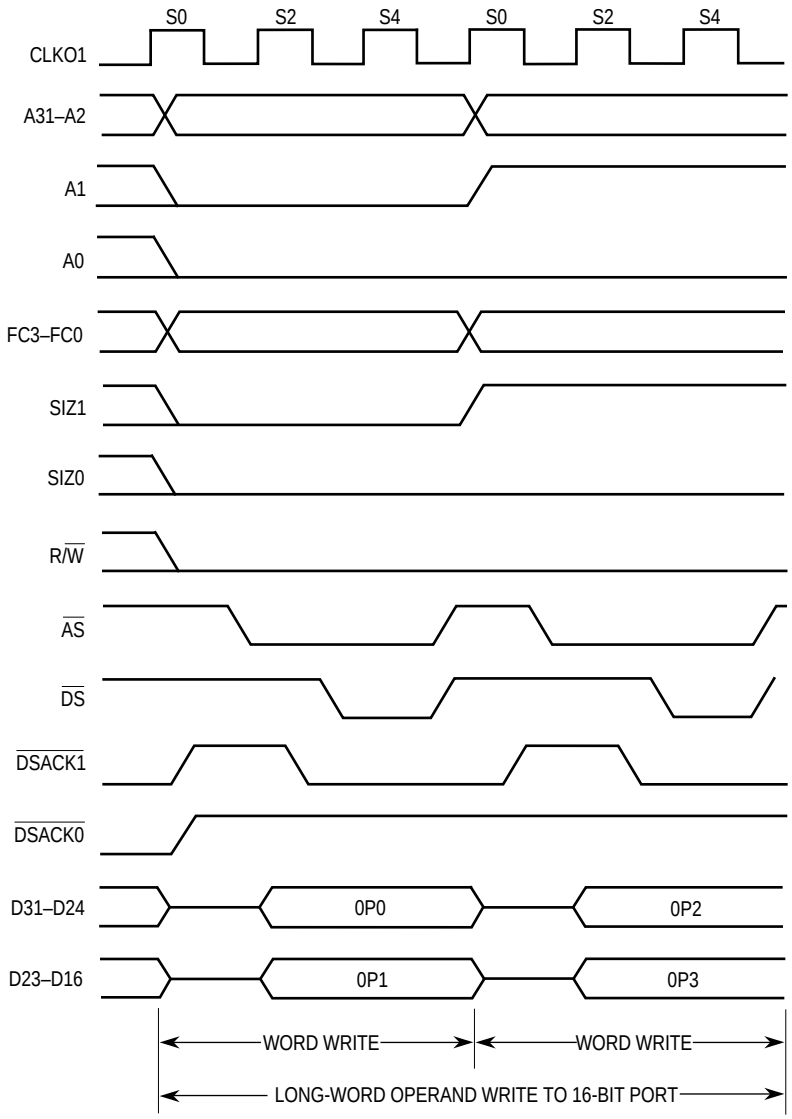



**Figure 4-5. Long-Word Operand Write Timing (16-Bit Data Port)**

Figure 4-6 shows a word transfer to an 8-bit bus port. Like the preceding example, this example requires two bus cycles. Each bus cycle transfers a single byte. The size signals for the first cycle specify two bytes; for the second cycle, they specify one byte. Figure 4-7 shows the associated bus transfer signal timing.

## 4.2.2 Misaligned Operands

Since operands may reside at any byte boundaries, they may be misaligned. A byte operand is properly aligned at any address; a word operand is misaligned at an odd address; a long word is misaligned at an address that is not evenly divisible by four. The MC68302, MC68000/MC68008, MC68010, and MC68340 implementations allow long-word transfers on odd-word boundaries but force exceptions if word or long-word operand transfers are attempted at odd-byte addresses. Although the QUICC does not enforce any alignment restrictions for data operands (including PC relative data addresses), some performance degradation occurs when additional bus cycles are required for long-word or word operands

## Table 5-3. Condition Code Computations

| Operations | X | N | Z | V | C | Special Definition |
|---|---|---|---|---|---|---|
| ABCD | * | U | ? | U | ? | C = Decimal Carry<br>Z = Z Λ Rm Λ ... Λ R0 |
| ADD, ADDI, ADDQ | * | * | * | ? | ? | V = Sm Λ Dm Λ $\overline{Rm}$ V $\overline{Sm}$ Λ $\overline{Dm}$ Λ Rm<br>C = Sm Λ Dm V Rm Λ Dm V Sm Λ Rm |
| ADDX | * | * | ? | ? | ? | V = Sm Λ Dm Λ $\overline{Rm}$ V $\overline{Sm}$ Λ $\overline{Dm}$ Λ Rm<br>C = Sm Λ Dm V Rm Λ Dm V Sm Λ Rm<br>Z = Z Λ Rm Λ ... Λ R0 |
| AND, ANDI, EOR, EORI, MOVEQ, MOVE, OR, ORI, CLR, EXT, NOT, TAS, TST | — | * | * | 0 | 0 | |
| CHK | — | * | U | U | U | |
| CHK2, CMP2 | — | U | ? | U | ? | Z = (R = LB) V (R = UB)<br>C = (LB < UB) Λ (IR < LB) V (R > UB) V (UB < LB) Λ (R > UB) Λ (R < LB) |
| SUB, SUBI, SUBQ | * | * | * | ? | ? | V = $\overline{Sm}$ Λ Dm Λ $\overline{Rm}$ V Sm Λ $\overline{Dm}$ Λ Rm<br>C = Sm Λ Dm V Rm Λ Dm V Sm Λ Rm |
| SUBX | * | * | ? | ? | ? | V = $\overline{Sm}$ Λ Dm Λ $\overline{Rm}$ V Sm Λ $\overline{Dm}$ Λ Rm<br>C = Sm Λ Dm V Rm Λ Dm V Sm Λ Rm<br>Z = Z Λ Rm Λ ... Λ R0 |
| CMP, CMPI, CMPM | — | * | * | ? | ? | V = $\overline{Sm}$ Λ Dm Λ $\overline{Rm}$ V Sm Λ $\overline{Dm}$ Λ Rm<br>C = Sm Λ Dm V Rm Λ Dm V Sm Λ Rm |
| DIVS, DIVU | — | * | * | ? | 0 | V = Division Overflow |
| MULS, MULU | — | * | * | ? | 0 | V = Multiplication Overflow |
| SBCD, NBCD | * | U | ? | U | ? | C = Decimal Borrow<br>Z = Z Λ Rm Λ ... Λ R0 |
| NEG | * | * | * | ? | ? | V = Dm Λ Rm<br>C = Dm V Rm |
| NEGX | * | * | ? | ? | ? | V = Dm Λ Rm<br>C = Dm V Rm<br>Z = Z Λ Rm Λ ... Λ R0 |
| ASL | * | * | * | ? | ? | V = Dm Λ (Dm – 1 V ... V $\overline{Dm - r}$) V $\overline{Dm}$ Λ (Dm – 1 V ... + Dm – r)<br>C = Dm – r + 1 |
| ASL (r = 0) | — | * | * | 0 | 0 | |
| LSL, ROXL | * | * | * | 0 | ? | C = Dm – r + 1 |
| LSR (r = 0) | — | * | * | 0 | 0 | |
| ROXL (r = 0) | — | * | * | 0 | ? | C = X |
| ROL | — | * | * | 0 | ? | C = Dm – r + 1 |
| ROL (r = 0) | — | * | * | 0 | 0 | |
| ASR, LSR, ROXR | * | * | * | 0 | ? | C = Dr – 1 |
| ASR, LSR (r = 0) | — | * | * | 0 | 0 | |
| ROXR (r = 0) | — | * | * | 0 | ? | C = X |
| ROR | — | * | * | | 0 | ? |
| ROR (r = 0) | — | * | * | | 0 | 0 |

## Table 5-16. Exception Vector Assignments

| Vector Number | Vector Offset | | | Assignment |
|---|---|---|---|---|
| | Dec | Hex | Space | |
| 0 | 0 | 000 | SP | Reset: Initial Stack Pointer |
| 1 | 4 | 004 | SP | Reset: Initial Program Counter |
| 2 | 8 | 008 | SD | Bus Error |
| 3 | 12 | 00C | SD | Address Error |
| 4 | 16 | 010 | SD | Illegal Instruction |
| 5 | 20 | 014 | SD | Zero Division |
| 6 | 24 | 018 | SD | CHK, CHK2 Instructions |
| 7 | 28 | 01C | SD | TRAPcc, TRAPV Instructions |
| 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 | 48 | 030 | SD | Hardware Breakpoint |
| 13 | 52 | 034 | SD | (Reserved for Coprocessor Protocol Violation) |
| 14 | 56 | 038 | SD | Format Error |
| 15 | 60 | 03C | SD | Uninitialized Interrupt |
| 16–23 | 64<br>92 | 040<br>05C | SD | (Unassigned, Reserved)<br>— |
| 24 | 96 | 060 | SD | Spurious Interrupt |
| 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<br>188 | 080<br>0BC | SD | Trap Instruction Vectors (0–15)<br>— |
| 48–58 | 192<br>232 | 0C0<br>0E8 | SD | (Reserved for Coprocessor)<br>— |
| 59–63 | 236<br>252 | 0EC<br>0FC | SD | (Unassigned, Reserved)<br>— |
| 64–255 | 256<br>1020 | 100<br>3FC | SD | User-Defined Vectors (192) |

**CAUTION**

Because there is no protection on the 64 processor-defined vectors, external devices can access vectors reserved for internal purposes. This practice is strongly discouraged.

word and SIZ indicates a remaining byte or word. SIZ must be set to long. All other fields should be left unchanged. The bus controller uses the modified fault address and SIZ field to rerun the complete released write cycle.

Manipulating the stacked SSW can cause unpredictable results because RTE checks only the RR bit to determine if a bus cycle must be rerun. Inadvertent alteration of the control bits could cause the bus cycle to be a read instead of a write or could cause access to a different address space than the original bus cycle. If the rerun bus cycle is a read, returned data will be ignored.

**5.5.3.2.3 Type II—Correcting Faults via RTE.** Instructions aborted because of a type II fault are restarted upon return from the exception handler. A fault handler must establish safe restart conditions. If a fault is caused by a nonresident page in a demand-paged virtual memory configuration, the fault address must be read from the stack, and the appropriate page retrieved. An RTE instruction terminates the exception handler. After unstacking the machine state, the instruction is refetched and restarted.

**5.5.3.2.4 Type III—Correcting Faults via Software.** Sufficient information is contained in the stack frame to complete MOVEM in software. After the cause of the fault is corrected, the faulted bus cycle must be rerun. Perform the following procedures to complete an instruction through software:

A. Set Up for Rerun

1. Read the MOVEM opcode and extension from locations pointed to by stack frame PC and PC + 2. The EA need not be recalculated since the next operand address is saved in the stack frame. However, the opcode EA field must be examined to determine how to update the address register and PC when the instruction is complete.
2. Adjust the mask to account for operands already transferred. Subtract the stacked operand transfer count from 16 to obtain the number of operands transferred. Scan the mask using this count value. Each time a set bit is found, clear it and decrement the counter. When the count is zero, the mask is ready for use.
3. Adjust the operand address. If the predecrement addressing mode is in effect, subtract the operand size from the stacked value; otherwise, add the operand size to the stacked value.

B. Rerun Instruction

1. Scan the mask for set bits. Read/write the selected register from/to the operand address as each bit is found.
2. As each operand is transferred, clear the mask bit and increment (decrement) the operand address. When all bits in the mask are cleared, all operands have been transferred.
3. If the addressing mode is predecrement or postincrement, update the register to complete the execution of the instruction.
4. If TR is set in the stacked SSW, create a six-word stack frame and execute the trace handler. If either B1 or B0 is set in the SSW, create another six-word stack frame and execute the hardware breakpoint handler.

Spurious Interrupt Monitor

If no interrupt arbitration occurs during an interrupt acknowledge cycle, the bus error signal is asserted internally.

Software Watchdog Timer (SWT)

The SWT asserts a reset or level 7 interrupt (as selected by the system protection control register (SYPCR)) if the software fails to service the SWT for a designated period of time (i.e., because the software is trapped in a loop or lost). There are eight selectable timeout periods. After a system reset, this function is enabled, selects a timeout of approximately 1 second, and asserts a system reset if the timeout is reached. The SWT may be disabled, or its timeout period may be changed in the SYPCR; however, once SYPCR is written, it cannot be written again until a system reset. This mechanism is used to ensure the proper operation of the SWT.

Periodic Interrupt Timer (PIT)

The SIM60 provides a timer to generate periodic interrupts for use with a real-time operating system or the application software. The PIT period can vary from 122 ms to 15.94 s (assuming a 32.768-kHz crystal is used to generate the general system clock). This function can be disabled.

Freeze Support

The SIM60 allows control of whether the SWT and PIT should continue to run during freeze mode.

Low-Power Stop Support

When executing the LPSTOP instruction, the QUICC can provide reduced power consumption with only the SIM60 remaining active.

Low-Power Standby Support

In addition to the low-power stop support, the QUICC can provide low power consumption while other modules or sub-modules are functioning. In this mode, the baud rate generators and serial ports run with a fixed frequency while the rest of the chip (including the SIM60) runs with a divided clock.

Figure 6-2 shows a block diagram of the system configuration and protection logic.

### 6.3.1 System Configuration

Many aspects of the system configuration are controlled by the MCR.

For debug purposes, accesses to internal peripherals can be shown on the external bus. This function is called show cycles. The SHEN1, SHEN0 bits in the MCR control the show cycles. External bus arbitration can be either enabled or disabled during show cycles.

The SIM60 provides eight bus arbitration levels for determining the priority of bus access (0–7). The SIM60 is fixed at the highest level (level 7). The CPU32+ is fixed at the lowest level (level 0). Only the SIM60, the CPU32+, the two-channel independent direct memory access (IDMA), and the serial direct memory access (SDMA) can be bus masters and arbitrate for

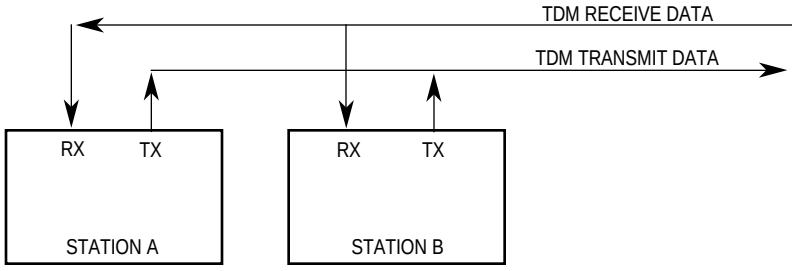



**Figure 7-27. Using the SWTR Feature**

The SWTR option gives station B the opportunity to listen to transmissions from station A and to transmit data to Station A. To do this, station B would set the SWTR bit in its receive route RAM. For this entry, receive data is taken from the L1TXD pin and data is transmitted on the L1RXD pin. If the user only wants to listen to Station A's transmissions and not transmit data on L1RXD, then the CSEL bits in the corresponding transmit route RAM entry should be cleared to prevent transmission on the L1RXD pin.

It is also possible for station B to transmit data to station A by setting the SWTR bit of the entry in its receive route RAM. Data is transmitted on the L1RXD pin rather than the L1TXD pin, according to the transmit route RAM. Note that this configuration could cause collisions with other data on the L1RXD pin unless care is taken to choose an available (quiet) time slot. If the user only wants to transmit on L1RXD and not receive data on L1TXD, then the CSEL bits in the receive route RAM should be cleared to prevent reception of data on L1TXD.

**NOTE**

If the transmit and receive sections of the TDM do not use a single clock source, this feature will give erratic results.

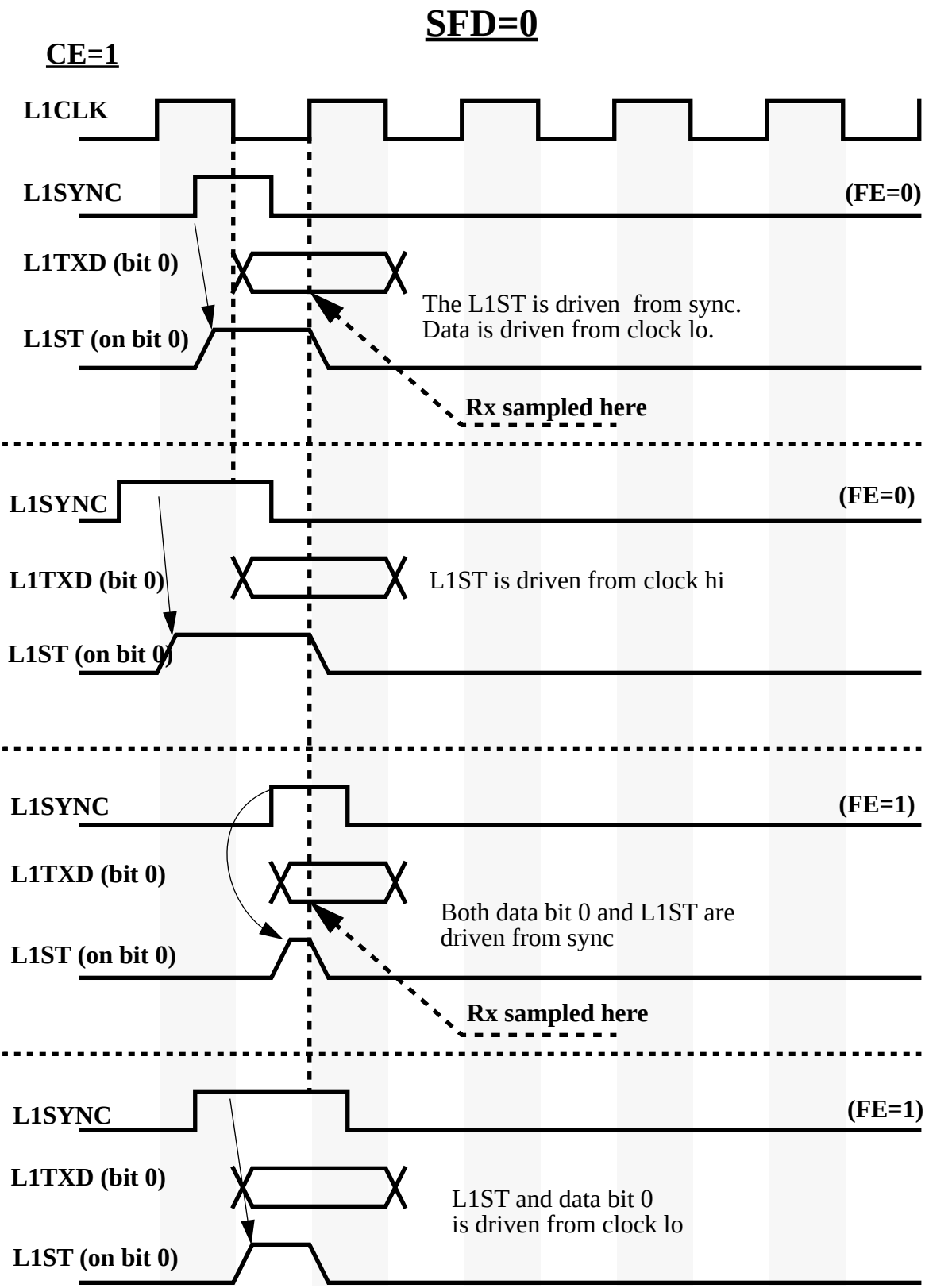
0 = Normal operation of the L1TXD and L1RXD pins.
1 = Data is transmitted on the L1RXD pin and is received from the L1TXD pin for the duration of this entry.

SSEL1–SSEL4—Strobe Select

The four strobes (L1STA1, L1STA2, L1STB1, and L1STB2) may be assigned to the receive RAM and asserted/negated with L1RCLKa or L1RCLKb or assigned to the transmit RAM and asserted/negated with L1TCLKa or L1TCLKb. Each bit corresponds to the value the strobe should have during this bit/byte group. Multiple strobes can be asserted simultaneously, if desired.

If a strobe is configured to be asserted in two consecutive SI RAM entries, then it will remain continuously asserted during the processing of both SI RAM entries. If a strobe is asserted on the last entry in the table, the strobe will be negated after the processing of that last entry is complete.

Bit 9—Reserved



Although TBPTR need never be written by the user in most applications, it may be modified by the user when the transmitter is disabled or when the user is sure that no transmit buffer is currently in use (e.g., after STOP TRANSMIT command is issued, or after a GRACEFUL STOP TRANSMIT command is issued, and the frame completes its transmission.)

**7.10.7.6 OTHER GENERAL PARAMETERS.** Additional parameters are listed in Table 7-5. 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 and for debugging.

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 internal 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 internal byte count is a down-count value that is initialized with the MRBLR value and decremented with every byte written by the SDMA channels.

**NOTE**

To extract data from a partially full receive buffer, the CLOSE Rx BD command may be used.

The Rx internal state, Tx internal state, Rx temp, Tx temp, and reserved areas are for RISC use only.

## 7.10.8 Interrupts from the SCCs

Interrupt handling for each of the SCC channels is configured on a global (per channel) basis in the CPM interrupt pending register, CPM interrupt mask register, and CPM in service register. Within each of these registers, one bit is used to either mask, enable, or report the presence of an interrupt in an SCC channel. The interrupt priority between the four SCCs is programmable in the CP interrupt configuration register. An SCC interrupt may be caused by a number of events. To allow interrupt handling for these (SCC-specific) events, further event registers are provided within the SCCs.

A number of events can cause the SCC to interrupt the processor. The events differ slightly according to the protocol selected. For a detailed description of the events see the specific protocol paragraphs. These events are handled independently for each channel by the SCC event register and the SCC mask register.

Events that can cause interrupts due to the $\overline{\text{CTS}}$ and $\overline{\text{CD}}$ modem lines are described in 7.14.9 Port C Pin Functions.

**7.10.8.1 SCC EVENT REGISTER (SCCE).** The 16-bit SCC event 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). The SCC event register appears to the user as a memory-mapped register and may be read at any time. A bit is cleared by writing a one (writing a zero does not affect a bit's value). More than one bit may be cleared at a time. This register is cleared at reset.

- May Be Used with the SCC DPLL
- Four Address Comparison Registers with Mask
- Maintenance of Five 16-Bit Error Counters
- Flag/Abort/Idle Generation/Detection
- Zero Insertion/Deletion
- 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

**7.10.17.2 HDLC CHANNEL FRAME TRANSMISSION PROCESSING.** The HDLC transmitter is designed to work with almost no intervention from the CPU32+ core. When the CPU32+ 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 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 CRC, if selected, and the closing flag are appended. In HDLC, the LSB of each octet is transmitted first, and the MSB of the CRC is transmitted first. A typical HDLC frame is shown in Figure 7-50.

| OPENING FLAG | ADDRESS | CONTROL | INFORMATION (OPTIONAL) | CRC | CLOSING FLAG |
|---|---|---|---|---|---|
| 8 BITS | 16 BITS | 8 BITS | 8N BITS | 16 BITS | 8 BITS |

**Figure 7-50. HDLC Framing Structure;**

Following the transmission of the closing flag, the HDLC controller writes the frame status bits into the BD and clears the R-bit. When the end of the current BD has been reached and the last bit is not set (working in multibuffer mode), only the R-bit is cleared. In either mode, an interrupt may be issued if the I-bit in the Tx BD is set. The HDLC controller will then proceed to the next Tx 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 CP 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.

To insert a high-priority frame without aborting the current frame, the GRACEFUL STOP TRANSMIT command may be issued. A special interrupt (GRA) can be generated in the event register when the current frame is complete.

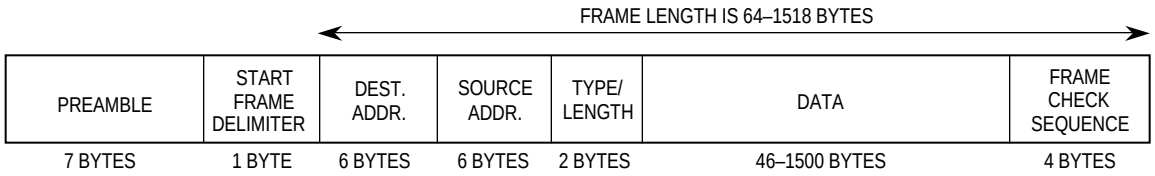
a busy (out-of-buffers) condition since only one Rx BD was prepared.

## 7.10.22 RAM Microcodes

Additional protocols may be added in the future through the use of RAM microcodes. See Appendix C RISC Microcode from RAM for more information.

## 7.10.23 Ethernet Controller

The Ethernet/IEEE 802.3 protocol is a widely used LAN that is based on the carrier sense multiple access/collision detect (CSMA/CD) approach. Ethernet and IEEE 802.3 frames are very similar and can co-exist on the same LAN. The two protocols are referred to synonymously as "Ethernet" in this manual unless specifically noted. Ethernet/IEEE 802.3 frames are based on the frame structure shown in Figure 7-65.




NOTE: The LSB of each octet is transmitted first.

**Figure 7-65. Ethernet/802.3 Frame Format;**

The frame begins with a 7-byte preamble of alternating ones and zeros. Since the frame is Manchester encoded, the preamble gives receiving stations a known pattern on which to lock. The start frame delimiter, which signifies the beginning of the frame, follows the preamble. The 48-bit destination address is next, followed by the 48-bit source address. Original versions of the IEEE 802.3 specification allowed 16-bit addressing; however, this addressing has never been significantly used in the industry.

The next field is the type field in Ethernet and the length field in IEEE 802.3. The type field is used to signify the protocol used in the rest of the frame (e.g., TCP/IP). The length field is used to specify the length of the data portion of the frame. For Ethernet and IEEE 802.3 frames to co-exist on the same LAN, the length field of the frame must always be unique from any type fields used in Ethernet. This has limited the length of the data portion of the frame to 1500 bytes, and therefore the total frame length to 1518 bytes.

The final 4 bytes of the frame are the FCS. This is the standard 32-bit CCITT-CRC polynomial used in many other protocols.

When a station wishes to transmit, it checks for activity on the LAN. When the LAN becomes silent for a specified period, the station begins transmission. During transmission, the station continually checks for collision on the LAN. If a collision is detected, the station forces a jam of all ones on its frame and ceases transmission. Collisions usually occur close to the beginning of a frame. The station then waits a random period of time (backoff) before attempting to transmit again. Once the backoff is complete, the station waits for silence on the LAN and

—Physical—One 48-Bit Address Recognized or 64-Bin Hash Table for Physical Addresses
—Logical—64-Bin Group Address Hash Table plus Broadcast Address Checking
—Promiscuous—Receives All Addresses, but Discards Frame If Reject Pin Asserted

- External CAM Support on Both Serial and System Bus Interfaces
- Up to Eight Parallel I/O Pins May Be Sampled and Appended to Any Frame
- Heartbeat Indication
- Transmitter Network Management and Diagnostics

—Lost Carrier Sense
—Underrun
—Number of Collisions Exceeded the Maximum Allowed
—Number of Retries per Frame
—Deferred Frame Indication
—Late Collision

- Receiver Network Management and Diagnostics

—CRC Error Indication
—Nonoctet Alignment Error
—Frame Too Short
—Frame Too Long
—Overrun
—Busy (Out of Buffers)

- Error Counters

—Discarded Frames (Out of Buffers or Overrun Occurred)
—CRC Errors
—Alignment Errors

- Internal and External Loopback Mode

**7.10.23.3 LEARNING ETHERNET ON THE QUICC.** The following paragraphs detail the Ethernet functionality on the QUICC. However, they show the additions made to the standard SCC functionality to implement Ethernet. Therefore, the reader is encouraged to learn the basics of the SCCs and the overall architecture of the CPM before attempting to learn this section in great detail.

A first-time user of the QUICC who plans to use Ethernet on the QUICC should first read the following sections of this user manual.

1. 7.1 RISC Controller**,** 7.2 Command Set, and 7.3 Dual-Port RAM. The RISC controller is used to issue special commands to the Ethernet channel. The dual-port RAM is used to load Ethernet parameters and initialize BDs for use by the Ethernet channel.
2. 7.7 SDMA Channels discusses how SDMA channels are used to transfer data to/from the Ethernet channel and system memory.
3. 7.8.9 NMSI Configuration explains how clocks are routed to the SCCs through the bank of clocks.
4. 7.10.1 SCC Overview contains more detailed information on the SCCs that are appli-

W—Wrap (Final BD in Table)

0 = This is not the last buffer descriptor in the Rx BD Table.
1 = This is the last buffer descriptor in the Rx BD Table. After this buffer has been used, the CP will receive incoming data into the first BD in the table (the BD pointed to by RBASE). The number of Rx BDs in this table is programmable, and is determined only by the wrap bit and the overall space constraints of the dual-port RAM.

I—Interrupt

0 = No interrupt is generated after this buffer has been filled.
1 = The RX bit in the Centronics event register will be set when this buffer has been completely filled by the CP, indicating the need for the CPU32+ core to process the buffer. The RX bit can cause an interrupt if it is enabled.

C—Control Character

0 = This buffer does not contain a control character.
1 = This buffer contains a control character. The last byte in the buffer is one of the user defined control characters.

CM—Continuous Mode

0 = Normal Operation.
1 = The E-bit is not cleared by the CP after this buffer is closed, allowing the associated data buffer to be overwritten automatically when the CP next accesses this BD.

SL—Silence

The buffer was closed due to the expiration of the programmable silence period timer (defined in MAX_SL).

**7.13.8.23 CENTRONICS RECEIVER EVENT REGISTER (PIPE).** When the Centronics Receiver protocol is selected, the SMC2 event register is called the Centronics Receiver event register. It is an 8-bit register which is used to report events recognized by the Centronics channel and generate interrupts. On recognition of an event, the Centronics controller will set its corresponding bit in the Centronics event register.

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | CCR | BSY | CHR | RX |

The Centronics event register is a memory-mapped register that may be read at any time. A bit is cleared by writing a one (writing a zero does not affect a bit's value). More than one bit may be cleared at a time. All unmasked bits must be cleared before the CP will clear the internal interrupt request. This register is cleared at reset.

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. Reception continues as soon as an empty buffer is provided.

HP4–HP0—Highest Priority

These bits specify the 5-bit interrupt number of the single CPIC interrupt source that is to be advanced to the highest priority in the table. These bits may be dynamically modified. To keep the original priority order intact, simply program these bits to 11111.

VBA2–VB0—Vector Base Address

These three bits are concatenated with five bits provided by the CPIC for each specific interrupt source to form an 8-bit interrupt vector number. If these bits are not written, the uninitialized vector (value $0F) is provided for all CPM sources. These bits should not be dynamically modified.

Bits 4–1—Reserved

SPS—Spread Priority Scheme

This bit, which selects the relative SCC priority scheme, may not be changed dynamically.

0 = Grouped. The SCCs are grouped in priority at the top of the table.
1 = Spread. The SCCs are spread in priority throughout the table.

**7.15.5.2 CPM INTERUPT PENDING REGISTER (CIPR).** Each bit in the 32-bit read-write CIPR corresponds to a CPM interrupt source. When a CPM interrupt is received, the CPIC sets the corresponding bit in the CIPR.

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 171 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PC0 | SCC1 | SCC2 | SCC3 | SCC4 | PC1 | TIMER1 | PC2 | PC3 | SDMA | IDMA1 | IDMA2 | — | TIMER2 | R–TT | — |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PC4 | PC5 | — | TIMER3 | PC6 | PC7 | PC8 | — | TIMER4 | PC9 | SPI | SMC1 | SMC2 / PIP | PC10 | PC11 | — |

In a vectored interrupt scheme, the CIPR clears the CIPR bit when the vector number corresponding to the CPM interrupt source is passed during an interrupt acknowledge cycle, unless an event register exists for that interrupt source. (Event registers exist for interrupt sources that have multiple source events. For example, the SCCs have multiple events that can cause an SCC interrupt.)

In a polled interrupt scheme, the user must periodically read the CIPR. When a pending interrupt is handled, the user clears the corresponding bit in the CIPR. (However, if an event register exists, the unmasked event register bits should be cleared instead, causing the CIPR bit to be cleared.) To clear a bit in the CIPR, the user writes a one to that bit. Since the user can only clear bits in this register, bits written as zeros will not be affected. The CIPR is cleared at reset.

**NOTES**

The SCC CIPR bit positions are NOT changed according to the relative priority between SCCs (as determined by the SCxP and SPS bits in the CICR).

No bit in the CIPR is set if the error vector is issued.

QUICC serial interface clock route register (SICR). Note that many more options are now available.

The TCS bit is implemented in the bank of clocks control in the three TxCS bits of the QUICC SICR. Note that many more options are now available.

The EXTC bit becomes the EXTC1–EXTC0 bits in the BRGC. Note that more options are now available. For compatibility, if the EXTC bit was cleared, then the EXTC1–EXTC0 bits should also be cleared.

The WOMS bit gives the MC68302 wired-OR capability on the serial transmit data pin. This can now be implemented (as well as other wired-OR capability) in the QUICC PxODR registers, which can configure a pin to operate as wired-OR, regardless of whether it is connected to the SCC.

The SCM register is implemented using the general SCC mode register (GSMR) and the protocol-specific mode register (PSMR). One GSMR and one PSMR exist for each SCC. The definition of the PSMR differs based on the protocol used.

The two MODE bits have been expanded to four MODE bits in the GSMR to include the new protocols that the QUICC provides.

The ENT bit still exists, but is shifted two bit positions left in the GSMR.

The ENR bit still exists, but is shifted two bit positions left in the GSMR.

The DIAG bits still exist on the QUICC but are shifted left two positions. Normal operation = 00, loopback = 01, and automatic echo = 10 still exist on the QUICC. Software operation = 11 is now implemented by programming the PCPAR, PCDIR, PCODR, and PCINT in the port C parallel I/O section. The new structure for software operation is much more flexible than on the MC68302 and provides hardware-updated, accurate, real-time status. Since the software operation combination is not needed in the QUICC DIAG bits in the GSMR, this combination is used to support a new feature, simultaneous loopback and echo.

The rest of the SCM bits differ based on protocol and are discussed in the following paragraphs by protocol.

The UART mode register is now implemented using the GSMR and the PSMR. One GSMR and one PSMR exist for each SCC. The definition of the PSMR differs based on the protocol used.

The SL bit is located in the PSMR.

The RTSM bit is located in the GSMR.

The CL bit becomes two bits in the PSMR for more character length options.

The FRZ bit is located in the PSMR.

The UM bits are located in the PSMR. Note that the 10 combination for asynchronous DDCMP is now reserved.

The PEN bit is located in the PSMR.

The one RPM bit becomes two RPM bits in the PSMR for more receive parity options.

**9.4.2.4 FLASH EPROM.** Figure 9-11 shows the interface to flash EPROM devices. In this design, the assumption is made that only the MC68EC040 will access this array. The inverter is only required if DRAM is used elsewhere in the system, since the $\overline{OE}$ function is lost when the AMUX pin is used. This design assumes that the write operations are $\overline{CE}$ controlled, rather than $\overline{WE}$ controlled. Most flash EPROM manufacturers now support this alternative timing method.

**9.4.2.5 REGULAR SRAM.** Figure 9-12 shows the interface to SRAM. In this design, both the MC68EC040 and the QUICC may access the SRAM array. The inverter is only required if DRAM is used elsewhere in the system, since the $\overline{OE}$ function is lost when the AMUX pin is used. This design also allows the QUICC to support bursting by the MC68EC040 using the BADD3–BADD2 signals. The QUICC also uses these signals as A3–A2 address lines during its normal SRAM accesses. If MC68EC040 bursting is not supported, the BADD3–BADD2 signals can be replaced with the A3–A2 signals.

**9.4.2.6 BURST SRAM.** Figure 9-13 shows the interface to Motorola's 32K $\times$ 9 MCM62940A bursting SRAM. This can provide better memory throughput speeds than the regular SRAM array. Figure 9-13 shows the solution for an array that is accessed by the MC68EC040 and the QUICC.

To speed access times, the chip select pin (S0) of the burst SRAM is connected directly to the A27 pin of the system bus. This gives half of the usable address space to this array, although this should not be a problem in most systems considering the 28 address pins available. No chip select pin from the QUICC is used.

The inverter on the R/$\overline{W}$ signal is only required if DRAM is used elsewhere in the system, since the $\overline{OE}$ function is lost when the AMUX pin is used. The CLKO1 signal is derived directly from the QUICC CLKO1 pin. Parity is also supported in the array, using the PRTY3–PRTY0 signals on the system bus.

Although not used in this design, Motorola also offers a larger version of the MCM62940A, called the MCM67M618, that is organized into a 64K $\times$ 18 array.

In Figure 9-13, the $\overline{TSC}$ pin is connected to the $\overline{AS}$ pin to handle the QUICC accesses. The QUICC accesses do not use the bursting feature of the MCM62940A. $\overline{BAA}$, which could be asserted during the QUICC accesses due to the QUICC $\overline{DSACK1}$ being multiplexed with the MC68EC040 $\overline{TA}$, is a don't care as long as $\overline{TSC}$ remains low. After $\overline{TSC}$ negated, the $\overline{OE}$ and $\overline{WE}$ signals are negated, disabling the MC62940A burst

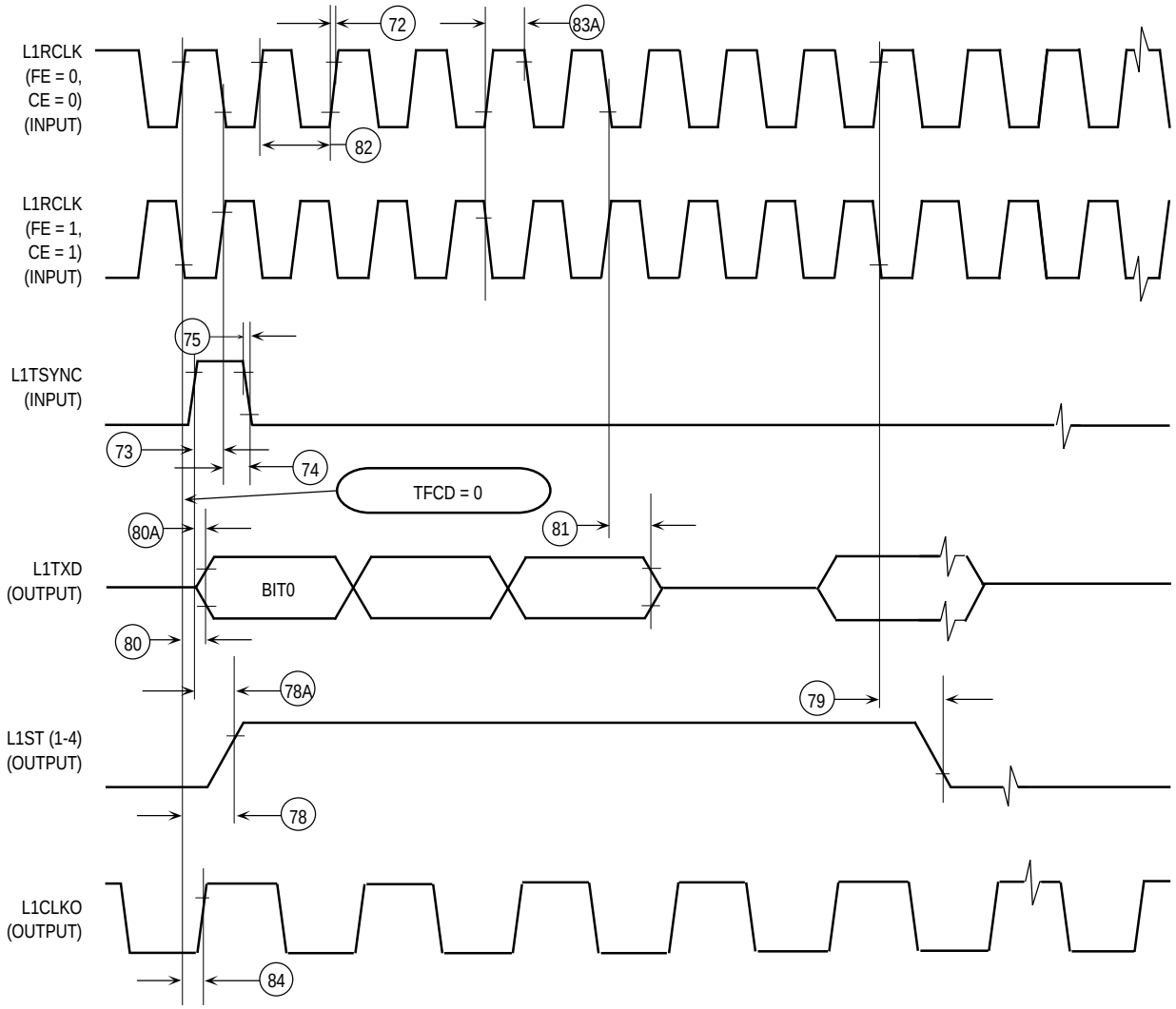



**Figure 10-63. SI Transmit Timing with Double Speed Clocking (DSC = 1)**

| High-Speed Channels | | Low-Speed Channels | | Comments/Restrictions |
|---|---|---|---|---|
| Number | Speed | Number | Speed | |
| 1 HDLC | 8 Mbps | — | — | |
| 2 HDLC | 4 Mbps | — | — | |
| 3 HDLC | 2.6 Mbps | — | — | |
| 4 HDLC | 2.05 Mbps | — | — | |
| 1 ENET | 10 Mbps | 3 HDLC | 2.05 Mbps | Minor restrictions on HDLC buffer length |
| 1 ENET | 10 Mbps | 2 HDLC | 2.05 Mbps | |
| 2 ENET | 10 Mbps | 1 HDLC | 1 Mbps | |
| 4 UART | 625 kbps | — | — | Async SCC |
| 4 UART-S | 625 kbps | — | — | Sync UART SCC |
| 4 BISYNC | 460 kbps | — | — | |
| 1 TRAN | 8 Mbps | — | — | |
| 2 TRAN | 4 Mbps | — | — | |
| 3 TRAN | 2.6 Mbps | — | — | |
| 4 TRAN | 2.05 Mbps | — | — | |
| 2 TRAN | 1.56 Mbps | — | — | SMC Channels |
| 2 UART | 100 kbps | — | — | SMC Channels |

NOTES:

1. These numbers are estimates generated prior to silicon. Additional work will be performed to improve the accuracy of the SCC performance numbers and will be reported in later revisions of this manual.
2. All performance calculations assume a 25-MHz system clock and sync clock for the SCCs. The results scale linearly with different system clock speeds. A change in the sync clock will not affect performance as long as the required 1:2.25 or 1:2.5 ratio is maintained.
3. User should consult with receive and transmit clock timing of the SIA to ensure clock pulse width high and width low are satisfied for high speed serial communications.
4. In all cases, the numbers assume data is stored in external system RAM.
5. The performance is not expected to degrade based on the number of wait states in the system, as long as the system RAM has between 0 and 9 wait states.
6. The performance table is also applicable when the time-slot assigner on the QUICC is used.
7. When the performance of a high-speed channel together with a low-speed channel was measured, the high-speed channel was always SCC1.
8. Performance in *HDLC bus* mode is the same as HDLC performance, for both full duplex and half duplex operation. (Half duplex is the normal configuration of HDLC bus mode, thus HDLC half duplex performance numbers would normally be used.)
9. All results assume that the other RISC features are not operating—i.e., the RISC timer tables, the IDMA auto buffer and buffer chaining modes, the parallel interface port, and the other serial channels. If these features are operating, performance may be slightly reduced. The DRAM refresh controller has no effect on the performance.
10. Except for Ethernet, all table results assume continuous full-duplex operation. Results for half-duplex operation are roughly 2x better.
11. The SMC performance results are without the SCCs operating; otherwise, SMC performance may be reduced. Although the exact SMC performance that can be obtained is determined by many RISC utilization factors and is therefore difficult to estimate, it is expected that 9.6 kbps on both SMC UARTs could be simultaneously supported in most situations.

# APPENDIX B
# DEVELOPMENT TOOLS AND SUPPORT

Several software development packages are offered as a set of independent modules that provide the following features:

- QUICC Chip Evaluation
- By running the modules on the development board described in B.4 M68360QUADS Development System, it is possible to examine and evaluate the QUICC. Symbolic, user-friendly menus allow control of all parts of the QUICC.
- QUICC Simple Drivers
- Written in C, the source code of the QUICC drivers is available on the Motorola Freeware Bulletin Board (512-891-3733, 8 data, no parity, 1 stop) or through Anonymous FTP to freeware.aus.sps.mot.com under /pub/mcu360. These drivers were developed to support the needs of the general user. Although they are based on the regular QUICC drivers, they provide call-oriented interfaces and self-contained QUICC initialization routines and interrupt handling for a given protocol.
- QUICC Chip Drivers
- Written in C, the source code of the QUICC drivers is available on electronic media. These drivers were developed to support the needs of the protocol implementations.
- Protocol Implementations
- Modules implementing common ISO/OSI layer 2 and 3 protocols are available under license. These are in the form of source code written in C.
- Portability
- All of the higher layer software modules may be ported from source to different QUICC implementations and combined with user-developed code. Software interface documentation is available for all provided modules.

## B.1 MOTOROLA SOFTWARE MODULES

Chip driver routines written in C illustrate initialization of the QUICC, interrupt handling, and the management of data transmission and reception on all channels.

In addition to the chip drivers, protocol modules are provided. Layer 2 modules include LAPB and LAPD. The layer 3 module is the X.25 packet layer protocol.

Since the modules require some minimal operating system services, the EDX operating system kernel is provided. EDX is the kernel implemented on the QUADS board (see B.4 M68360QUADS Development System). Use of EDX with the protocol modules is not required as long as some other operating system support is provided by the user.