Welcome to E-XFL.COM

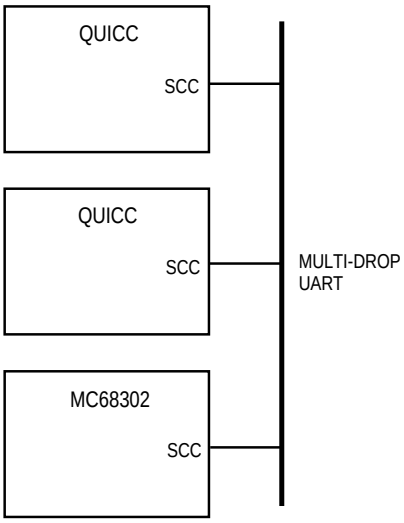
Understanding Embedded - Microprocessors

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

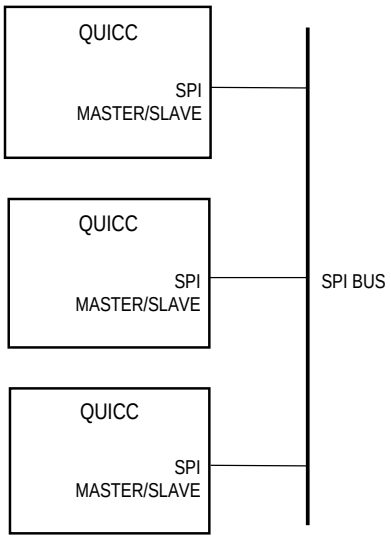
Details	
Product Status	Obsolete
Core Processor	CPU32+
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	25MHz
Co-Processors/DSP	Communications; CPM
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	10Mbps (1)
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	357-BBGA
Supplier Device Package	357-PBGA (25x25)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68360vr25lr2



NOTES:
1. Simple LAN based on UART mode.
2. Ninth bit is an "address" bit.

Figure 1-8. UART LAN Implementation

Figure 1-9 shows how the SPIs on the QUICC can be used to connect devices together into a local bus. The SPI exists on many other Motorola devices, such as the MC68HC11 microcontroller, and a number of peripherals such as A/D and D/A converters, LED drivers, LCD drivers, real-time clocks, serial EEPROM, PLL frequency synthesizers, and shift registers.



NOTE: SPI bus configuration—each QUICC can be the master in turn.

Figure 1-9. SPI Local Bus Implementation

The interrupt acknowledge cycle is a read cycle. It differs from the read cycle described in 4.3.1 Read Cycle in that it accesses the CPU address space. Specifically, the differences are as follows:

1. FC3–FC0 are set to $7 (FC3/FC2/FC1/FC0 = 0111) for CPU address space.
2. A3, A2, and A1 are set to the interrupt request level, and the IACKx strobe corresponding to the current interrupt level is asserted. (Either the function codes and address signals or the IACKx strobes can be monitored to determine that an interrupt acknowledge cycle is in progress and the current interrupt level.)
3. The CPU32+ space type field (A19–A16) is set to $F (interrupt acknowledge).
4. Other address signals (A31–A20, A15–A4, and A0) are set to one.

The responding device places the vector number on the data bus during the interrupt acknowledge cycle. Beyond this, the cycle is terminated normally with DSACKx.

Figure 4-26 is a flowchart of the interrupt acknowledge cycle; Figure 4-27 shows the timing for an interrupt acknowledge cycle terminated with DSACKx.

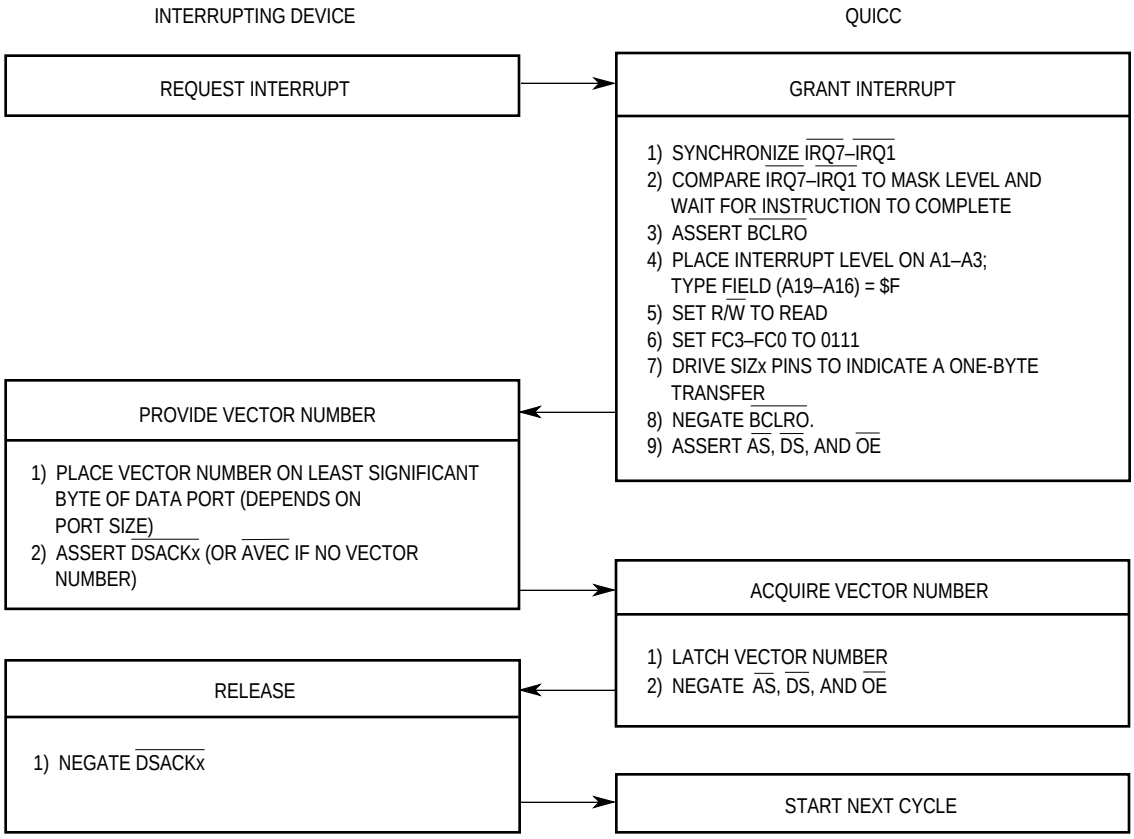


Figure 4-26. Interrupt Acknowledge Cycle Flowchart

without generating spikes on the CLKO1 pin. If the CLKO1 pin is not connected to external circuits, set both bits (disabling the clock output) to minimize noise and power dissipation.

The COM1 bits are cleared at system reset, unless MODCK = 01, in which case they are ones. This prevents CLKO1 and CLKO2 from both defaulting to an active state after reset, for all four combinations of the MODCK1-0 pins. This reduces the potential for system noise at reset. CLKO1 may be enabled later, if desired.

00 = Clock Out Enabled, Full-Strength Output Buffer
01 = Clock Out Enabled, 2/3-Strength Output Buffer
10 = Clock Out Enabled, 1/3-Strength Output Buffer
11 = Clock Out Disabled (Driving 1).

NOTE

If a continuous clock source is needed by the user when MODCK = 01, then the user should use the output of the external oscillator instead of the CLKO1 pin.

The sum of strength of CLKO1 and CLKO2 should not exceed 1. (If COM2 is set to 2/3 drive configuration, then COM1 cannot be greater than 1/3 drive configuration)

When MODCK is set to 01, CLOCKO1 is disabled at reset until the COM1 bit is changed.

The CLKO1 logic is as follows:

when COM1 bits in the CLKOCR = 11, CLKO1 is driven high;

when COM1 bits in the CLKOCR ≠11, CLKO1 is driven according to the following conditions:

a. Driven low if the PLL is NOT locked AND $\overline{\text{RESETH}}$ is asserted.

b. Driven with the same frequency as EXTAL clock if the PLL is locked.

c. Driven low if the PLL unlocked due to MF change.

6.9.3.10 PLL CONTROL REGISTER (PLLCR). The PLLCR controls the operation of the PLL. It can be read or written only in supervisor mode. Writing into this register is allowed only if the PLLWP bit is zero. The reset state of PLLCR produces an operating frequency of 13.14 MHz when the PLL is referenced to a 32.768-kHz crystal or to 4.192 MHz. Two pins (MODCK1–MODCK0) are sampled during hardware reset (see Table 6-1).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLLEN	PLLWP	PREEN	STSIM	MF11	MF10	MF9	MF8	MF7	MF6	MF5	MF4	MF3	MF2	MF1	MF0
RESET:															
1*	0	0	0	0	0	0	MODCK1	MODCK1	0	0	MODCK1	0	0	0	0

Note:
The default value is one unless MODCK1-MODCK0 pins are tdriven with 00 during reset.

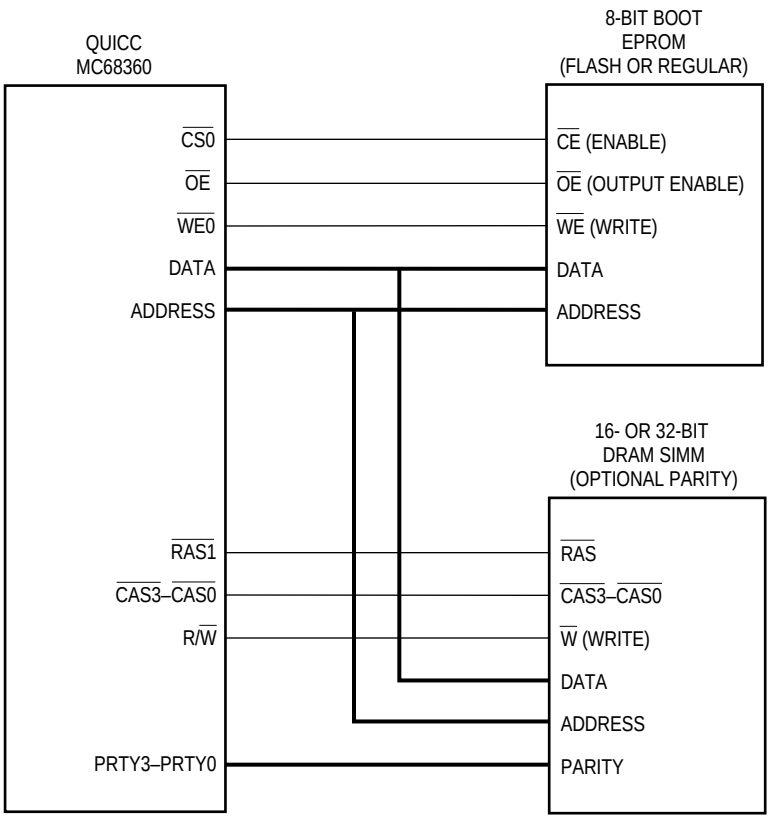


Figure 6-11. Minimum QUICC System Configuration

If a larger system is required, the only additional glue logic that may be needed is external buffers (see Figure 6-12). In this case, a boot EPROM and a flash EPROM are supported. Also, two DRAM SIMMs are supported using RAS1 and RAS2.

Each of the eight memory banks may be used by an external master such as an MC68EC040, MC68030, or even another QUICC. Whenever an external master accesses DRAM, SRAM, or a peripheral within one of the regions of the memory banks, the memory controller will control the access for that external master.

If DRAM is accessed by an external master, an external multiplexer must be provided. In that case, the QUICC AMUX signal can be used to control the multiplexing. The DRAM controller supports use by an MC68EC040 and another QUICC or MC68030-type device. In such a case, the MC68EC040 and QUICC/MC68030-type device can access the DRAM in different modes and at different rates. For instance, the MC68EC040 can access the DRAM using two-clock bursts, while an external QUICC accesses the DRAM using page mode with three-clock page hits, four-clock page normal, and five-clock page miss accesses. Thus, the MC68EC040 access to DRAM is not slowed by the presence of other slower masters on the system bus. In addition, the MC68EC040 is not slowed by the performance of the DRAM accesses by the QUICC's internal bus masters (CPU32+, IDMAs, SDMAs, etc.) All accesses may occur at different rates, with the MC68EC040 parameters being programmed independently and the external QUICC/MC68030-type master being up to one wait state

controller may sample $\overline{TS}$ with a one-clock-phase delay. This will delay the assertion of the $\overline{CS}$ or $\overline{RAS}$ in the MC68EC040 memory cycle by one clock phase. It will delay the rest of the bus cycle by one clock (effectively adding one extra clock cycle per bus cycle).

NOTE

In general, the user determines whether this bit must be set before to selecting the WBT40 and TCYC bits.

0 = Do not sample $\overline{TS}$.
1 = Sample $\overline{TS}$ prior to using it.

NCS—No CPU Space

This attribute specifies whether the $\overline{CS}$/$\overline{RAS}$ signal will assert on a CPU space access cycle. If both supervisor data and program accesses are desired, while ignoring CPU space accesses, then this bit should be set. (Note that an interrupt acknowledge cycle is a CPU space access, but a user or supervisor read/write cycle is not.) A CPU space access has the function code value 0111.

0 = Assert $\overline{CS}$/$\overline{RAS}$ on CPU space accesses (default).
1 = Suppress $\overline{CS}$/$\overline{RAS}$ on CPU space accesses.

NOTE

In default state, user should program the FC3-FC0 in both the Option Registers and Base Registers so that CS/RAS will not get asserted in an undesirable address range.

GAMX—Global Address Mux Enable

This attribute determines whether the QUICC will provide internal address multiplexing for DRAM banks. If not, the address multiplexing must be provided externally, with the QUICC's AMUX pin being used to control the multiplexers. AMUX is high to signify the row, low to signify the column address, and then negated (high) at the end of the DRAM bus cycle.

There are two situations in which the user may wish to provide address multiplexing externally. First, external multiplexers are required when an external master exists in the system and that external master needs to access the DRAM. Second, using external address multiplexing causes the clock to address valid timing as slightly accelerated, which may be beneficial in certain high-performance situations.

0 = Disable internal address multiplexing for all DRAM banks.
1 = Enable internal address multiplexing for all DRAM banks.

Bits 4–0—Reserved

6.13.2 Memory Controller Status Register (MSTAT)

The MSTAT register reports memory controller error information to the user. These bits are set, regardless of whether an internal or external master originated the cycle. Bits are reset by writing a one to that bit; writing a zero has no effect. The register may be read at any time and is cleared by reset. No interrupts are generated from this register; however, an internal

NOTE

CSNT40 is ignored for an SRAM cycle by an external master if the SYNC bit is cleared. CSNT40 = 1 is not valid for external DSACK assertion

BACK40—Burst Acknowledge MC68EC040

This bit is used to acknowledge a burst cycle to the MC68040. If set, bursts are enabled in this bank. The QUICC generates address lines 2,3 on the BADDR3–BADDR2 pins.

0 = Do not acknowledge burst.
1 = Acknowledge burst; MC68040 bursts are handled by the memory controller for this bank.

TRLXQ—Timing Relax

This bit delays the beginning of the internal QUICC or external QUICC/MC68030-type bus master cycle to relax the timing constraints on the user. This attribute is useful for slow peripherals that require additional address setup time. Chip selects are delayed by one phase, and the cycle is delayed by one clock. For accesses to DRAM, $\overline{RAS}$ is delayed by one phase, and $\overline{CAS}$ and AMUX are delayed by two phases, giving a total cycle increase of one clock. See Figures 6-16 and 6-17 for timing diagrams of different cases.

0 = Do not relax timing.
1 = Relax timing at the beginning of the cycle. One additional clock cycle is added when this bit is set.

NOTE

TRLXQ is ignored for an SRAM cycle by an external master if the SYNC bit is cleared.

To relax the MC68EC040 cycles, use the TSS40 bit in the GMR.

User should avoid setting both TRLXQ and CSNTQ = 1, when TCYC = 0. This bit combination will result in a bus cycle without CS assertion.

FC3–FC0—Function Code

This field can be used to specify that accesses with the memory bank be limited to a certain address space type. These bits are used in conjunction with the FCM3–FCM0 bits in the OR.

BA31–BA11—Base Address

The base address field, the upper 21 bits of each BR, and the function code field are compared to the address on the address bus to determine if a DRAM/SRAM region is being accessed by an internal QUICC master.

If the SRAM/DRAM region is being accessed by an external master and the $\overline{WE}$ lines are not used, then A31–A28 address lines and the BA31–BA28 bits are also used in the comparison. If, however, the SRAM/DRAM region is being accessed by an external master

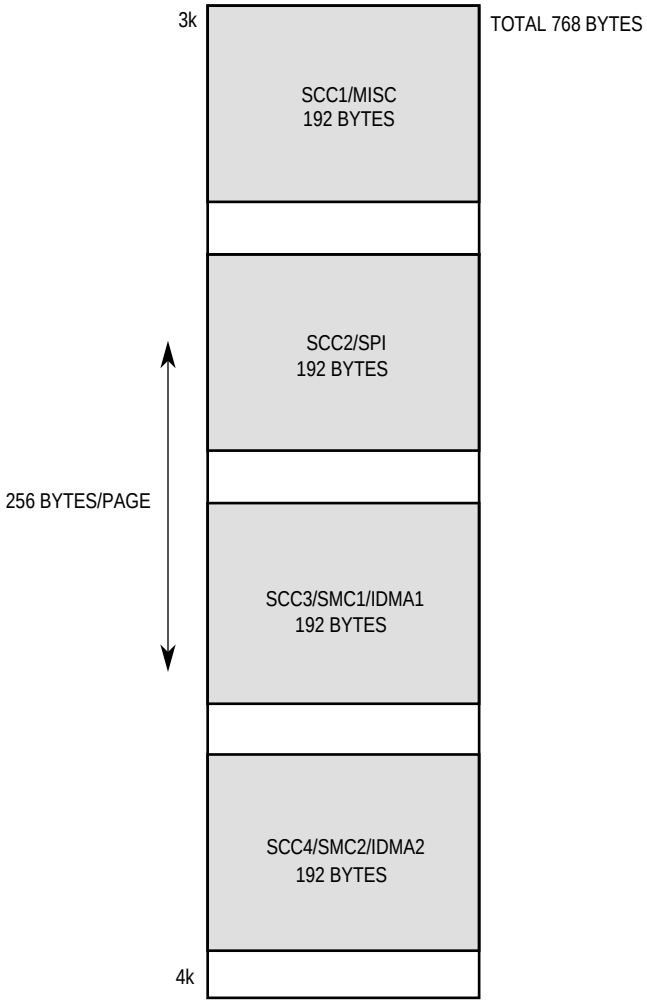


Figure 7-4. Parameter RAM Overview

7.4 RISC TIMER TABLES

The RISC controller has the ability to control up to 16 timers. These timers are separate from the four general-purpose timers and baud rate generators in the CPM. The 16 timers are ideally used in protocols that do not require extreme precision, but in which it is desirable to off-load the host CPU from having to scan the timer tables that are created in software. These timers are clocked from an internal timer used only by the RISC.

The features of the RISC timer tables are as follows:

- Up to 16 Timers Supported
- Two Timer Modes: One-Shot and Restart
- Maskable Interrupt on Timer Expiration
- Programmable Timer Resolution As Low As 41 µs at 25 MHz
- Maximum Timeout Period of 172 Sec at 25 MHz

FRZ1–FRZ0—Freeze

These bits determine the action to be taken when the FREEZE signal is asserted. The IDMA negates its internal bus request and keeps it negated until FREEZE is negated or the IDMA is reset.

00 = The IDMA channels ignore the FREEZE signal.
01 = Reserved.
10 = The IDMA channels freeze on the next bus cycle.
11 = Reserved.

ARBP—Arbitration Priority

These two bits select the arbitration priority between the two IDMA channels.

00 = IDMA channel 1 has priority over channel 2.
01 = IDMA channel 2 has priority over channel 1.
10 = Rotating priority.
11 = Reserved.

ISM—Interrupt Service Mask

These bits contain the interrupt service mask. When the interrupt service level on the IMB is greater than the interrupt service mask, the IDMA vacates the bus and negates its bus request to the IMB until the interrupt level service is less than or equal to the interrupt service mask.

NOTE

The user should program ISM to 7 for typical user applications. This gives the IDMA priority over all interrupt handlers. These bits MUST be set to 7 if the QUICC is in slave mode.

Bits 7, 3–0—Reserved

IAID—IDMA Arbitration ID

These bits establish bus arbitration priority level among sub-blocks that have the capability of becoming bus master. In the QUICC, the IDMAs, the SDMAs, and the SIM60 DRAM refresh controller can become bus masters. An arbitration ID uses a number (0–7) to decide the priority of multiple bus masters that are requesting the IMB. A 0 is the lowest priority and a 7 is the highest priority.

The value programmed into the IAID bits is the arbitration ID of the highest priority IDMA channel. The arbitration ID of the lowest priority IDMA channel is IAID minus 2. The ARBP bits determine which IDMA channel has the higher priority. If round-robin priority is selected, then the IDMA channels alternate between the two IAID values.

Example: If ARBP = 00, selecting IDMA channel 1 to always have the highest priority, the IAID values are:

IDMA channel 1 arbitration ID = IAID

IDMA channel 2 arbitration ID = IAID – 2

NOTES

The user should program IAID to 2 in typical user applications. IAID should not be programmed to a value less than 2. This val-

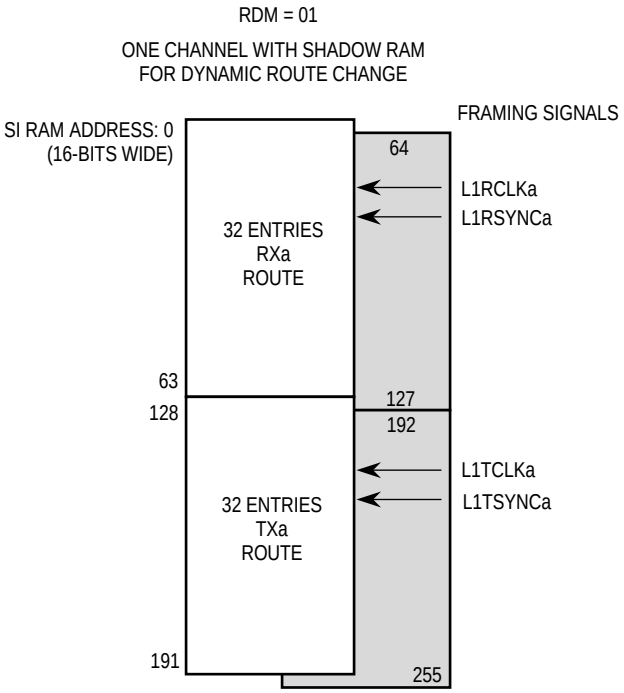


Figure 7-24. SI RAM: One TDM with Dynamic Frames

7.8.4.3 TWO MULTIPLEXED CHANNELS WITH STATIC FRAMES. With this configuration (see Figure 7-25), there are 32 entries for transmit data and strobe routing and 32 entries for receive data and strobe routing. This configuration should be chosen when two TDMs are required and the routing on that TDM does not need to be changed dynamically.

TCI—Transmit Clock Invert

0 = Normal operation.

1 = The internal transmit clock (TCLK) is inverted by the SCC before it is used. This option allows the SCC to clock data out one-half clock earlier, on the rising edge of TCLK rather than the falling edge. In this mode, the SCC offers a minimum and maximum "rising clock edge to data" specification. Data output by the SCC after the rising edge of an external transmit clock can be latched by the external receiver one clock cycle later on the next rising edge of the same transmit clock. This option is recommended for Ethernet, HDLC, or transparent operation when the clock rates are high (e.g., above 8 MHz) to improve data setup time for the external receiver.

TSNC—Transmit Sense

This bit indicates the amount of time the internal sense signal will stay active after the last transition on the RXD pin, indicating that the line is free. For instance, these bits can be used in the AppleTalk protocol to avoid the spurious $\overline{CS}$-changed interrupt that would otherwise occur during the frame sync sequence that precedes the opening flags.

If RDCR is configured to 1× mode, the delay is the greater of the two numbers listed. If RDCR is configured to 8×, 16×, or 32× mode, the delay is the lesser of the two numbers listed.

00 = Infinite—carrier sense is always active (default)

01 = 14 or 6.5 bit times as determined by the RDCR bits

10 = 4 or 1.5 bit times as determined by the RDCR bits (normally chosen for AppleTalk)

11 = 3 or 1 bit times as determined by the RDCR bits

RINV—DPLL Receive Input Invert Data

0 = No invert

1 = Invert the data before it is sent to the on-chip DPLL for reception. This setting is used to produce FM1 from FM0, NRZI space from NRZI mark, etc. It may also be used in regular NRZ mode to invert the data stream.

NOTE

This bit must be 0 in HDLC BUS mode.

TINV—DPLL Transmit Input Invert Data

0 = No invert

1 = Invert the data before it is sent to the on-chip DPLL for transmission. This setting is used to produce FM1 from FM0, NRZI space from NRZI mark, etc. It may also be used in regular NRZ mode to invert the data stream.

NOTE

This bit must be 0 in HDLC BUS mode.

In T1 applications, setting TINV and TEND creates a continuously inverted HDLC data stream.

TPL—Tx Preamble Length

The TPL bits determine the length of the preamble configured by the TPP bits.

000 = No preamble (default)
001 = 8 bits (1 byte)
010 = 16 bits (2 bytes)
011 = 32 bits (4 bytes)
100 = 48 bits (6 bytes) (Select this setting for Ethernet operation.)
101 = 64 bits (8 bytes)
110 = 128 bits (16 bytes)
111 = Reserved

TPP—Tx Preamble Pattern

The TPP bits determine what, if any, bit pattern should precede the start of each transmit frame. The preamble pattern will be sent prior to the first flag/sync of the frame. TPP is ignored if the SCC is programmed to UART mode. The length of the preamble is programmed in TPL. The preamble pattern is typically transmitted to a receiving station that uses a DPLL for clock recovery. The receiving DPLL uses the regular pattern of the preamble to help it lock onto the received signal in a short, predictable time period.

00 = All zeros
01 = Repeating 10’s (Select this setting for Ethernet operation.)
10 = Repeating 01’s
11 = All ones (Select this setting for LocalTalk operation.)

Tend—Transmitter Frame Ending

This bit is intended particularly for the NMSI transmitter encoding of the DPLL. Tend determines whether the TXD line should idle in a high state or in an encoded ones state (which may be either high or low). It may, however, be used with other encodings besides NMSI.

0 = Default operation. The TXD line is encoded only when data is transmitted (including the preamble and opening and closing flags/syncs). When no data is available to transmit, the line is driven high.
1 = The TXD line is always encoded (even when idles are transmitted).

TDCR—Transmit Divide Clock Rate

The TDCR bits determine the divider rate of the transmitter. If the DPLL is not used, the 1× value should be chosen, except in asynchronous UART mode where 8×, 16×, or 32× must be chosen. The user should program TDCR to equal RDCR in most applications.

If the DPLL is used in the application, the selection of TDCR depends on the encoding. NRZI usualy requires 1×; whereas, FM0/FM1, Manchester, and Differential Manchester allow 8×, 16×, or 32×. The 8× option allows highest speed; whereas, the 32× option provides the greatest resolution. TDCR is usually equal to RDCR to allow the same clock frequency source to control both the transmitter and receiver.

00 = 1× clock mode (Only NRZ or NRZI encodings are allowed.)
01 = 8× clock mode
10 = 16× clock mode (normally chosen for UART and AppleTalk)
11 = 32× clock mode

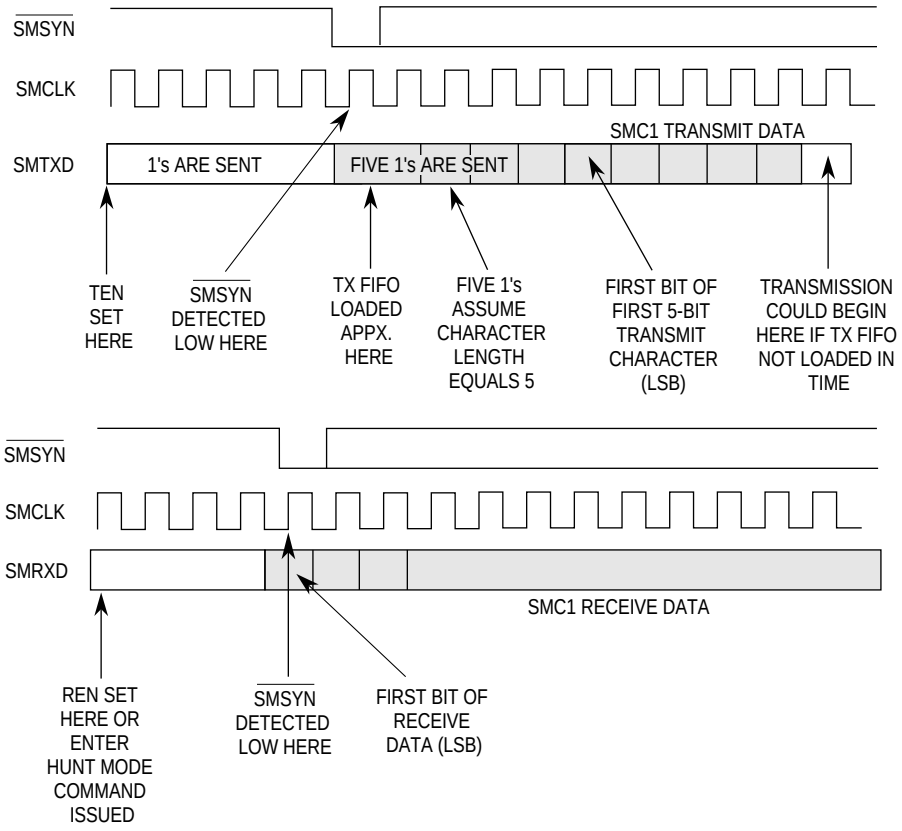
7.10.23.23 SCC ETHERNET EXAMPLE. The following list is an initialization sequence for an Ethernet channel. SCC1 is used. The CLK1 pin is used for the Ethernet receiver, and the CLK2 pin is used for the Ethernet transmitter.

1. The SDCR (SDMA Configuration Register) should be initialized to $0740, rather than being left at its default value of $0000.
2. Configure the port A pins to enable the TXD1 and RXD1 pins. Write PAPAR bits 0 and 1 with ones. Write PADIR bits 0 and 1 with zeros. Write PAODR bit 1 with zero.
3. Configure the port C pins to enable $\overline{\text{CTS1}}$ (CLSN) and $\overline{\text{CD1}}$ (RENA). Write PCPAR bits 4 and 5 with zeros. Write PCDIR bits 4 and 5 with zero. Write PCSO bits 4 and 5 with ones.
4. Do not enable the $\overline{\text{RTS1}}$ (TENA) pin yet because the pin is still functioning as $\overline{\text{RTS}}$ (inactive in the high state), and transmission on the LAN could accidentally begin.
5. Configure port A to enable the CLK1 and CLK2 pins. Write PAPAR bits 8 and 9 with a ones. Write PADIR bits 8 and 9 with zeros.
6. Connect the CLK1 and CLK2 pins to SCC1 using the SI. Write the R1CS bits in SICR to 101. Write the T1CS bits in SICR to 100.
7. Connect the SCC1 to the NMSI (i.e., its own set of pins). Clear the SC1 bit in the SICR.
8. Write RBASE and TBASE in the SCC parameter RAM to point to the Rx BD and Tx BD in the dual-port RAM. Assuming one Rx BD at the beginning of dual-port RAM, and one Tx BD following that Rx BD, write RBASE with $0000 and TBASE with $0008.
9. Program the CR to execute the INIT RX & TX PARAMS command for this channel. For instance, to execute this command for SCC1, write $0001 to the CR. This command causes the RBPTR and TBPTR parameters of the serial channel to be updated with the new values just programmed into RBASE and TBASE.
10. Write RFCR with $18 and TFCR with $18 for normal operation.
11. Write MRBLR with the maximum number of bytes per receive buffer. For this case, assume 1520 bytes, so MRBLR = $05F0. (In this example, the user wants to receive an entire frame into one buffer, so the MRBLR value is simply chosen to be the first value larger than 1518 that is evenly divisible by 4).
12. Write C_PRES with $FFFFFFFF to comply with 32-bit CCITT-CRC.
13. Write C_MASK with $DEBB20E3 to comply with 32-bit CCITT-CRC.
14. Clear CRCEC, ALEC, and DISFC for the sake of clarity.
15. Write PAD with $8888 for the PAD value.
16. Write RET_Lim with $000F.
17. Write MFLR with $05EE to make the maximum frame size 1518 bytes.
18. Write MINFLR with $0040 to make the minimum frame size 64 bytes.
19. Write MAXD1 and MAXD2 with $05EE to make the maximum DMA count 1518 bytes.

NOTE

Regardless of whether the transmitter or receiver uses the $\overline{\text{SMSYNx}}$ signal, the $\overline{\text{SMSYNx}}$ signal must make glitch-free transitions from high to low or low to high. Glitches on $\overline{\text{SMSYNx}}$ may cause errant behavior of the SMC.

Once the REN bit is set in SMCMR, the first rising edge of SMCLK that detects the $\overline{\text{SMSYNx}}$ pin as low causes the SMC receiver to achieve synchronization. Data will begin to be received (latched) on the same rising edge of SMCLK that latched $\overline{\text{SMSYNx}}$. This will be the first bit of data received. The receiver will never lose synchronization again, regardless of the state of $\overline{\text{SMSYNx}}$, until the REN bit is cleared by the user.



NOTES:
1. SMCLK is an internal clock derived from an external CLKPIN or a baud rate generator.
2. This example shows the SMC receiver and transmitter enabled separately. If the REN and TEN bits were set at the same time, a single falling edge of SMSYN would synchronize both.

Figure 7-78. Synchronization with the $\overline{\text{SMSYNx}}$ Pin

Once the TEN bit is set in SMCMR, the first rising edge of SMCLK that detects the $\overline{\text{SMSYNx}}$ pin as low causes the SMC transmitter to achieve synchronization. The SMC transmitter will begin transmitting ones asynchronously from the falling edge of $\overline{\text{SMSYNx}}$. After one character of ones is transmitted, if the transmit FIFO is loaded (i.e., the Tx BD was ready with

12. Write $00000020 to the CIMR to allow the SPI to generate a system interrupt. (The CICR should also be initialized.)
13. Write $0370 to SPMODE to enable normal operation (not loopback), master mode, SPI enabled, 8-bit characters, and the fastest speed possible.
14. Write PBDAT bit 0 with zero to assert the SPI select pin.
15. Set the STR bit in the SPCOM to start the transfer.

NOTE

After 5 bytes have been transmitted, the Tx BD is closed. Additionally, the receive buffer is closed after 5 bytes have been received because the L-bit of the Tx BD was set.

7.12.7 SPI Slave Example

The following list is an initialization sequence for use of the SPI as a slave. It is very similar to the SPI master example except that the $\overline{\text{SPISEL}}$ pin is used, rather than a general-purpose I/O pin.

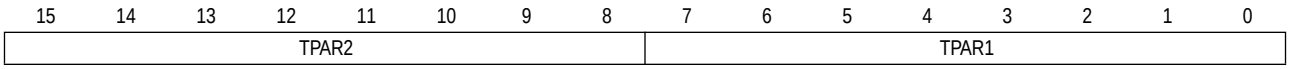
1. The SDCR (SDMA Configuration Register) should be initialized to $0740, rather than being left at its default value of $0000.
2. Configure the port B pins to enable the SPIMOSI, SPIMISO, $\overline{\text{SPISEL}}$, and SPICLK pins. Write PBPAR bits 0, 1, 2, and 3 with ones. Write PBDIR bits 0, 1, 2, and 3 with ones. Write PBODR bits 0, 1, 2, and 3 with zeros.
3. Write RBASE and TBASE in the SPI parameter RAM to point to the Rx BD and Tx BD in the dual-port RAM. Assuming one Rx BD at the beginning of dual-port RAM and one Tx BD following that Rx BD, write RBASE with $0000 and TBASE with $0008.
4. Program the CR to execute the INIT RX & TX PARAMS command for this channel. For instance, to execute this command for SCC1, write $0001 to the CR. This command causes the RBPTR and TBPTR parameters of the serial channel to be updated with the new values just programmed into RBASE and TBASE.
5. Write RFCR with $18 and TFCR with $18 for normal operation.
6. Write MRBLR with the maximum number of bytes per receive buffer. For this case, assume 16 bytes, so MRBLR = $0010.
7. Initialize the Rx BD. Assume the Rx data buffer is at $00001000 in main memory. Write $B000 to Rx_BD_Status. Write $0000 to Rx_BD_Length (not required—done for instructional purposes only). Write $00001000 to Rx_BD_Pointer.
8. Initialize the Tx BD. Assume the Tx data buffer is at $00002000 in main memory and contains five 8-bit characters. Write $B800 to Tx_BD_Status. Write $0005 to Tx_BD_Length. Write $00002000 to Tx_BD_Pointer.
9. Write $FF to the SPIE to clear any previous events.
10. Write $37 to the SPIM to enable all possible SPI interrupts.
11. Write $00000020 to the CIMR to allow the SPI to generate a system interrupt. (The

T/R—Transmit/Receive Select

This bit selects transmitter or receiver operation for the PIP when it is using the interlocked, pulsed, or transparent handshake modes.

0 = Data is input to the PIP.
1 = Data is output from the PIP.

7.13.7.3 PIP TIMING PARAMETERS REGISTER (PTPR). The PTPR is a 16-bit read-write register that is cleared at reset. The PTPR holds two timing parameters, TPAR1 and TPAR2, which are used in the pulsed handshake modes for both a PIP transmitter and a receiver.



TPAR1—Timing Parameter 1

This 8-bit value defines the number of system clocks for TPAR1 in the transmitter or receiver pulsed handshake mode. The value $00 corresponds to 1 QUICC general system clock, and the value $FF corresponds to 256 QUICC general system clocks. A general system clock defaults to 40 ns, assuming a 25-MHz QUICC system.

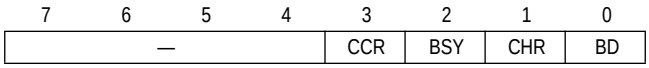
TPAR2—Timing Parameter 2

This 8-bit value defines the number of system clocks for TPAR2 in the transmitter or receiver pulsed handshake mode. The value $00 corresponds to 1 QUICC general system clock, and the value $FF corresponds to 256 QUICC general system clocks. A general system clock defaults to 40 ns, assuming a 25-MHz QUICC system.

7.13.7.4 PIP BUFFER DESCRIPTORS. BDs for the receiver and transmitter that support PIP operation were still in preparation at the time of writing.

7.13.7.5 PIP EVENT REGISTER (PIPE). The PIPE is an 8-bit register used to report events recognized by the PIP and to generate interrupts. It shares the same address as the SMC2 event register; thus, SMC2 cannot be used simultaneously with the PIP. Upon recognition of an event, the PIP sets its corresponding bit in the PIPE. Interrupts generated by this register may be masked in the PIP mask register.

The PIPE 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.



Bits 7–4—Reserved

CCR—Control Character Received

A control character was received (with reject (R) = 1) and stored in the receive control character register.

9.1.1.2 CLOCKING STRATEGY. In this application, the system clock is generated from a 32.768-kHz crystal into the QUICC. The QUICC's internal phase-locked loop (PLL) then multiplies the frequency up to 25 MHz, and outputs 25 MHz on CLKO1 and 50 MHz on CLKO2. Neither CLKO pin is required for the application. It is recommended that the CLKO outputs be disabled in software to save power.

The use of a 32.768-kHz crystal is not a requirement in the application. A 4-MHz crystal or a 25-MHz external oscillator could have been used, if desired.

The QUICC clocking section allows for the clock oscillator to be kept running through the VDDSYN pin in a power-down situation. This section does not address low-power issues, however.

9.1.1.3 RESETTING THE QUICC. If a QUICC is configured to provide the global chip select, it will also provide an internal power-on reset generation. Thus, the reset support function is very simple. If a pushbutton switch is needed, it can be connected by an open-drain buffer to the hard reset ($\overline{\text{RESETH}}$) pin, once debounced. The soft reset ($\overline{\text{RESETS}}$) pin is not used in this design except to indicate that an internal QUICC soft reset is in progress.

9.1.1.4 INTERRUPTS. External interrupts may be brought into the QUICC through either the $\overline{\text{IRQx}}$ pins or parallel I/O pins. This design shows no external interrupts (the $\overline{\text{IRQ7}}$–$\overline{\text{IRQ1}}$ pins are pulled high), but this could be easily changed if desired. Without any external interrupts requiring autovector capability, the $\overline{\text{AVEC}}$ pin is also pulled high.

Internal interrupts from the QUICC may be generated in the SIM60 or the CPM. No additional hardware is required.

9.1.1.5 BUS ARBITRATION. This design assumes that no alternate bus masters exist in the system. Thus, $\overline{\text{BR}}$ is pulled high, and $\overline{\text{BGACK}}$ is not connected, but pulled high since it is an open-drain signal.

9.1.1.6 BREAKPOINT GENERATION. The QUICC can be used to generate a hardware breakpoint signal. The result of a breakpoint (either internally generated using the breakpoint address register or externally generated using the $\overline{\text{BKPT}}$ pin) is a CPU32+ breakpoint cycle. In this application, the $\overline{\text{BKPT}}$ pin is tied high and is not used.

9.1.1.7 BUS MONITOR FUNCTION. The QUICC can be programmed to monitor the bus for bus cycles that are not properly terminated. If $\overline{\text{AS}}$ is asserted but not negated, the cycle will terminate with the $\overline{\text{BERR}}$ pin being asserted.

9.1.1.8 SPURIOUS INTERRUPT MONITOR. The QUICC will watch for spurious interrupt cycles on the levels that it supports internally. If such a condition occurs, $\overline{\text{BERR}}$ will be asserted by the QUICC.

9.1.1.9 SOFTWARE WATCHDOG. If desired, the QUICC software watchdog can be used to generate a level 7 interrupt or a system reset. In this application, the software watchdog is configured in software to generate a reset. No additional hardware is required.

9.4.1.2 CLOCKING STRATEGY. In this application, the system clock is generated from a 25-MHz external oscillator into the QUICC. The MODCKx pins are configured for this. The QUICC internal PLL then multiplies the frequency by 2, and outputs 25 MHz on CLKO1 (not used) and 50 MHz on CLKO2 (with minimal skew between EXTAL, CLKO1, and CLKO2). Minimal skew is very important in this application.

The oscillator is connected to the MC68EC040 BCLK, and the QUICC EXTAL and CLKO2 are connected to the MC68EC040 PCLK. Why not connect the oscillator to the QUICC EXTAL pin and connect the QUICC CLKO1 to the MC68EC040 BCLK? The answer is that the CLKO1 signal does not oscillate while the PLL is locking during power-on reset, and the MC68EC040 needs continuous clocks, even during reset. (CLKO2, however, does oscillate at the EXTAL frequency during power-on reset.) Thus, this solution provides continuous clocks to the MC68EC040 at all times.

An external low-frequency crystal cannot be used in this design because the CLKO1 and CLKO2 pins will stop oscillating while the PLL relocks, which would occur when the software writes to the PLL to multiply the system frequency up to 25 MHz. This would violate the MC68EC040 minimum operating frequency requirement of 16 MHz.

The QUICC clocking section allows for the clock oscillator to be kept running through the VDDSYN pin in a power-down situation. Low-power modes should not be used in this design due to the MC68EC040 requirement of a minimum operating frequency of 16 MHz.

9.4.1.3 RESET STRATEGY. If a QUICC is configured to provide the global chip select, it will also provide an internal power-on reset generation. Thus, the $\overline{\text{RESET}}$ pin of the QUICC just needs to be connected to the $\overline{\text{RSTI}}$ pin on the MC68EC040. The $\overline{\text{RSTO}}$ pin on the MC68EC040 is not used in this design; thus, the MC68040 RESET instruction will have no effect. (It could be added by using an external reset circuit to reset the MC68EC040 and to connect the $\overline{\text{RSTO}}$ to the $\overline{\text{RESET}}$ pin on the QUICC through an open-drain gate.) If a push-button switch is needed, it can be connected by an open-drain buffer to the hard reset ($\overline{\text{RESETH}}$) line, once debounced. The soft reset ($\overline{\text{RESETS}}$) line is not used in this design. It could be used to reset the QUICC without resetting the parallel I/O pins, chip selects, etc.

9.4.1.4 INTERRUPTS. External interrupts may be brought into the QUICC through either the $\overline{\text{IRQx}}$ pins or parallel I/O pins. The QUICC prioritizes these interrupts with its own internally generated interrupts (e.g., timers) to obtain the current highest pending request. In slave mode, the QUICC can output this request to another processor in the system.

NOTE

When the QUICC is in slave mode, the user should *not* use an $\overline{\text{IRQx}}$ pin that is on an interrupt level occupied by either the CPM, periodic interrupt timer, or sofware watchdog.

The request can take the form of a single request pin ($\overline{\text{RQOUT}}$) or three request pins ($\overline{\text{IOUT2}}$–$\overline{\text{IOUT0}}$) that encode the priority of the request. Since the MC68EC040 uses encoded inputs ($\overline{\text{IPL2}}$–$\overline{\text{IPL0}}$), the $\overline{\text{IOUT2}}$–$\overline{\text{IOUT0}}$ pins are chosen.

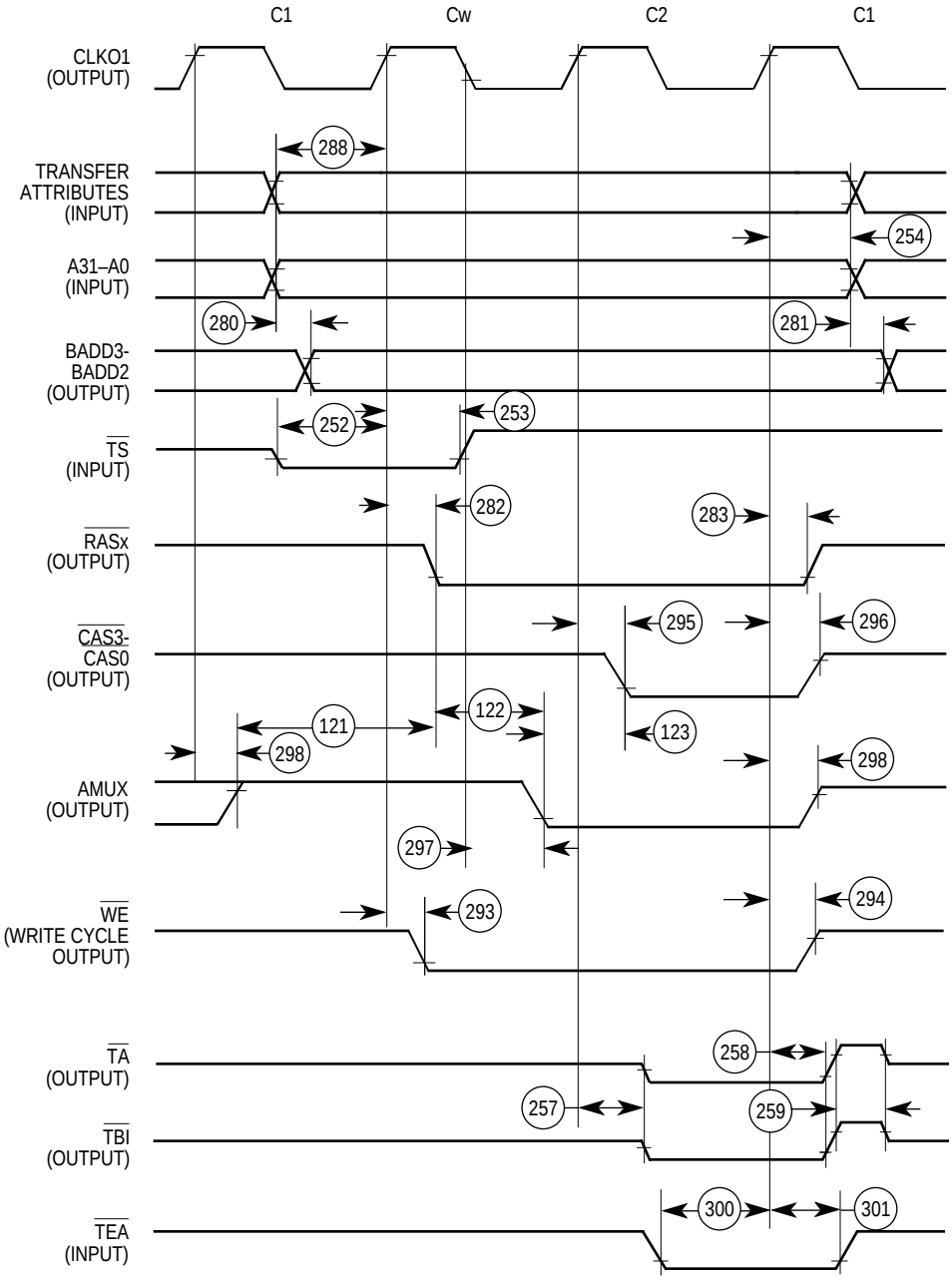


Figure 10-46. External MC68040 DRAM Cycles Timing Diagram