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	3.3V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	357-BBGA
Supplier Device Package	357-PBGA (25x25)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68360zq25vlr2

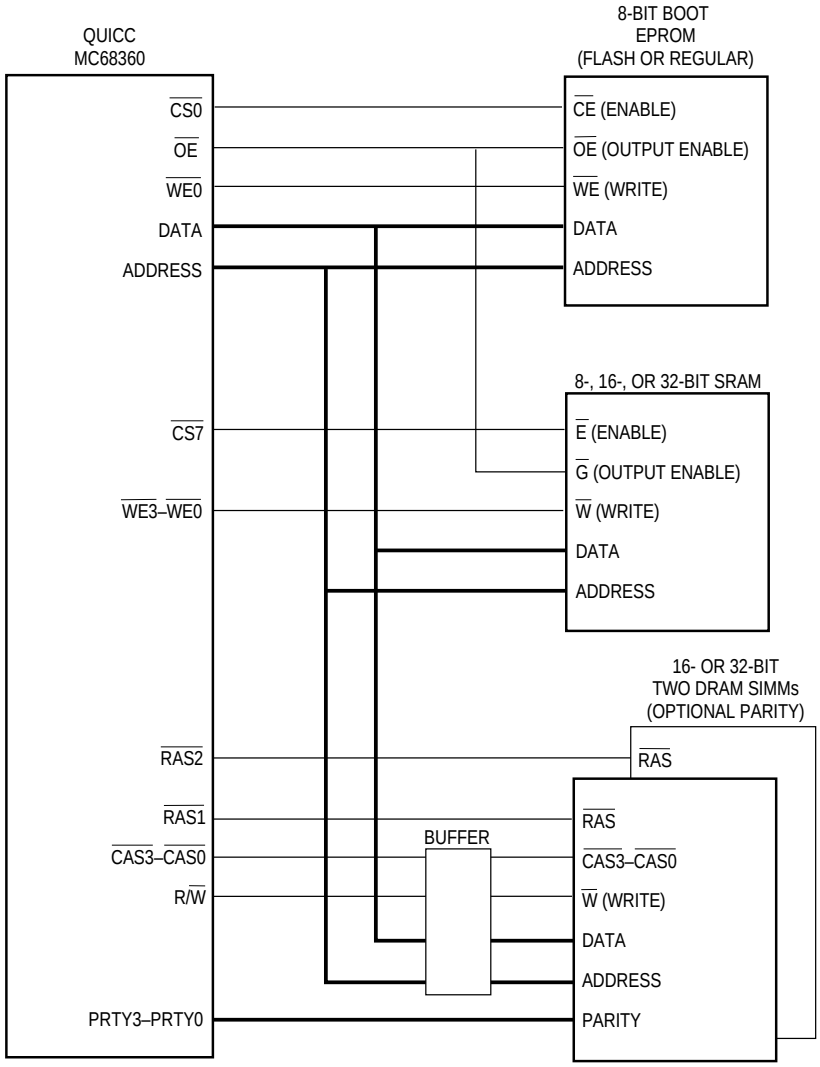


Figure 1-3. Larger QUICC System Configuration

1.5 QUICC SERIAL CONFIGURATIONS

The QUICC offers an extremely flexible set of communications capabilities. Although a full understanding of the possibilities requires reading the appropriate sections, some of the possibilities are shown in the following diagrams. They show possible connections between QUICC devices. In addition, connections are often shown between QUICCs and the MC68302 to show the compatibility between these devices.

For readability, transceivers are usually omitted in the following diagrams. For local on-board communications, however, transceivers are often optional and depend on the protocol used.

Figure 1-4 shows the Ethernet LAN capability of the QUICC. An external SIA transceiver is required to complete the interface to the media. This functionality is implemented in the MC68160 enhanced Ethernet serial transceiver (EEST™). The MC68160 EEST supports

Freescale Semiconductor, Inc.

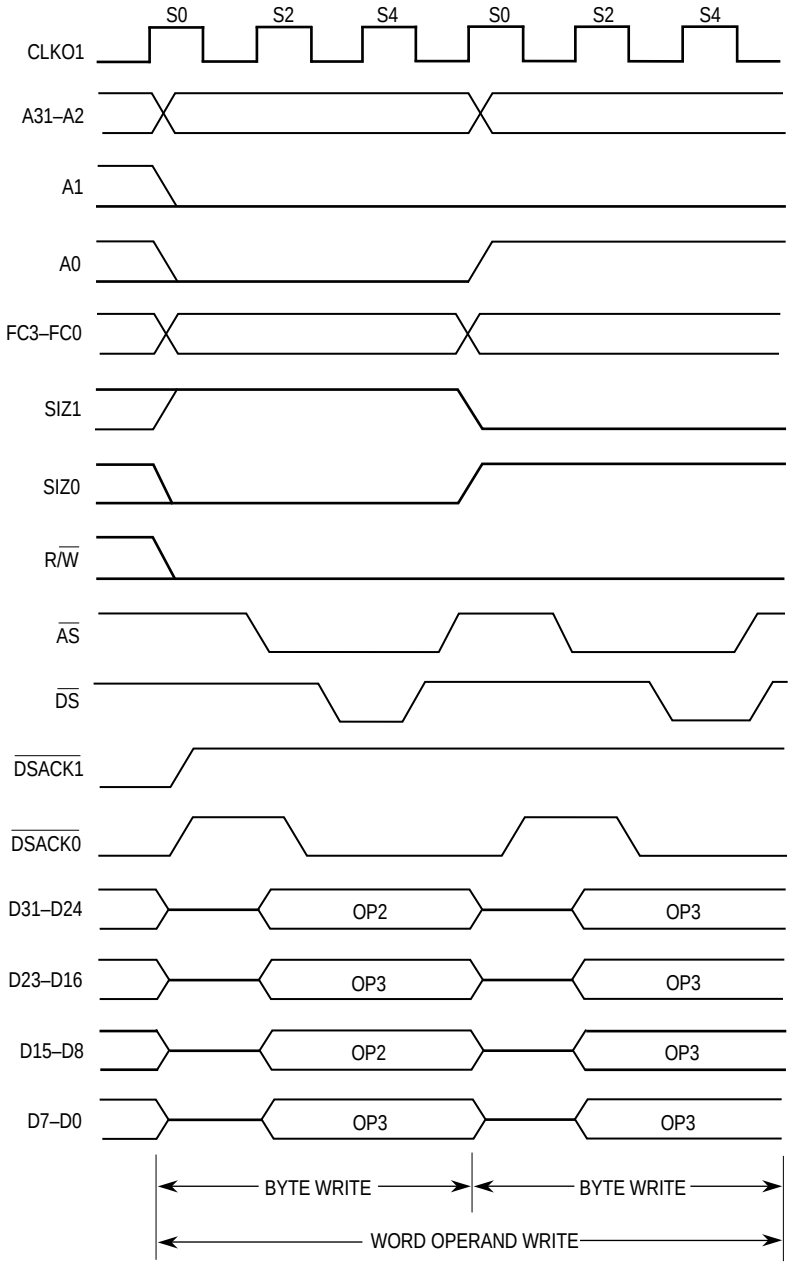
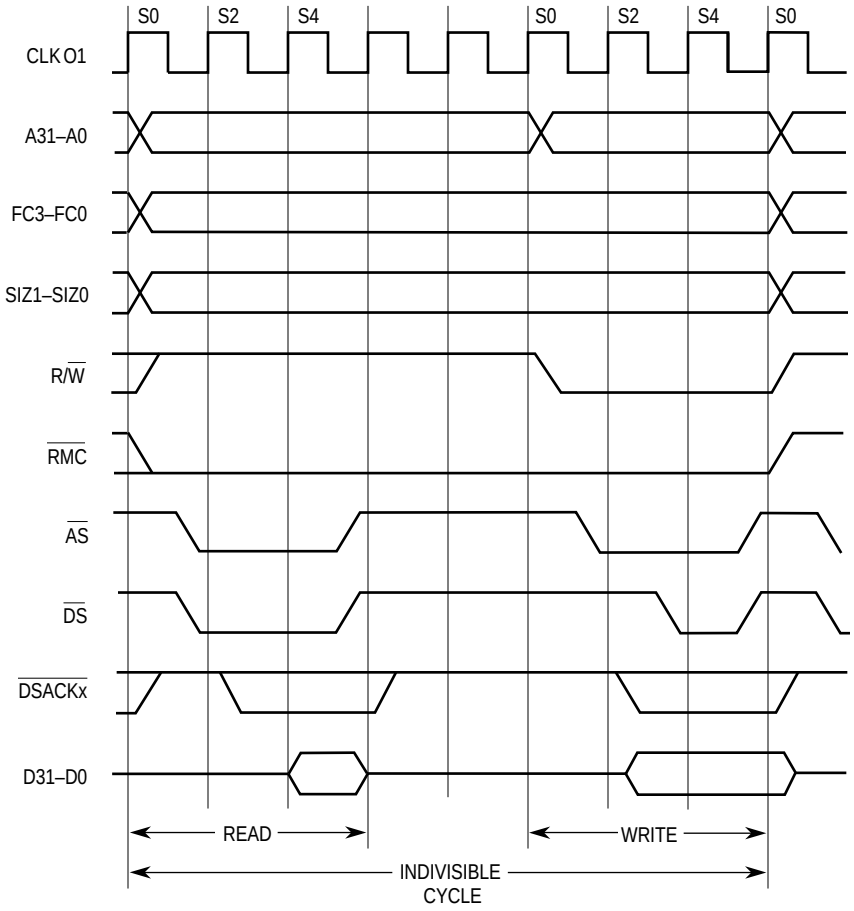


Figure 4-7. Word Operand Write Timing (8-Bit Data Port)

Figure 4-8 shows the transfer of a long-word operand to an odd address in word-organized memory, which requires three bus cycles. For the first cycle, the SIZx signals specify a long-word transfer, and the address offset (A2–A0) is 001. Since the port width is 16 bits, only the first byte of the long word is transferred. The slave device latches the byte and acknowledges the data transfer, indicating that the port is 16 bits wide. When the processor starts the second cycle, the SIZx signals specify that three bytes remain to be transferred with an address offset (A2–A0) of 010. The next two bytes are transferred during this cycle. The processor then initiates the third cycle, with the SIZx signals indicating one byte remaining to be transferred. The address offset (A2–A0) is now 100; the port latches the final byte, and the operation is complete. Figure 4-9 shows the associated bus transfer signal timing.

Freescale Semiconductor, Inc.



NOTE: OE and WE3–WE0 are not shown.

Figure 4-21. Read-Modify-Write Cycle Timing

State 0—The QUICC asserts RMC in S0 to identify a read-modify-write cycle. The QUICC places a valid address on A31–A0 and valid function codes on FC3–FC0. The function codes select the address space for the operation. SIZ1 and SIZ0 become valid in S0 to indicate the operand size. The QUICC drives R/W high for the read cycle.

State 1—One-half clock later in S1, the QUICC asserts AS, indicating a valid address on the address bus. The QUICC also asserts OE and DS during S1.

State 2—The selected device uses OE, R/W, SIZ1, SIZ0, A0, and DS to place information on the data bus. Any of the bytes (D31–D24, D23–D16, D15–D8, and D7–D0) are selected by SIZ1, SIZ0, A1, and A0. Concurrently, the selected device may assert DSACKx.

State 3—As long as at least one of the DSACKx signals is recognized by the end of S2 (meeting the asynchronous input setup time requirement), data is latched on the next falling edge of the clock, and the cycle terminates. If DSACKx is not recognized by the start of S3, the QUICC inserts wait states instead of proceeding to S4 and S5. To ensure that wait states are inserted, both DSACK1 and DSACK0 must remain negated throughout the asynchro-

Freescale Semiconductor, Inc.

interrupt acknowledge cycle internally, the spurious interrupt monitor generates an internal bus error signal to terminate the vector acquisition. The QUICC automatically generates the spurious interrupt vector number, 24, instead of the interrupt vector number in this case. When an external device does not respond to an interrupt acknowledge cycle with AVEC or DSACKx, a bus monitor must assert BERR, which results in the CPU32+ taking the spurious interrupt vector. If HALT is also asserted, the QUICC retries the interrupt acknowledge cycle instead of using the spurious interrupt vector.

4.5 BUS EXCEPTION CONTROL CYCLES

The bus architecture requires assertion of DSACKx from an external device to signal that a bus cycle is complete. Neither DSACKx nor AVEC is asserted in the following cases:

1. DSACKx in fast-termination cycles.
2. AVEC when programmed to respond internally.
3. The external device does not respond.
4. Various other application-dependent errors occur.

The QUICC provides BERR when no device responds by asserting DSACKx/AVEC within an appropriate period of time after the QUICC asserts AS. This mechanism allows the cycle to terminate and the QUICC to enter exception processing for the error condition. HALT is also used for bus exception control. This signal can be asserted by an external device for debugging purposes to cause single bus cycle operation or, in combination with BERR, a retry of a bus cycle in error. To properly control termination of a bus cycle for a retry or a bus error condition, DSACKx, BERR, and HALT can be asserted and negated with the rising edge of the QUICC clock. This assures that when two signals are asserted simultaneously, the required setup and hold time for both is met for the same falling edge of the QUICC clock. This or an equivalent precaution should be designed into the external circuitry to provide these signals. Alternatively, the internal bus monitor could be used. The acceptable bus cycle terminations for asynchronous cycles are summarized in relation to DSACKx assertion as follows (case numbers refer to Table 4-8):

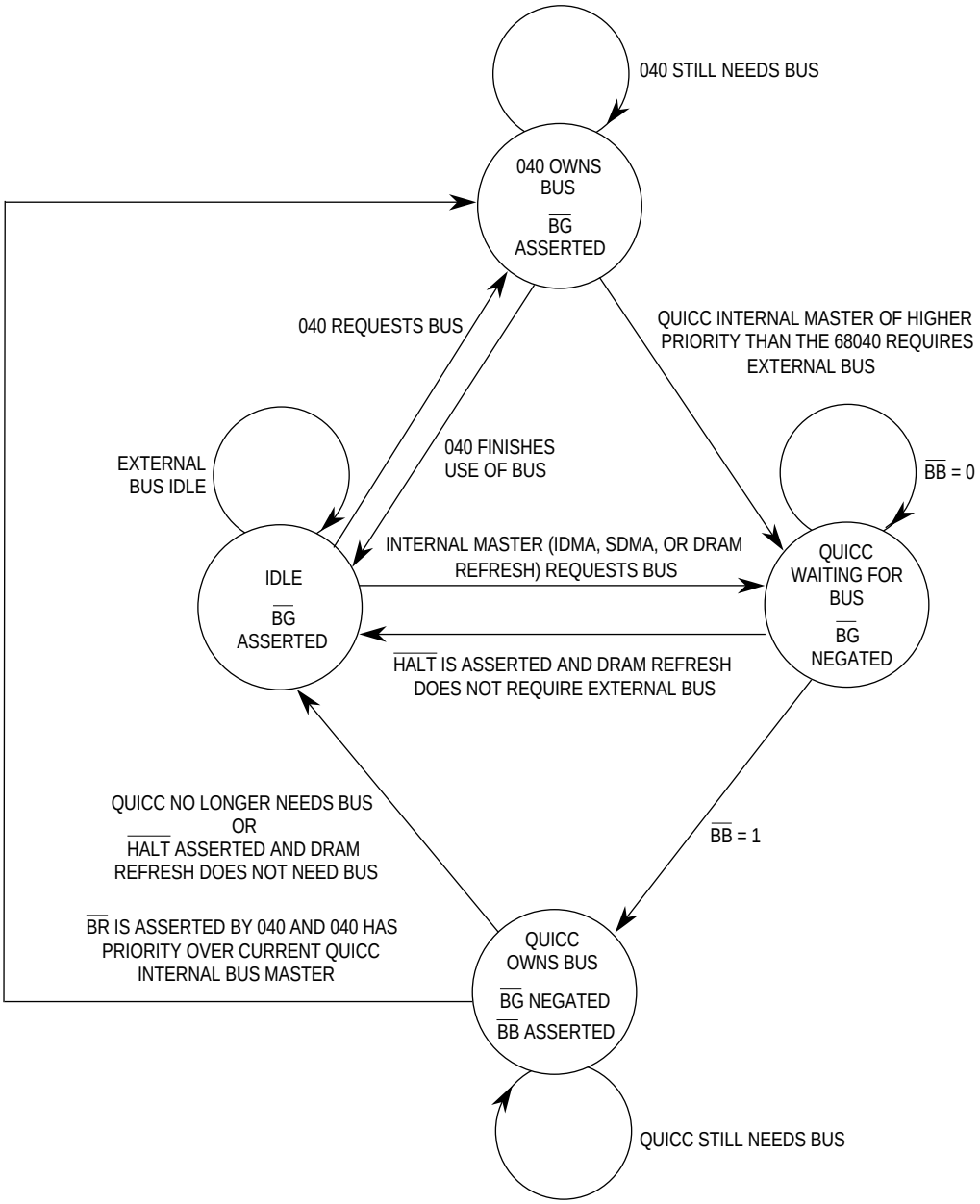
1. Normal Termination: DSACKx is asserted; BERR and HALT remain negated (case 1).
2. Halt Termination: HALT is asserted at the same time or before DSACKx, and BERR remains negated (case 2).
3. Bus Error Termination: BERR is asserted in lieu of, at the same time, or before DSACKx (case 3) or after DSACKx (case 4), and HALT remains negated; BERR is negated at the same time or after DSACKx.
4. Retry Termination: HALT and BERR are asserted in lieu of, at the same time, or before DSACKx (case 5) or after DSACKx (case 6); BERR is negated at the same time or after DSACKx, and HALT may be negated at the same time or after BERR.

Table 4-8 shows various combinations of control signal sequences and the resulting bus cycle terminations. To ensure predictable operation, BERR and HALT should be negated according to the specifications in Section 10 Electrical Characteristics. DSACKx, BERR, and HALT may be negated after AS. If DSACKx or BERR remain asserted into S2 of the next bus cycle, that cycle may be terminated prematurely.

Freescale Semiconductor, Inc.

3. If the 68040 requests the bus at the same time that a QUICC internal master is requesting the bus, the BR040ID bits are used to determine who will acquire the bus first.
4. When the QUICC no longer needs the bus, it deasserts $\overline{BB}$ and asserts $\overline{BG}$.

The state machine for the MC68040 companion mode arbitration is shown in Figure 4-39.



NOTES:
1. If the 68040 and the QUICC Internal Master requests the bus at the same time, the highest priority requester wins.
2. The transition from "040 Owns Bus" to "QUICC Waiting for Bus" may be delayed, until the write portion of an 040 locked cycle if an 040 locked cycle is in progress when the higher priority QUICC internal master requests the bus.
3. $\overline{BB}$ is only asserted by QUICC during the state "QUICC Owns Bus", otherwise $\overline{BB}$ is three-stated by the QUICC.

Figure 4-39. MC68040 Companion Mode Bus Arbitration State Machine

Freescale Semiconductor, Inc.

5.3.1 M68000 Family Compatibility

It is the philosophy of the M68000 Family that all user-mode programs should execute unchanged on a more advanced processor and that supervisor-mode programs and exception handlers should require only minimal alteration.

The CPU32+ can be thought of as an intermediate member of the M68000 family. Object code from an MC68000 or MC68010 may be executed on the CPU32+, and many of the instruction and addressing mode extensions of the MC68020 are also supported.

5.3.1.1 NEW INSTRUCTIONS. Two instructions have been added to the M68000 instruction set: LPSTOP and TBL.

5.3.1.2 LOW-POWER STOP (LPSTOP). In applications where power consumption is a consideration, the CPU32+ can force the device into a low-power standby mode when immediate processing is not required. The low-power mode is entered by executing the LPSTOP instruction. The processor remains in this mode until a user-specified or higher level interrupt or a reset occurs.

5.3.1.3 TABLE LOOKUP AND INTERPOLATE (TBL). To maximize throughput for real-time applications, reference data is often precalculated and stored in memory for quick access. The storage of sufficient data points can require an inordinate amount of memory. The TBL instruction uses linear interpolation to recover intermediate values from a sample of data points, thus conserving memory.

When the TBL instruction is executed, the CPU32+ looks up two table entries bounding the desired result and performs a linear interpolation between them. Byte, word, and long-word operand sizes are supported. The result can be rounded according to a round-to-nearest algorithm or returned unrounded along with the fractional portion of the calculated result (byte and word results only). This extra precision can be used to reduce cumulative error in complex calculations. See 5.3.4 Using the TBL Instructions for examples.

5.3.1.4 UNIMPLEMENTED INSTRUCTIONS. The ability to trap on unimplemented instructions allows user-supplied code to emulate unimplemented capabilities or to define special-purpose functions. However, Motorola reserves the right to use all currently unimplemented instruction operation codes for future M68000 enhancements. See 5.5.2.8 Illegal or Unimplemented Instructions for more details.

5.3.2 Instruction Format and Notation

All instructions consist of at least one word. Some instructions can have as many as seven words, as shown in Figure 5-6. The first word of the instruction, called the operation word, specifies instruction length and the operation to be performed. The remaining words, called extension words, further specify the instruction and operands. These words may be immediate operands, extensions to the effective address mode specified in the operation word, branch displacements, bit number, special register specifications, trap operands, or argument counts.

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

Table 5-11. System Control Operations

Instruction	Operand Syntax	Operand Size	Operation
Privileged			
ANDI	#⟨data⟩, SR	16	Immediate Data Λ SR ⇒ SR
EORI	#⟨data⟩, SR	16	Immediate Data ⊕ SR ⇒ SR
MOVE	⟨ea⟩, SR SR, ⟨ea⟩	16 16	Source ⇒ SR SR ⇒ Destination
MOVEA	USP, An An, USP	32 32	USP ⇒ An An ⇒ USP
MOVEC	Rc, Rn Rn, Rc	32 32	Rc ⇒ Rn Rn ⇒ Rc
MOVES	Rn, ⟨ea⟩ ⟨ea⟩, Rn	8, 16, 32	Rn ⇒ Destination using DFC Source using SFC ⇒ Rn
ORI	#⟨data⟩, SR	16	Immediate Data V SR ⇒ SR
RESET	none	none	Assert $\overline{\text{RESET}}$ line
RTE	none	none	(SP) ⇒ SR; SP + 2 ⇒ SP; (SP) ⇒ PC; SP + 4 ⇒ SP; restore stack according to format
STOP	#⟨data⟩	16	Immediate Data ⇒ SR; STOP
LPSTOP	#⟨data⟩	none	Immediate Data ⇒ SR; interrupt mask ⇒ EBI; STOP
Trap Generating			
BKPT	#⟨data⟩	none	If breakpoint cycle acknowledged, then execute returned operation word, else trap as illegal instruction.
BGND	none	none	If background mode enabled, then enter background mode, else format/vector offset ⇒ – (SSP); PC ⇒ – (SSP); SR ⇒ – (SSP); (vector) ⇒ PC
CHK	⟨ea⟩, Dn	16, 32	If Dn < 0 or Dn < (ea), then CHK exception
CHK2	⟨ea⟩, Rn	8, 16, 32	If Rn < lower bound or Rn > upper bound, then CHK exception
ILLEGAL	none	none	SSP – 2 ⇒ SSP; vector offset ⇒ (SSP); SSP – 4 ⇒ SSP; PC ⇒ (SSP); SSP – 2 ⇒ SSP; SR ⇒ (SSP); Illegal instruction vector address ⇒ PC
TRAP	#⟨data⟩	none	SSP – 2 ⇒ SSP; format/vector offset ⇒ (SSP); SSP – 4 ⇒ SSP; PC ⇒ (SSP); SR ⇒ (SSP); vector address ⇒ PC
TRAPcc	none #⟨data⟩	none 16, 32	If cc true, then TRAP exception
TRAPV	none	none	If V set, then overflow TRAP exception
Condition Code Register			
ANDI	#⟨data⟩, CCR	8	Immediate Data Λ CCR ⇒ CCR
EORI	#⟨data⟩, CCR	8	Immediate Data ⊕ CCR ⇒ CCR
MOVE	⟨ea⟩, CCR CCR, ⟨ea⟩	16 16	Source ⇒ CCR CCR ⇒ Destination
ORI	#⟨data⟩, CCR	8	Immediate Data V CCR ⇒ CCR

5.3.3.10 CONDITION TESTS. Conditional program control instructions and the TRAPcc instruction execute on the basis of condition tests. A condition test is the evaluation of a logical expression related to the state of the CCR bits. If the result is 1, the condition is true. If

Freescale Semiconductor, Inc.

available at the supervisor level, but execution of some instructions is not permitted at the user level. There are separate SPs for each level. The S-bit in the SR indicates privilege level and determines which SP is used for stack operations. The processor identifies each bus access (supervisor or user mode) via function codes to enforce supervisor and user access levels.

In a typical system, most programs execute at the user level. User programs can access only their own code and data areas and are restricted from accessing other information. The operating system executes at the supervisor privilege level, has access to all resources, performs the overhead tasks for the user level programs, and coordinates their activities.

5.4.2.1 SUPERVISOR PRIVILEGE LEVEL. If the S-bit in the SR is set, supervisor privilege level applies, and all instructions are executable. The bus cycles generated for instructions executed in supervisor level are normally classified as supervisor references, and the values of the function codes on FC2–FC0 refer to supervisor address spaces.

All exception processing is performed at the supervisor level. All bus cycles generated during exception processing are supervisor references, and all stack accesses use the SSP.

Instructions that have important system effects can only be executed at supervisor level. For instance, user programs are not permitted to execute STOP, LPSTOP, or RESET instructions. To prevent a user program from gaining privileged access, except in a controlled manner, instructions that can alter the S-bit in the SR are privileged. The TRAP #n instruction provides controlled user access to operating system services.

5.4.2.2 USER PRIVILEGE LEVEL. If the S-bit in the SR is cleared, the processor executes instructions at the user privilege level. The bus cycles for an instruction executed at the user privilege level are classified as user references, and the values of the function codes on FC2–FC0 specify user address spaces. While the processor is at the user level, implicit references to the system SP and explicit references to address register seven (A7) refer to the USP.

5.4.2.3 CHANGING PRIVILEGE LEVEL. To change from user privilege level to supervisor privilege level, a condition that causes exception processing must occur. When exception processing begins, the current values in the SR, including the S-bit, are saved on the supervisor stack, and then the S-bit is set to enable supervisor access. Execution continues at supervisor privilege level until exception processing is complete.

To return to user access level, a system routine must execute one of the following instructions: MOVE to SR, ANDI to SR, EORI to SR, ORI to SR, or RTE. These instructions execute only at supervisor privilege level and can modify the S-bit of the SR. After these instructions execute, the instruction pipeline is flushed, then refilled from the appropriate address space.

The RTE instruction causes a return to a program that was executing when an exception occurred. When RTE is executed, the exception stack frame saved on the supervisor stack can be restored in either of two ways.

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

ue should be less than the SDMA arbitration ID so that the SDMA channels have priority over the IDMA channels. User must program this field to 7 when the QUICC is configured in slave mode.

7.6.2.2 CHANNEL MODE REGISTER (CMR). Each IDMA channel contains a 16-bit CMR that is reset to $0000. It is used to configure most of the IDMA options.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECO	SRM	S/D	RCI	REQG		SAPI	DAPI	SSIZE		DSIZE		BT		RST	STR

ECO — External Control Option

Dual Address Mode: this bit defines which device is connected to the control signals.

0 = The control signals ($\overline{\text{DREQx}}$, $\overline{\text{DACKx}}$, and $\overline{\text{DONEx}}$) are associated with the destination (write) portion of the transfer.

1 = The control signals ($\overline{\text{DREQx}}$, $\overline{\text{DACKx}}$, and $\overline{\text{DONEx}}$) are associated with the source (read) portion of the transfer.

Single Address Mode: this bit defines the direction of the transfer.

0 = The device writes to memory, and the control signals ($\overline{\text{DREQx}}$, $\overline{\text{DACKx}}$, and $\overline{\text{DONEx}}$) are used by the device to provide data during the destination (write) portion of the transfer.

1 = The device reads from memory, and the control signals ($\overline{\text{DREQx}}$, $\overline{\text{DACKx}}$, and $\overline{\text{DONEx}}$) are used by the device to write data during the source (read) portion of the transfer.

NOTE

If REQG is programmed to be internal (REQG = 0X), $\overline{\text{DREQx}}$ is ignored.

SRM — Synchronous Request Mode

This bit controls how external devices may use the $\overline{\text{DREQx}}$ pin for IDMA service. This bit is only relevant for applications that use external request mode or use the external $\overline{\text{DONEx}}$ pin to terminate the IDMA operation.

0 = Asynchronous request mode is selected. The $\overline{\text{DREQx}}$ and $\overline{\text{DONEx}}$ input signals are internally synchronized to the IDMA clock before they are used by the IDMA.

1 = Synchronous request mode is selected. The $\overline{\text{DREQx}}$ and $\overline{\text{DONEx}}$ input signals are used by the IDMA without first being internally synchronized. This results in faster operation, but should only be used if setup and hold times can be met.

S/D — Single/Dual Address Transfer

0 = The IDMA channel runs standard dual address transfers. Each transfer requires at least two bus cycles. Data packing is performed using the DHR.

1 = The IDMA channel runs single address transfers from a peripheral to memory or from memory to a peripheral. The transfer requires one bus cycle. The DHR is not used for these transfers because the data is transferred directly into the destination location.

Freescale Semiconductor, Inc.

nored. Data can be programmed to appear on the TXD pin, or the TXD pin can remain high by programming the port A register. The $\overline{RTS}$ line can also be programmed to be disabled in the appropriate parallel I/O register. In TDM modes, the L1TXDx and L1RQx lines can be programmed to be either asserted normally or to remain inactive by programming the serial interface mode register (SIMODE).

When using local loopback mode, the clock source for the transmitter and the receiver must be the same. Thus, the same baud rate generator may be used for both transmitter and receiver, or the same external CLKx pin may be used for both transmitter or receiver. (Separate CLKx pins may be used with the transmitter and receiver as long as the CLKx pins are connected to the same external clock signal source.)

01 =Local loopback mode

NOTE

If external loopback is desired, the DIAG bits should be selected for normal operation, and an external connection should be made between the TXD and RXD pins. Clocks may be generated internally by a baud rate generator or generated externally. The user may physically connect the appropriate control signals ($\overline{RTS}$ connected to $\overline{CD}$, and $\overline{CTS}$ grounded) or the port C register may be used to cause the $\overline{CD}$ and $\overline{CTS}$ pins to be permanently asserted to the SCC.

In automatic echo mode, the channel automatically retransmits the received data on a bit-by-bit basis using whatever receive clock is provided. The receiver operates normally and can receive data if $\overline{CD}$ is asserted. The transmitter simply transmits received data. In this mode, the $\overline{CTS}$ line is ignored.

The echo function may also be accomplished in software by receiving buffers from an SCC, linking them to Tx BDs and then transmitting them back out of that SCC.

10 = Automatic echo mode

In loopback/echo mode, loopback operation and echo operation occur simultaneously. The $\overline{CD}$ pin and $\overline{CTS}$ pins are ignored. See the loopback bit description for clocking requirements.

11 = Loopback and echo mode

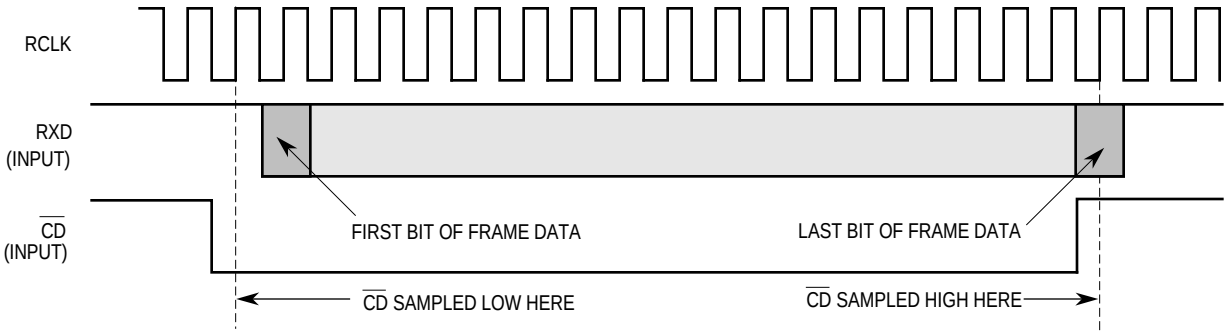
NOTE

Users familiar with the MC68302 may notice that the QUICC does not contain "software operation" mode. The software operation mode as implemented on the MC68302 can be implemented on the QUICC using parallel I/O port C.

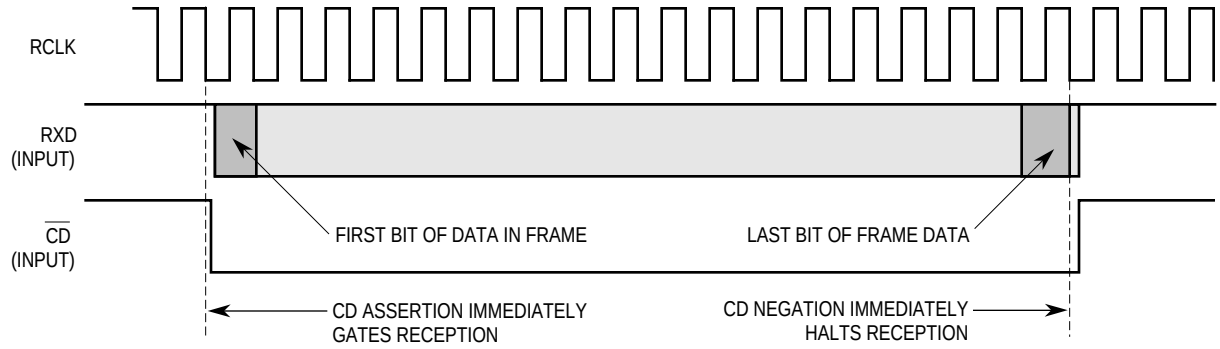
ENR—Enable Receive

This bit enables the receiver hardware state machine for this SCC. When ENR is cleared, the receiver is disabled, and any data in the receive FIFO is lost. If ENR is cleared during reception, the receiver aborts the current character. ENR may be set or cleared regard-

Freescale Semiconductor, Inc.



NOTES:
1. CDS = 0 in GSMR; CDP = 0.
2. If CD is negated prior to the last bit of the receive frame, CD LOST is signaled in the frame BD.
3. If CDP = 1, CD LOST cannot occur, and CD negation has no effect on reception.



NOTES:
1. CDS = 1 in GSMR; CDP = 0.
2. If CD is negated prior to the last bit of the receive frame, CD lost is signaled in the frame BD.
3. If CDP = 1, CD lost cannot occur, and CD negation has no effect on reception.

Figure 7-42. Using CD to Control Reception of Synchronous Protocols

If the CD pin is programmed to envelope the data, the CD pin must remain asserted during frame transmission, or a CD lost error occurs. The negation of the CD pin terminates reception. If the CDS bit in the GSMR is zero, the CD pin must be sampled by the SCC before a CD lost is recognized. Otherwise, the negation of CD immediately causes the CD lost condition.

NOTE

If the CDS bit in GSMR is set, all CD transitions must occur while the receive clock is low.

7.10.11.2 ASYNCHRONOUS PROTOCOLS. The RTS pin is asserted when the SCC data is loaded into the transmit FIFO and a falling transmit clock occurs. The CD and CTS pins may be used to control reception and transmission in the same manner as the synchronous protocols. The first bit of data transmission in an asynchronous protocol is the start bit of the first character. In addition, the UART protocol has an option for CTS flow control as described in 7.10.16 UART Controller.

Freescale Semiconductor, Inc.

is still obtained from the SCC's individual $\overline{\text{CTSx}}$ pin; thus, the $\overline{\text{CTS}}$ pin must be configured in port C to connect to the desired SCC. Since the SCC only receives clocks during its time slot, the $\overline{\text{CTS}}$ pin is only sampled during the transmit clock edges of the SCC's particular time slot.

7.10.18.3 HDLC BUS MEMORY MAP AND PROGRAMMING. HDLC bus on the QUICC is implemented using the HDLC controller with certain bits set. Otherwise, the user should consult the HDLC controller section for detailed information on the programming of HDLC.

7.10.18.3.1 GSMR Programming. The GSMR programming sequence is as follows:

1. Set the MODE bits to HDLC.
2. Set the ENT and ENR bits as desired.
3. Set the DIAG bits for normal operation.
4. Set the RDCR and TDCR bits for 1x clock.
5. Set the TENC and RENC bits for NRZ.
6. Clear RTSM.
7. Set CTSS to one and all other bits to zero or to their default condition.

7.10.18.3.2 PSMR Programming. The PSMR programming sequence is as follows:

1. Set the NOF bits as desired.
2. Set the CRC to 16-bit CRC CCITT.
3. Set the RTE bit.
4. Set the BUS bit.
5. Set the BRM bit to one or zero as desired.
6. Set all other bits to zero or to their default condition.

7.10.18.3.3 HDLC Bus Controller Example. Except for the previously discussed register programming, the HDLC Example #1 may be followed.

7.10.19 AppleTalk Controller

AppleTalk is a set of protocols developed by Apple Computer Inc. to provide a LAN service between Macintosh computers and printers. Although AppleTalk can be implemented over a variety of physical and link layers, including Ethernet, the AppleTalk protocols have traditionally been most closely associated with one particular physical and link layer protocol called LocalTalk.

The term LocalTalk refers to an HDLC-based link layer and physical layer protocol that runs at the rate of 230.4 kbps. In this document, the term AppleTalk controller refers to a support that the QUICC provides for the LocalTalk protocol.

The AppleTalk controller provides the required frame synchronization, bit sequence, preamble, and postamble onto standard HDLC frames. These capabilities, as well as the use of the HDLC controller in conjunction with the DPLL operating in FM0 mode, provide the proper connection formats to the LocalTalk bus.

Freescale Semiconductor, Inc.

Additionally, the CAM control logic may wish to provide additional information on the PB15–PB8 pins. The QUICC Ethernet controller will write this additional byte to memory during the last SDMA write if the SIP bit is set in the PSMR. This information tag is sampled by the QUICC Ethernet controller as the last FCS byte is read from the receive FIFO. The information TAG should be provided by the CAM control logic no later than when RENA is negated at the end of a non-collision frame, and should be held stable on the PB15–PB8 pins until the $\overline{\text{SDACK2}}$–$\overline{\text{SDACK1}}$ pins signal that the tag byte is being written to memory.

The parallel interface option is shown in Figure 7-69. The QUICC outputs two signals every time it writes Ethernet frame data to system memory. The signals SDMA acknowledge ($\overline{\text{SDACK2}}$–$\overline{\text{SDACK1}}$), are asserted during all bus cycles on which Ethernet frame data is written to memory. (These signals are not used for other protocols.)

The CAM control logic uses these pins to enable the CAM writes simultaneously with system memory writes. In this way, the CAM captures the frame data at the same time that it is being written to system memory. The chief advantage of this approach is that the data is already in parallel form when it leaves the QUICC.

The $\overline{\text{SDACK2}}$–$\overline{\text{SDACK1}}$ signals are asserted during all bus cycles writes of the frame data. A certain $\overline{\text{SDACK2}}$–$\overline{\text{SDACK1}}$ combination specifically identifies the first 32-bits of the frame, another identifies all mid-frame data, and a third combination identifies the last 32-bit bus write of the frame (only if the tag byte is appended). The tag byte is appended from the sample of PB15–PB8 if the SIP bit is set in the PSMR. The tag byte will always be in byte 3 of the last 32-bit write. The Rx BD Data Length does not include tag byte in the length calculation.

If the system memory is 32 bits, then the QUICC 32-bit write will take one bus cycle. If the system memory is 16 bits or 8 bits, then the QUICC 32-bit write will take two or four bus cycles. In any case, the $\overline{\text{SDACK2}}$–$\overline{\text{SDACK1}}$ signals are valid on each bus cycle of a 32-bit write cycle and only during bus cycles associated with the Ethernet receiver.

Additionally, the user may choose a unique function code (FC3–FC0) associated with the SDMA receive channel associated with the Ethernet SCC to have an alternate method of identifying accesses from this SCC.

NOTE

The tag byte is always written to byte 3 of the last SDMA write to the buffer, and is not necessarily appended to the last byte of the frame. The Rx BD Data Length does not show the length of the tag byte in the frame. Also the $\overline{\text{SDACK2-1}}$ signals will equal "00" whenever the frame length is not an even multiple of 4 (i.e., it does not depend on whether the tag byte is appended).

Freescale Semiconductor, Inc.

Although RBPTR need never be written by the user in most applications, it may be modified by the user when the receiver is disabled or when the user is sure that no receive buffer is currently in use.

7.11.4.5 TRANSMITTER BUFFER DESCRIPTOR POINTER (TBPTR). The TBPTR for each SMC channel points to the next BD that the transmitter will transfer data from when it is in idle state or to the current BD during frame transmission. After a reset or when the end of the BD table is reached, the CP initializes this pointer to the value programmed in the TBASE entry. 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 a STOP TRANSMIT command is issued, or after a GRACEFUL STOP TRANSMIT command is issued, and the frame completes its transmission.)

7.11.4.6 OTHER GENERAL PARAMETERS. Additional parameters are listed in Table 7-12. 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.

7.11.5 Disabling the SMCs on the Fly

If an SMC 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.

This sequence ensures that any buffers in use will be properly closed and that new data will be transferred to/from a new buffer. Such a sequence is required if the parameters that must be changed are not allowed to be changed dynamically. If the register or bit description states that dynamic (on-the-fly) changes are allowed, the following sequences are not required, and the register or bit may be changed immediately. In all other cases, the sequence should be used. For instance, the baudrate generators allow on-the-fly changes.

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

7.13.2 PIP Overview

The PIP is shown in Figure 7-84. The PIP may be operated as an 18-bit general-purpose I/O port or in one of three handshake modes:

- 8- or 16-Bit Strobed I/O Port with Two Interlocked Handshake Signals
- 8- or 16-Bit Strobed I/O Port with Two Pulsed Handshake Signals
- 8- or 16-Bit Transparent I/O Port with No Handshake Signals

When in one of the handshake modes, the PIP is controlled either by the RISC controller or the CPU32+ core. When the PIP is under RISC control, data is prepared by the CPU32+ core (or other host processor) using the same general BD structures as are used for the SCCs. Thus, the PIP can transfer or receive blocks of characters without interrupting the host processor. The data block may span several linked buffers; therefore, an entire block may be received or transmitted without intervention from the CPU32+ core. When the PIP is under CPU32+ core control (or the control of an external processor), the PIP is controlled directly by the core one byte/word at a time.

When the interlocked or pulsed handshake modes are used, the PIP offers programmable timing attributes such as setup time, pulse width, etc. The interlocked handshake mode supports level-sensitive handshake control signals. The pulsed handshake mode supports edge-sensitive handshakes like those used for the Centronics interface.

The PIP mode of operation may be configured independently for two groups of port B pins: PB7–PB0 and PB17–PB8. This configuration allows an 8-bit PIP data port to be defined, rather than a full 16-bit data port.

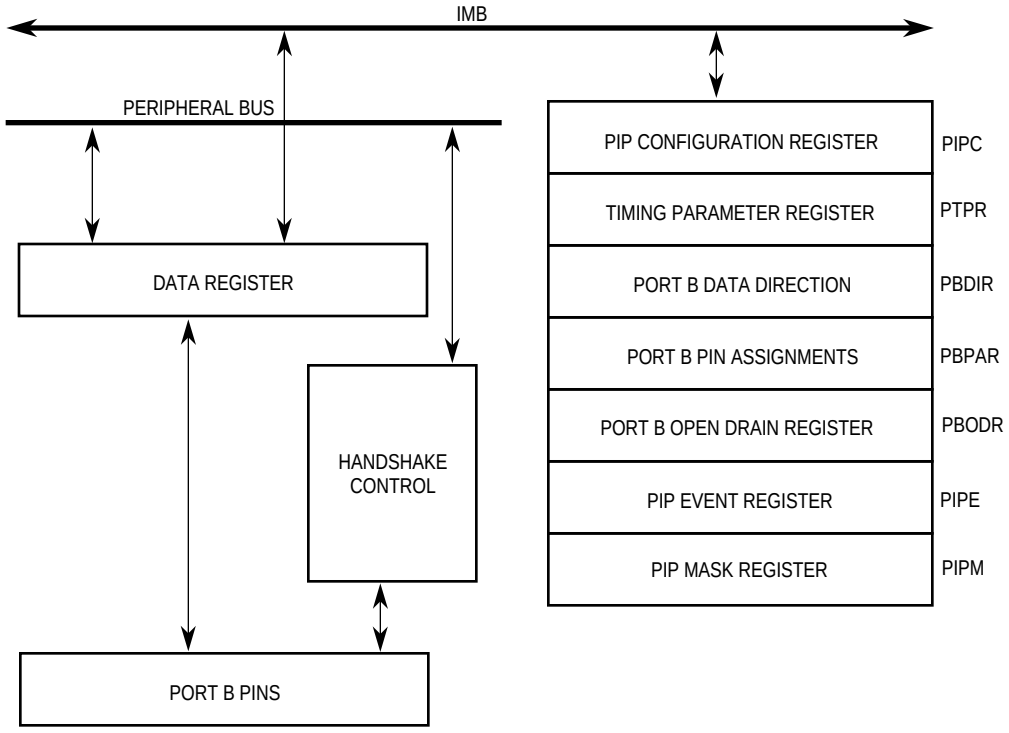


Figure 7-84. PIP Block Diagram

Freescale Semiconductor, Inc.

SCcP—SCCc Priority Order

These two bits define which SCC will assert its request in the SCCc priority position. The user should not program the same SCC to more than one priority position (a, b, c, or d). These bits may be changed dynamically.

00 = SCC1 will assert its request in the SCCc position.
01 = SCC2 will assert its request in the SCCc position.
10 = SCC3 will assert its request in the SCCc position.
11 = SCC4 will assert its request in the SCCc position.

SCbP—SCCb Priority Order

These two bits define which SCC will assert its request in the SCCb priority position. The user should not program the same SCC to more than one priority position (a, b, c, or d). These bits may be changed dynamically.

00 = SCC1 will assert its request in the SCCb position.
01 = SCC2 will assert its request in the SCCb position.
10 = SCC3 will assert its request in the SCCb position.
11 = SCC4 will assert its request in the SCCb position.

SCaP—SCCa Priority Order

These two bits define which SCC will assert its request in the SCCa priority position. The user should not program the same SCC to more than one priority position (a, b, c, or d). These bits may be changed dynamically.

00 = SCC1 will assert its request in the SCCa position.
01 = SCC2 will assert its request in the SCCa position.
10 = SCC3 will assert its request in the SCCa position.
11 = SCC4 will assert its request in the SCCa position.

IRL2–IRL0—Interrupt Request Level

The IRL field contains the priority request level of the interrupt from the CPM that is sent to the CPU32+ core. Level 7 indicates a nonmaskable interrupt; level 0 indicates that all CPM interrupts are disabled. The IRL field, therefore, acts as a master enable for the CPM interrupts in addition to specifying the interrupt priority level. The IRL field is initialized to zero during reset to prevent the CPM from generating an interrupt until this register has been initialized. Value $4 is a good value to choose for the IRL field in most systems.

NOTES

In systems with multiple QUICCs sharing the same system bus, assign these bits to a different request level in each QUICC.

If QUICC is in slave mode (CPU32+ disabled), then the external $\overline{\text{IRQx}}$ pin corresponding to the value programmed in IRL2–IRL0 should not be used. (For example, if IRL2–IRL0 has the value $5, then $\overline{\text{IRQ5}}$ on this QUICC should not be used externally.) This also applies to the programmable interrupt timer and software watchdog in the SIM60 of the slave QUICC.

Freescale Semiconductor, Inc.

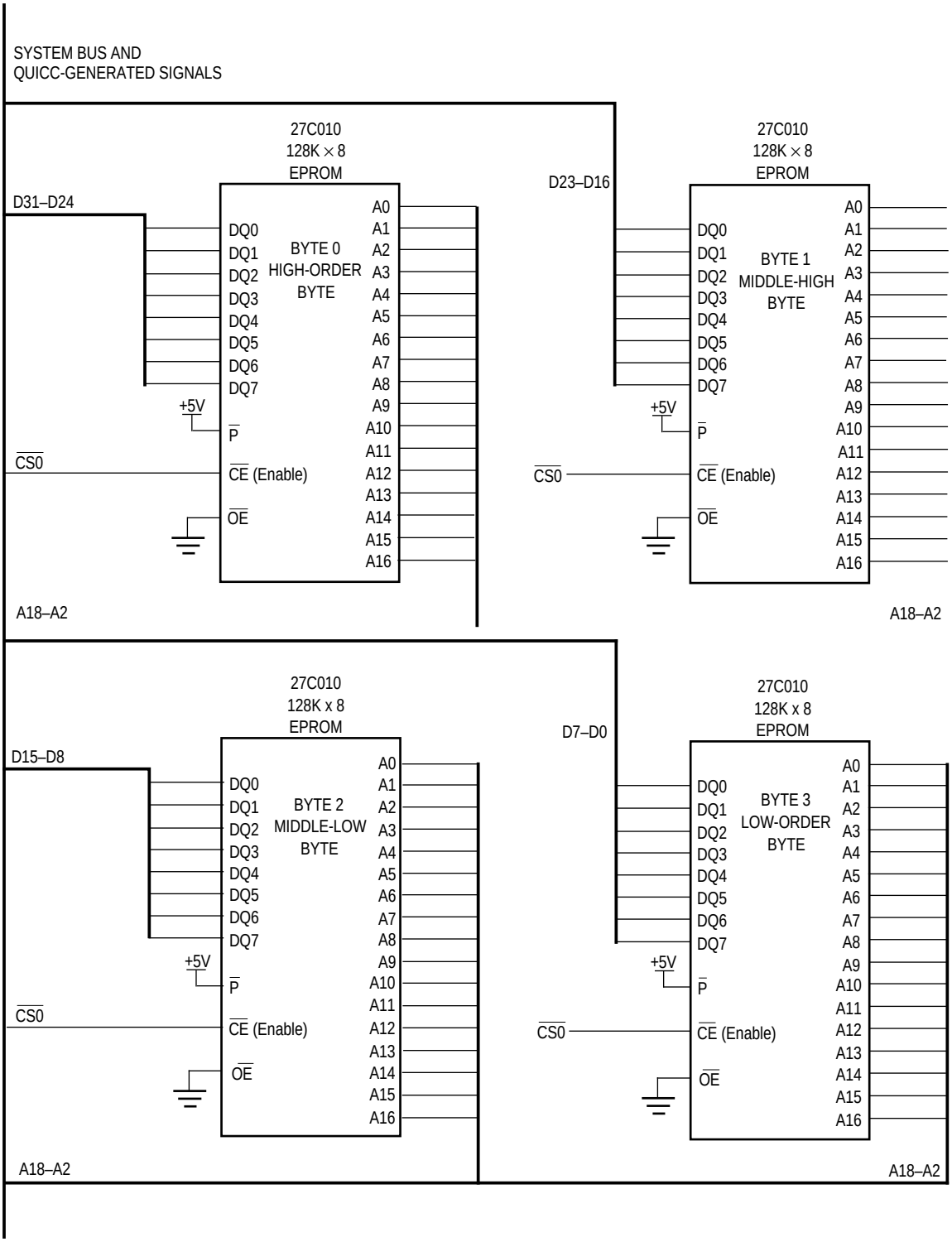


Figure 9-9. 512-Kbyte EPROM Bank—32 Bits Wide

Freescale Semiconductor, Inc.

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. 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 resident 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 (see Figure B-1).

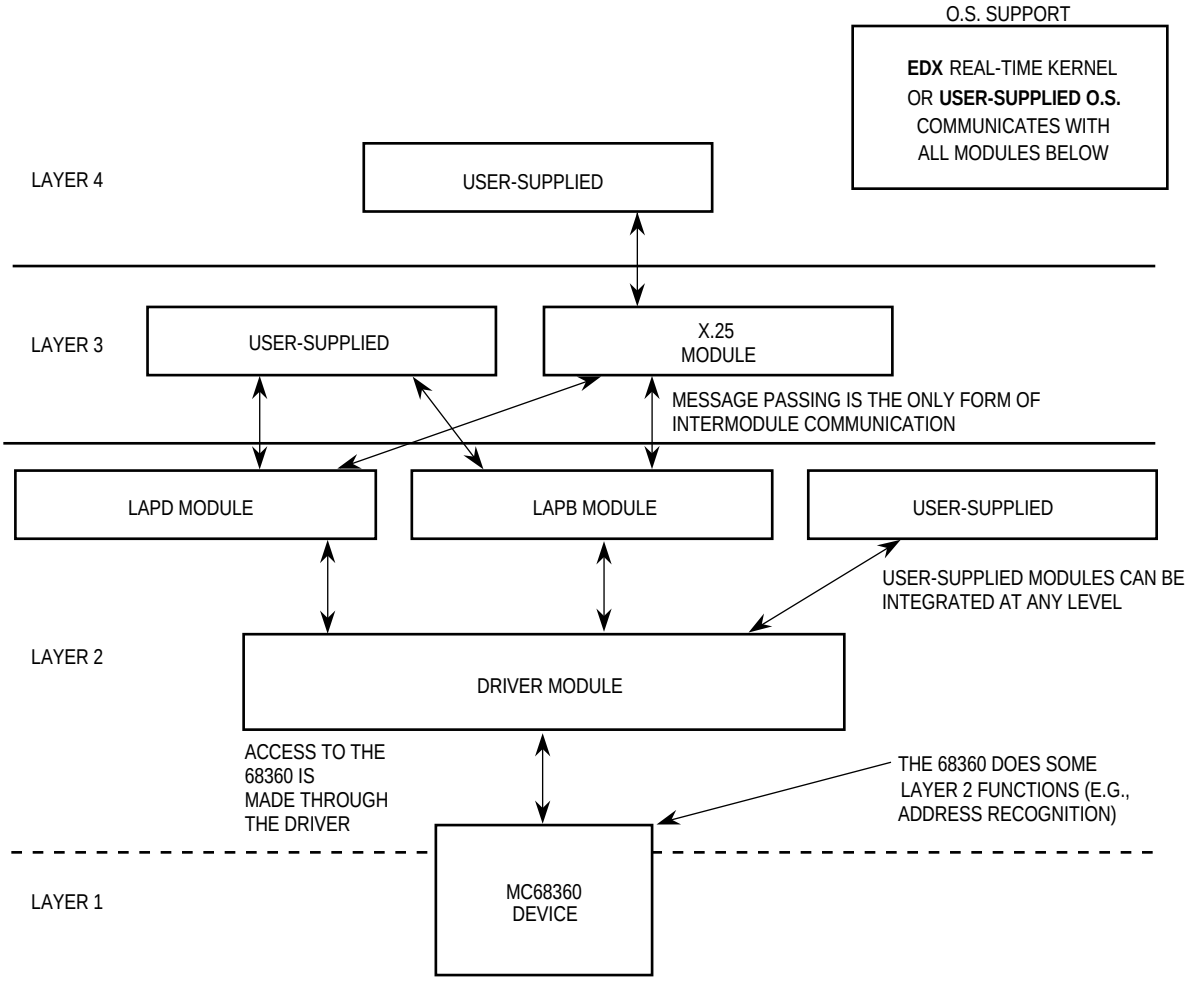


Figure B-1. Software Overview

The chip driver module and simple driver module features are as follows:

- Provides all software modules with an interface to the QUICC
- Illustrates QUICC initialization and interrupt handling
- Provides complete configuration of the QUICC on system start (or restart)
- Provides services for the QUICC serial ports

Freescale Semiconductor, Inc.

APPENDIX D
MC68MH360 PRODUCT BRIEF

The MC68MH360 is a derivative of the MC68360 QUICC.™ It is also known as the QUICC32.™

The QUICC32 has some modifications in the Communication Processor Module (CPM) hardware and a larger internal dual port RAM. These hardware changes in combination with a new ROM based microcode, called QMC (QUICC Multichannel Controller), emulates up to 32 HDLC channels within one SCC. The MH360 is ideal in primary rate ISDN applications and interfaces directly to a E1/T1 line and is capable of terminating all time slots using HDLC or transparent mode of operation. One of the time slot assigners (SI) is dedicated to the use of the QMC. The SI transfers the whole frame or selected time slots to one SCC. All other features remain unchanged and the QUICC32 is pin compatible with the other family members.

A QUICC32 in non-QMC mode has exactly the same functionality as a standard MC68360 with the exception that the Centronics and BISYNC protocols have been removed on the QUICC32 to create ROM space for the QMC protocol.

D.1 QUICC32 KEY FEATURES

The following list summarizes the key MC68MH360 QUICC32 features:

D.1.1 General

- Up to 32 Independent Communication Channels
- Arbitrary Mapping of Any of 0-31 Channels to Any of 0-31 TDM Time Slots
- Can Support Arbitrary Mapping of Any of 0-31 Channels to Any of 0-63 TDM Time Slots in Case of Common Rx & Tx Mapping
- Independent Mapping for Receive/Transmit
- Supports Either QUICC Transparent or QUICC HDLC Protocols for Each Channel
- QUICC-Like Manner for Channel Programming and Parameter Passing
- Up to 64 DMA Channels with Linear Buffer Array
- Interrupt Circular Buffer with Programmable Size and Overflow Identification
- Global Loop Mode
- Individual Channel Loop Mode
- Programmable Frame Length (Via QUICC's SI)

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com