Welcome to E-XFL.COM

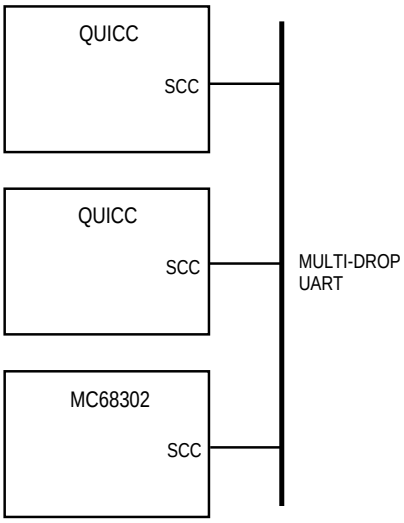
Understanding Embedded - Microprocessors

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

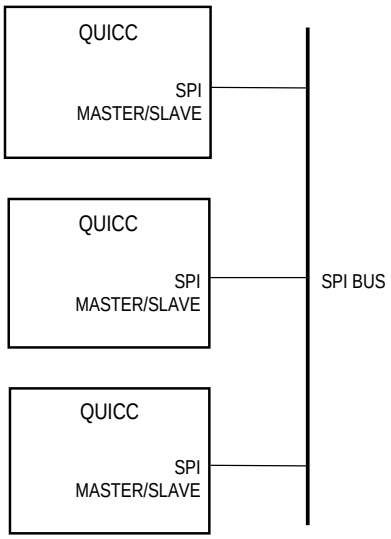
Details	
Product Status	Obsolete
Core Processor	CPU32+
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	33MHz
Co-Processors/DSP	Communications; CPM
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	10Mbps (1)
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	0°C ~ 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/mc68360em33l



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

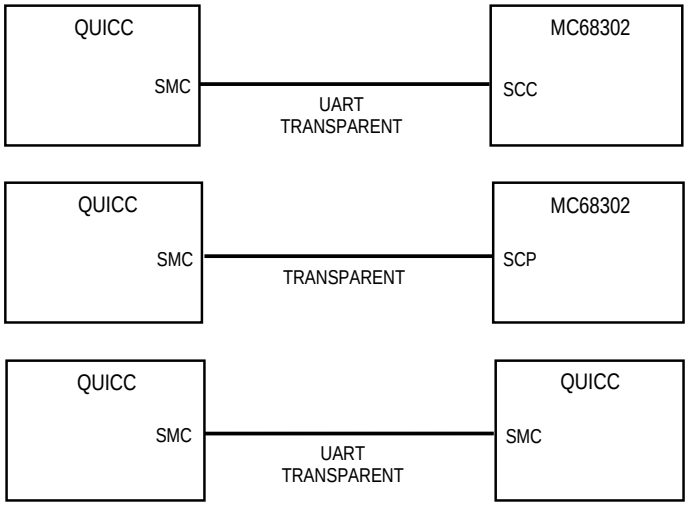
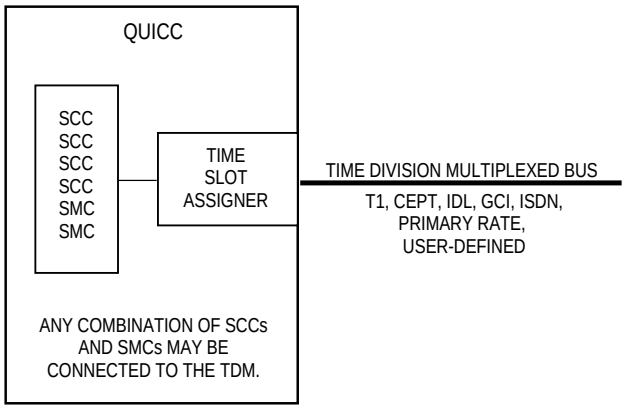


Figure 1-16. Other Point-to-Point Implementations

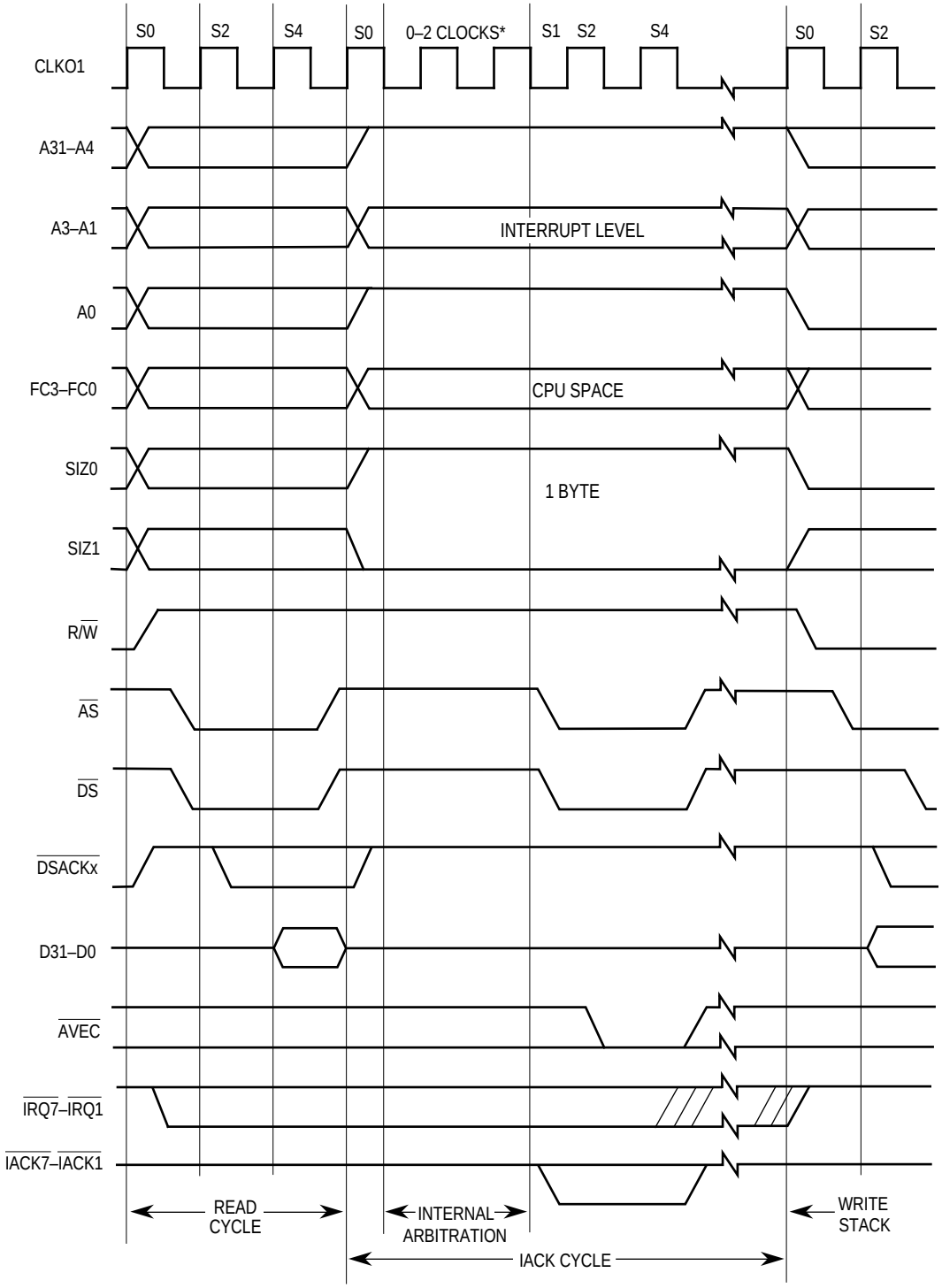
Figure 1-17 shows how up to six of the serial channels can connect to a TDM interface. The QUICC provides a built-in time-slot assigner for access to the TDM time slots. Other channels can work with their own set of pins, allowing possibilities like an Ethernet to T1 bridge, etc.



NOTE: Independent receive and transmit clocking, routing, and syncs are supported.

Figure 1-17. Serial Channel to TDM Bus Implementation

Figure 1-18 shows that the QUICC time-slot assigner can support two TDM buses. Each TDM bus can be of a different format—for example, one TDM can be a T1 line, and one can be a CEPT line. Also this technique could be used to bridge frames from basic rate ISDN to a T1/CEPT line, etc.



* Internal Arbitration may take between 0–2 clock cycles.

Figure 4-28. Autovector Operation Timing

4.4.4.3 SPURIOUS INTERRUPT CYCLE. Requested interrupts, whether internal or external, are arbitrated internally. When no internal module (including the SIM60, which responds for external requests) responds during an interrupt acknowledge cycle by arbitrating for the

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

Result Data:

The “command complete” response ($0FFFF) is returned during the next shift operation.

5.6.2.8.16 Future Commands. Unassigned command opcodes are reserved by Motorola for future expansion. All unused formats within any revision level will perform a NOP and return the ILLEGAL command response.

5.6.3 Deterministic Opcode Tracking

The CPU32+ utilizes deterministic opcode tracking to trace program execution. Two signals, IPIPE and IFETCH, provide all information required to analyze instruction pipeline operation.

5.6.3.1 INSTRUCTION FETCH (IFETCH). IFETCH indicates which bus cycles are accessing data to fill the instruction pipeline. IFETCH is pulse-width modulated to multiplex two indications on a single pin. Asserted for a single clock cycle, IFETCH indicates that the data from the current bus cycle is to be routed to the instruction pipeline. IFETCH held low for two clock cycles indicates that the instruction pipeline has been flushed. The data from the bus cycle is used to begin filling the empty pipeline. Both user and supervisor mode fetches are signaled by IFETCH.

Proper tracking of bus cycles via IFETCH on a fast bus requires a simple state machine. On a two-clock bus, IFETCH may signal a pipeline flush with associated prefetch followed immediately by a second prefetch. That is, IFETCH remains asserted for three clocks, two clocks indicating the flush/fetch and a third clock signaling the second fetch. These two operations are easily discerned if the tracking logic samples IFETCH on the two rising edges of CLKO1, which follow the AS (DS during show cycles) falling edge. Three-clock and slower bus cycles allow time for negation of the signal between consecutive indications and do not experience this operation.

5.6.3.2 INSTRUCTION PIPE (IPIPE1–IPIPE0). The internal instruction pipeline can be modeled as a three-stage FIFO (see Figure 5-28). Stage A is an input buffer—data can be used out of stages B and C. The IPIPE1–IPIPE0 signals indicate the advance of instructions in the pipeline.

The 16-bit instruction register A (IRA) and 16-bit instruction register L (IRL) hold incoming words as they are prefetched. No decoding occurs in IRA or IRL. Instruction register B (IRB) provides initial decoding of the opcode and decoding of extension words; it is a source of immediate data. Instruction register C (IRC) supplies residual opcode decoding during instruction execution.

IRA is of higher priority than IRL. IRL is only loaded from the IMB when a 32-bit instruction fetch is performed. IRA is loaded during every instruction fetch.

IRB is loaded from the contents of IRA or IRL, depending on which one is currently valid. If both IRA and IRL are valid, then IRA is loaded into IRB before IRL is loaded into IRB.

Instruction		Head	Tail	Cycles
BCHG	#, Dn	2	0	6(0/2/0)∗
BCHG	Dn, Dm	4	0	6(0/1/0)
BCHG	#, ⟨FEA⟩	1	2	8(0/2/1)∗
BCHG	Dn, ⟨FEA⟩	2	2	8(0/1/1)
BCLR	#, Dn	2	0	6(0/2/0)∗
BCLR	Dn, Dm	4	0	6(0/1/0)
BCLR	#, ⟨FEA⟩	1	2	8(0/2/1)∗
BCLR	Dn, ⟨FEA⟩	2	2	8(0/1/1)
BSET	#, Dn	2	0	6(0/2/0)∗
BSET	Dn, Dm	4	0	6(0/1/0)
BSET	#, ⟨FEA⟩	1	2	8(0/2/1)∗
BSET	Dn, ⟨FEA⟩	2	2	8(0/1/1)
BTST	#, Dn	2	0	4(0/2/0)∗
BTST	Dn, Dm	2	0	4(0/1/0)
BTST	#, ⟨FEA⟩	1	0	4(0/2/0)∗
BTST	Dn, ⟨FEA⟩	2	0	8(0/1/0)

∗ = An # fetch EA time must be added for this instruction: ⟨FEA⟩ + ⟨FEA⟩ + ⟨OPER⟩
Timing is calculated with the CPU32+ in 16-bit mode.

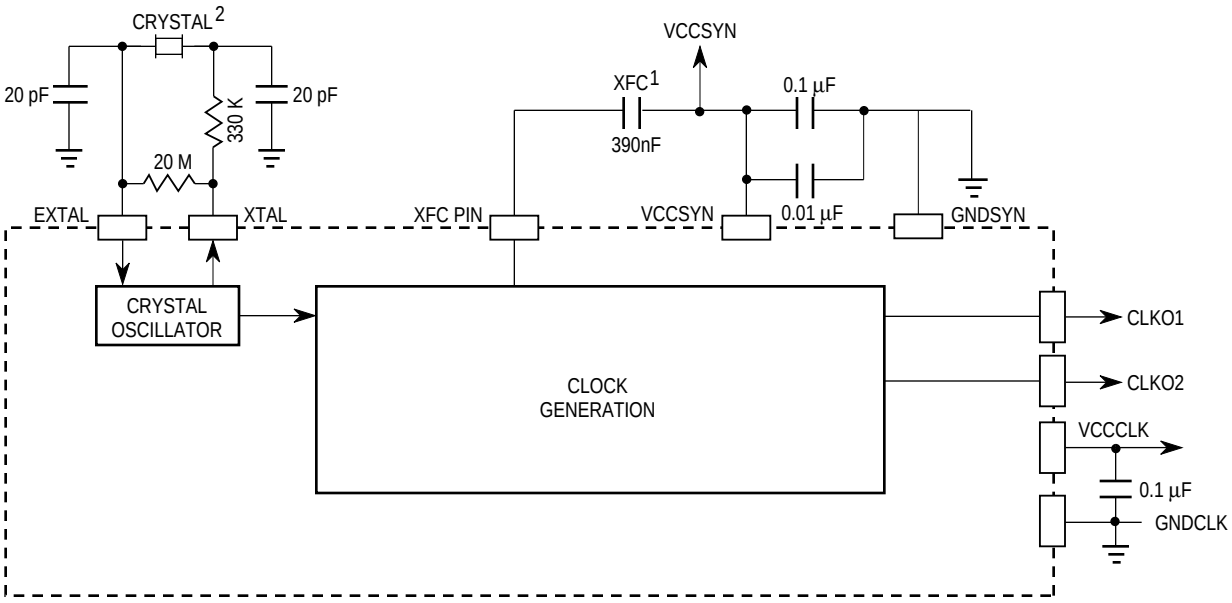
5.7.2.11 CONDITIONAL BRANCH INSTRUCTIONS. The conditional branch instruction timing table indicates the number of clock periods needed for the processor to perform the specified branch on the given branch size, with complete execution times given. No additional tables are needed to calculate total effective execution time for these instructions. The total number of clock cycles is outside the parentheses. The numbers inside parentheses (r/p/w) are included in the total clock cycle number. All timing data assumes two-clock reads and writes.

Instruction		Head	Tail	Cycles
Bcc	(taken)	2	–2	8(0/2/0)
Bcc.B	(not taken)	2	0	4(0/1/0)
Bcc.W	(not taken)	0	0	4(0/2/0)
Bcc.L	(not taken)	0	0	6(0/3/1)
DBcc	(T, not taken)	1	1	4(0/2/0)
DBcc	(F, –1, not taken)	2	0	6(0/2/0)
DBcc	(F, not –1, taken)	6	–2	10(0/2/0)
DBcc	(T, not taken)	4	0	6(0/1/0)∗
DBcc	(F, –1, not taken)	6	0	8(0/1/0)∗
DBcc	(F, not –1, taken)	6	0	10(0/0/0)∗

∗ = In loop mode
Timing is calculated with the CPU32+ in 16-bit mode.

divided prior to being used by any QUICC on-chip module). Furthermore, the divide-by-128 function allows the value of the final system frequency to be chosen with much greater precision, since it is a multiple of ~32 kHz rather than a multiple of ~4 MHz.

The choice of whether to use the divide-by-128 function is made with the MODCK1–MODCK0 pins. This resulting frequency is called CLKIN.



NOTE:
1. Must be low-leakage capacitor. See Section 10 Electrical Characteristics for recommended values.
2. Values are for 32 kHz crystal and may vary due to capacitance on PCB.

Figure 6-6. External Components

6.5.3 Phase-Locked Loop (PLL)

The PLL takes the CLKIN frequency and outputs a high-frequency source used to derive the general system frequency of the QUICC. The PLL is comprised of a phase detector, loop filter, voltage-controlled oscillator (VCO), and multiplication block. The VCO output can be as high as 50 MHz for a 25-MHz QUICC.

The PLL's main functions are frequency multiplication and skew elimination.

6.5.3.1 FREQUENCY MULTIPLICATION. The PLL can multiply the CLKIN input frequency by any integer between 1 and 4096. The output of the VCO is twice the QUICC system frequency after reset.

If a low frequency crystal is chosen (e.g., ~32 kHz), the multiplier defaults to 401, giving a 2× VCO output of ~26 MHz and an initial general system clock of ~13 MHz. The multiplication factor may then be changed to the desired value by writing the MF11–MF0 bits in the PLLCR. When the PLL multiplier is modified in software, the PLL will lose lock, and the clocking to the QUICC will stop until lock is regained (worst case is 2500 clocks; typical case is 500 clocks). See 6.5.4 Low-Power Divider for methods of reducing clock rates without losing lock.

ASTM—Arbitration Synchronous Timing Mode

This bit determines whether the EBI will synchronize the arbitration signals: $\overline{BR}$, $\overline{BG}$, and $\overline{BGACK}$. The synchronization will add a one-clock delay to the external bus arbitration.

0 = Asynchronous timing on the arbitration signals may be used. The arbitration signals will be synchronized internally by the QUICC and do not have to meet any timings relative to the system clock.

1 = Synchronous timing on the arbitration signals must be used. The arbitration control signals will not be synchronized internally and therefore must meet the system clock setup and hold timings.

FRZ1—Freeze SWT and PIT Enable

0 = When FREEZE is asserted, the SWT and the PIT counters continue to run. See 6.3.3 Freeze Support for more information.

1 = When FREEZE is asserted, the SWT and the PIT counters are disabled, preventing interrupts from occurring during software debugging.

FRZ0—Freeze Bus Monitor Enable

0 = When FREEZE is asserted, the bus monitor continues to operate as programmed.

1 = When FREEZE is asserted, both the internal and external bus monitors are disabled.

BCLROID2–BCLROID0—Bus Clear Out Arbitration ID

These bits contain the arbitration priority level for the assertion of the $\overline{BCLRO}$ signal. When internal masters (IDMA, SDMA, or DRAM refresh cycle) request the bus and the arbitration level on the IMB is greater than the bus clear out arbitration ID, the $\overline{BCLRO}$ signal will be asserted until the arbitration level is less than or equal to the bus clear out arbitration ID. $\overline{BCLRO}$ can be used to clear an external master from the external bus when a refresh cycle is pending. It may also be used to clear an external master from the bus when an SDMA or IDMA channel requests the external bus.

NOTE

Program this value to 3 in a normal system to allow the SDMA and DRAM refresh controller to clear other bus masters off the external bus.

SHEN1–SHEN0—Show Cycle Enable

These two control bits determine what the EBI does with the external bus during internal transfer operations (see Table 6-3). A show cycle allows internal transfers to be externally monitored. The address, data, and control signals (except for $\overline{AS}$) are driven externally. $\overline{DS}$ is used to signal address strobe timing for show cycles. Data is valid on the next falling clock edge after $\overline{DS}$ is negated. However, data is not driven externally and $\overline{AS}$ and $\overline{DS}$ are not asserted externally for internal accesses unless show cycles are enabled.

If external bus arbitration is disabled, the EBI will not recognize an external bus request until arbitration is enabled again. When SHEN1 is set, an external bus request causes an internal master to stop its current cycle and relinquish the internal bus. The internal master resumes running cycles on the bus after $\overline{BR}$ and $\overline{BGACK}$ are negated. To prevent bus

7.10.8.2 SCC MASK REGISTER (SCCM). The 16-bit, read-write SCC mask register allows the user to either enable or disable interrupt generation by the CP for specific events in each SCC channel. Note that an interrupt will only be generated if the SCC interrupts in this channel are enabled in the CPM interrupt mask register.

If a bit in the SCC mask register is zero, the CP will not proceed with its usual interrupt handling whenever that event occurs. Anytime a bit in the SCC mask register is set, a one in the corresponding bit in the SCC event register will set the SCC event bit in the CPM interrupt pending register.

The bit position of the SCC mask register is identical to that of the SCC event register.

7.10.8.3 SCC STATUS REGISTER (SCCS). The 8-bit, read-write SCC status register allows the user to monitor real-time status conditions on the RXD line, such as flags, idle, and data carrier sense. It does not show the real-time status of the $\overline{CTS}$ and $\overline{CD}$ pins. Their real-time status is available in the port C parallel I/O.

7.10.9 SCC Initialization

The SCCs require a number of registers and parameters to be configured after a power-on reset. The following outline is the proper sequence for initializing the SCCs, regardless of the protocol used. More detailed examples are given in the protocol sections.

1. Write the parallel I/O ports to configure the I/O pins to connect to the SCCs.
2. The SDCR (SDMA Configuration Register) should be initialized to $0740, rather than being left at its default value of $0000.
3. Write the port C registers to configure the $\overline{CTS}$ and $\overline{CD}$ pins to be parallel I/O with interrupt capability or to be direct connections to the SCC (if modem support is needed).
4. If the TSA is used, the SI must be configured. See 7.8 Serial Interface with Time Slot Assigner for a description of the steps required. If the SCC is used in the NMSI mode, then the SICR must still be initialized.
5. Write GSMR, but do not write the ENT or ENR bits yet.
6. Write the PSMR.
7. Write DSR.
8. Initialize the required values for this SCC in its parameter RAM.
9. Clear out any current events in the SCCE, if desired.
10. Write SCCM to enable the interrupts in the SCCE.
11. 1Write CICR to configure the SCC's interrupt priority.
12. 1Clear out any current interrupts in CIPR, if desired.
13. 1Write CIMR to enable interrupts to the CP interrupt controller.
14. 1Set the ENT and ENR bits in the GSMR.

The BDs may have their ready/empty bits set at any time. Notice that the CR does not need to be accessed following a power-on reset. An SCC should be disabled and reenabled after

GLt—Glitch on Tx

A clock glitch was detected by this SCC on the transmit clock.

AB—Auto Baud

An auto baud lock was detected. The CPU32+ core should rewrite the baud rate generator with the precise divider value for the desired baud rate. See 7.9 Baud Rate Generators (BRGs) for more details.

IDL—Idle Sequence Status Changed

A change in the status of the serial line was detected on the UART channel. The real-time status of the line may be read in SCCS. Idle is entered when one character of all ones is received. It is exited when a single zero is received.

GRA—Graceful Stop Complete

A graceful stop, which was initiated by the GRACEFUL STOP TRANSMIT command, is now complete. This bit is set as soon the transmitter has finished transmitting any buffer that was in progress when the command was issued. It will be set immediately if no buffer was in progress when the command was issued.

BRKe—Break End

The end of a break sequence was detected. This indication will be set no sooner than after one idle bit is received following a break sequence.

BRKs—Break Start

A break character was received. This is the first break of a break sequence. The user will not receive multiple BRKs events if a long break sequence is received.

CCR—Control Character Received

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

BSY—Busy Condition

A character was received and discarded due to lack of buffers. If the multidrop mode is selected, the receiver automatically enters hunt mode immediately. Otherwise, reception continues as soon as an empty buffer is provided. The latest that an Rx BD can be made empty (have its E-bit set) and still guarantee avoiding the busy condition is the middle of the stop bit of the first character to be stored in that buffer.

TX—Tx Buffer

A buffer has been transmitted over the UART channel. If CR = 1 in the Tx BD, this bit is set no sooner than when the last stop bit of the last character in the buffer begins to be transmitted. If CR = 0, this bit is set after the last character was written to the transmit FIFO.

RX—Rx Buffer

A buffer has been received over the UART channel. This event occurs no sooner than the middle of the first stop bit of the character that caused the buffer to be closed.

Refer to Figure 7-73 for the SMC block diagram. The SMC receiver and transmitter are double-buffered, as shown in the block diagram. This corresponds to an effective FIFO size (latency) of two characters.

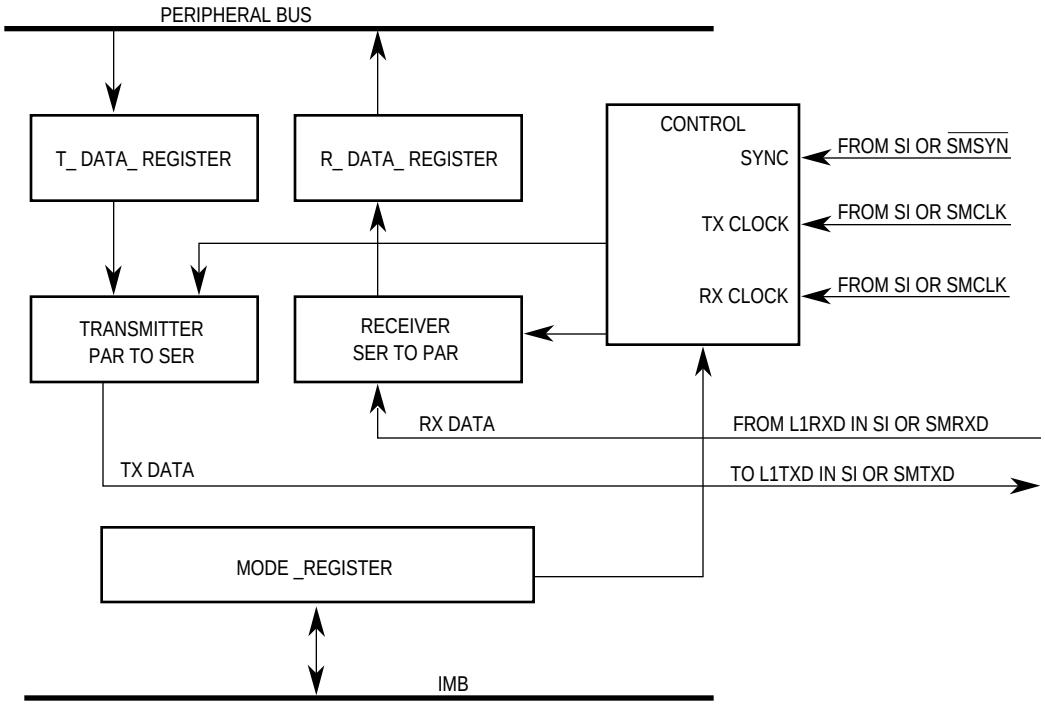


Figure 7-73. SMC Block Diagram

The receive data source for an SMC can be either the L1RXD pin if the SMC is connected to a TDM channel of the SI, or the SMRXD pin if the SMC is connected to the NMSI. The transmit data source can either be the L1TXD pin if the SMC is connected to a TDM, or the SMTXD pin if the SMC is connected to the NMSI.

If the SMC is connected to a TDM, the SMC receive clock and SMC transmit clock can be independent from each other as defined in the SI description. However, if the SMC is connected to the NMSI, the SMC receive clock and SMC transmit clock must be connected to a single clock source called SMCLK. SMCLK is an internal signal name for a clock that is generated from the bank of clocks defined in the SI description. SMCLK may originate from an external pin or one of the four internal baud rate generators. See 7.8.9 NMSI Configuration for more details.

If the SMC is connected to a TDM, it derives its synchronization pulse from the TSA as defined in the SI description. Otherwise, if the SMC is connected to the NMSI and the totally transparent protocol is selected, the SMC may use the $\overline{\text{SMSYN}}$ pin as a synchronization pin to determine when transmission and reception should begin. (The $\overline{\text{SMSYN}}$ pin is not used in the SMC UART mode.)

BD pointed to by the RBPTR if the Rx BD E-bit is set.

7.11.5.4 SMC RECEIVER SHORTCUT SEQUENCE. A shorter sequence is possible if the user desires to reinitialize the receive parameters to the state they had after reset. This sequence is as follows:

1. Clear the REN bit in the SMCMR.
2. INIT RX PARAMETERS command. Any additional modifications may now be made.
3. Set the REN bit in the SMCMR.

7.11.5.5 SWITCHING PROTOCOLS. Sometimes the user desires to switch the protocol that the SMC is executing (for instance, UART to Transparent) without resetting the board or affecting any other SMC. This can be accomplished using only one command and a short number of steps:

1. Clear the TEN and REN bits in the SMCMR.
2. INIT TX AND RX PARAMETERS command. This one command initializes both transmit and receive parameters. Any additional modifications may now be made in the SMCMR to change the protocol, etc.
3. Set the SMCMR TEN and REN bits. The SMC is enabled with the new protocol.

7.11.6 Saving Power

When the TEN and REN bits of an SMC are cleared, that SMC consumes a minimal amount of power.

7.11.7 SMC as a UART

The following paragraphs describe the use of the SMC as a UART.

7.11.7.1 SMC UART KEY FEATURES. The SMC UART contains the following key features:

- Flexible Message-Oriented Data Structure
- Programmable Data Length (5–14 Bits)
- Programmable 1 or 2 Stop Bits
- Even/Odd/No Parity Generation and Checking
- Frame Error, Break, and IDLE Detection
- Transmit Preamble and Break Sequences
- Received Break Character Length Indication
- Continuous Receive and Transmit Modes

7.11.7.2 SMC UART COMPARISON. As compared to the UART modes supported by the SCCs, the SMCs generally offer less functionality and performance. This fits with their purpose of providing simple debug/monitor ports rather than full-featured UARTs. The SMC UARTs do not support the following features:

- $\overline{\text{RTS}}$, $\overline{\text{CTS}}$, and $\overline{\text{CD}}$ Pins
- Receive and Transmit Sections Being Clocked at Different Rates

written to the break count register before this command is given to the SMC UART controller.

RESTART TRANSMIT Command. The RESTART TRANSMIT command enables the transmission of characters on the transmit channel. This command is expected by the SMC UART controller after disabling the channel in its SMC mode register and after the STOP TRANSMIT command. The SMC UART controller will resume transmission from the current TBPTR in the channel's Tx BD table.

INIT TX PARAMETERS Command. This command initializes all the transmit parameters in this serial channel's parameter RAM to their reset state. This command should only be issued when the transmitter is disabled. Note that the INIT TX AND RX PARAMETERS command may also be used to reset both transmit and receive parameters.

7.11.7.7.2 Receive Commands. The following paragraphs describe the UART receive commands.

ENTER HUNT MODE Command. This command should not be used for an SMC UART channel. The CLOSE RX BD command may be used instead.

CLOSE RX BD Command. The CLOSE RX BD command is used to force the SMC to close the current receive BD if it is currently being used, and to use the next BD in the list for any subsequent data that is received. If the SMC is not in the process of receiving data, no action is taken by this command.

iNIT RX PARAMETERS Command. This command Initializes all the receive parameters in this serial channel's parameter RAM to their reset state. This command should only be issued when the receiver is disabled. Note that the INIT TX AND RX PARAMETERS command may also be used to reset both receive and transmit parameters.

7.11.7.8 SEND BREAK (TRANSMITTER). A break is an all-zeros character without stop bits. A break is sent by issuing the STOP TRANSMIT command. The SMC UART completes transmission of any outstanding data and then sends a character with consecutive zeros (the number of zero bits in this character is the sum of the character length, plus the number of start, parity, and stop bits). The SMC UART transmits a programmable number of break characters according to the break count register and then reverts to idle or sends data if the RESTART TRANSMIT command was given before completion. At the completion of the break, the transmitter sends at least one character of idle before transmitting any data to guarantee recognition of a valid start bit.

7.11.7.9 SENDING A PREAMBLE (TRANSMITTER). A preamble sequence gives the programmer a convenient way of ensuring that the line goes idle before starting a new message. The preamble sequence length is constructed of consecutive ones of one character length. If the preamble bit in a BD is set, the SMC will send a preamble sequence before transmitting that data buffer. Example: for 8 data bits, no parity, 1 stop bit, and 1 start bit, a preamble of 10 ones would be sent before the first character in the buffer.

If no preamble sequence is sent, data from two ready transmit buffers may be transmitted without any delay occurring on the transmit pin between the two transmit buffers.

10.5 DC Electrical Specifications

(GND = 0 Vdc, TA = 0 to 70°C; The electrical specifications in this document are preliminary; see numbered notes)

Characteristic	Symbol	Min	Max	Unit
Input High Voltage (except EXTAL)	V_{IH}	2.0	V_{CC}	V
Input Low Voltage	V_{IL}	GND	0.8	V
Input Low Voltage (3.3V Part Only; PA8-15, PB1, PC5, PC7 TCK)	V_{IL}	GND	0.5	V
Input Low Voltage (3.3V Part Only; All Other Pins)	V_{IL}	GND	0.8	V
EXTAL Input High Voltage	V_{IHC}	0.8*(V_{CC})	V_{CC}+0.3	V
Undershoot	—	—	–0.8	V
Input Leakage Current (All Input Only Pins except for TMS, TDI and TRST) V_{in} = V_{CC} or GND	I_{in}	–2.5	2.5	µA
Hi-Z (Off-State) Leakage Current (All Noncrystal Outputs and I/O Pins) V_{in} = 0.5/2.4 V	I_{OZ}	–20	20	µA
Signal Low Input Current V_{IL} = 0.8 V (TMS, TDI and TRST Pins Only) Signal High Input CurrentV_{IH} = 2.0 V(TMS, TDI and TRST Pins Only)	I_L IH	–0.5 –0.5	0.5 0.5	mA mA
Output High Voltage I_{OH} = –0.8 mA, V_{CC} = 4.75 V All Noncrystal Outputs Except Open Drain Pins	V_{OH}	2.4	—	V
Output Low Voltage (For 5 Volt Part) I_{OL} = 2.0 mACLKO1–2 ,FREEZE, IPIPE0–1, IFETCH, BKPTO I_{OL} = 3.2 mAA31–A0, D31–D0, FC3–0, SIZ0–1, PA0, 2, 4, 6, 8–15, PB0–5, PB8–17, PC0–11, TDO, PERR, PRTY0–3, IOUT0–2, AVECO, AS, CAS3–0, BLCRO, RAS0–7 I_{OL} = 5.3 mA, DSACK0–1, R/W, DS, OE, RMC, BG, BGACK, BERR I_{OL} = 7 mATXD1–4 I_{OL} =8.9 mAPB6, PB7, HALT, RESET, BR (Output)	V_{OL}	—	0.5 0.5 0.5 0.5 0.5	V
Output Low Voltage (For 3.3 volt part) I_{OL} = 2.0 mAA31–A0, FC3–0, SIZ0–1, D31–D0, PRTY0–3,CLKO1–2 ,FREEZE, IPIPE0–1, IFETCH, BKPTO I_{OL} = 3.2 mA PA0, 2, 4, 6, 8–15, PB0–5, PB8–17, PC0–11, TDO, PERR, PRTY0–3, IOUT0–2, AVECO, AS, CAS3–0, BLCRO, RAS0–7 I_{OL} = 5.3 mA, DSACK0–1, R/W, DS, OE, RMC, BG, BGACK, BERR I_{OL} = 7 mATXD1–4 I_{OL} =8.9 mAPB6, PB7, HALT, RESET, BR (Output)	V_{OL}	—	0.5 0.5 0.5 0.5 0.5	V
Input Capacitance All I/O Pins	C_{in}	—	20	pF
Load Capacitance (except CLKO1–2)	C_L	—	100	pF
Load Capacitance (CLKO1-2)	C_Lc		50	pF
Power (For 5 Volt Version)	Vcc	4.75	5.25	V
Power (For 3.3 Volt Version)	Vcc	3.0	3.6	V
Minimum Vcc Ramp up time on power on reset	^{t}Ranp	4	—	mSec

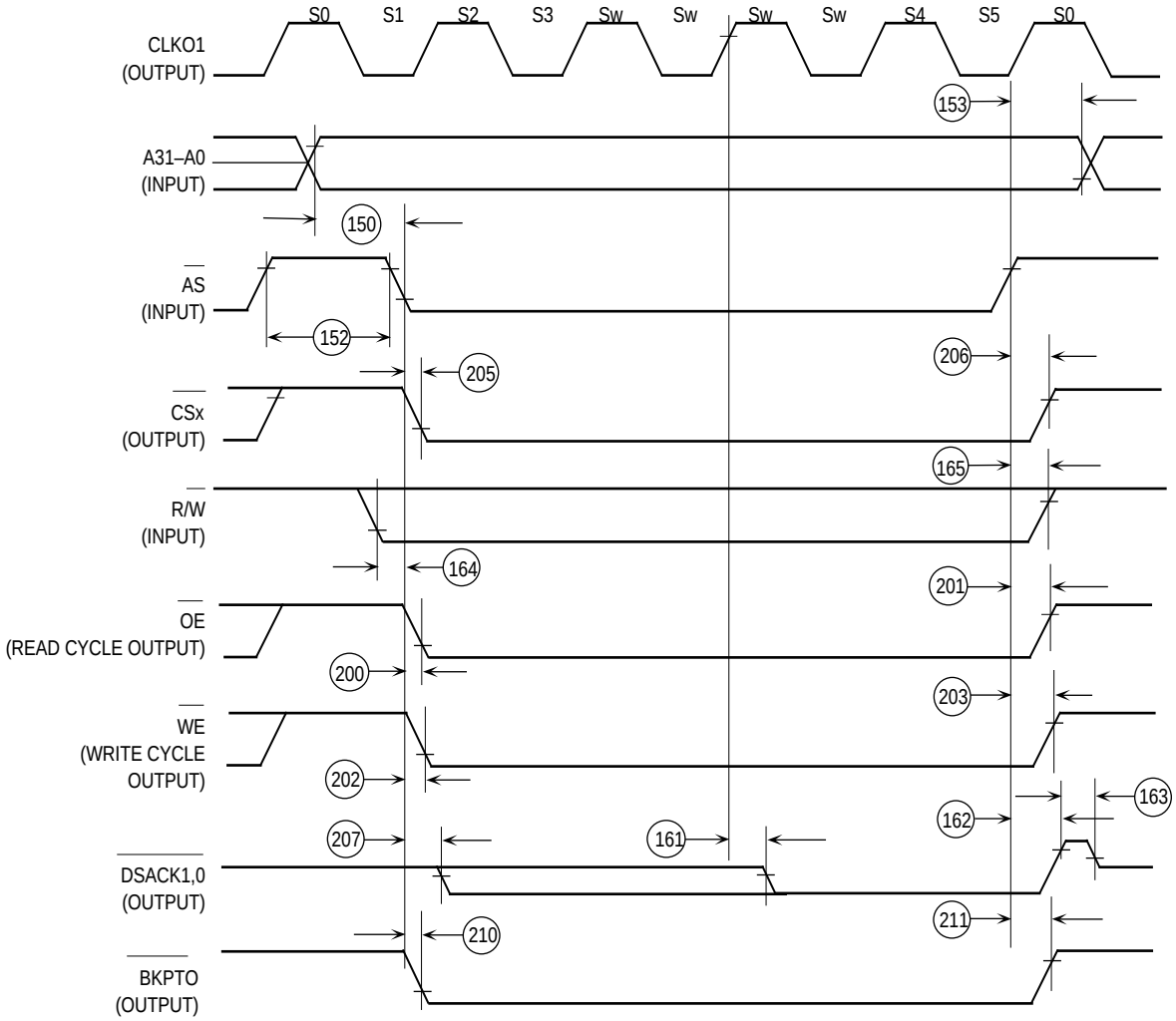


Figure 10-33. External MC68030/MC68360 SRAM Asynchronous Cycle Timing Diagram (BSTN = 0/1; SYNC = 0)

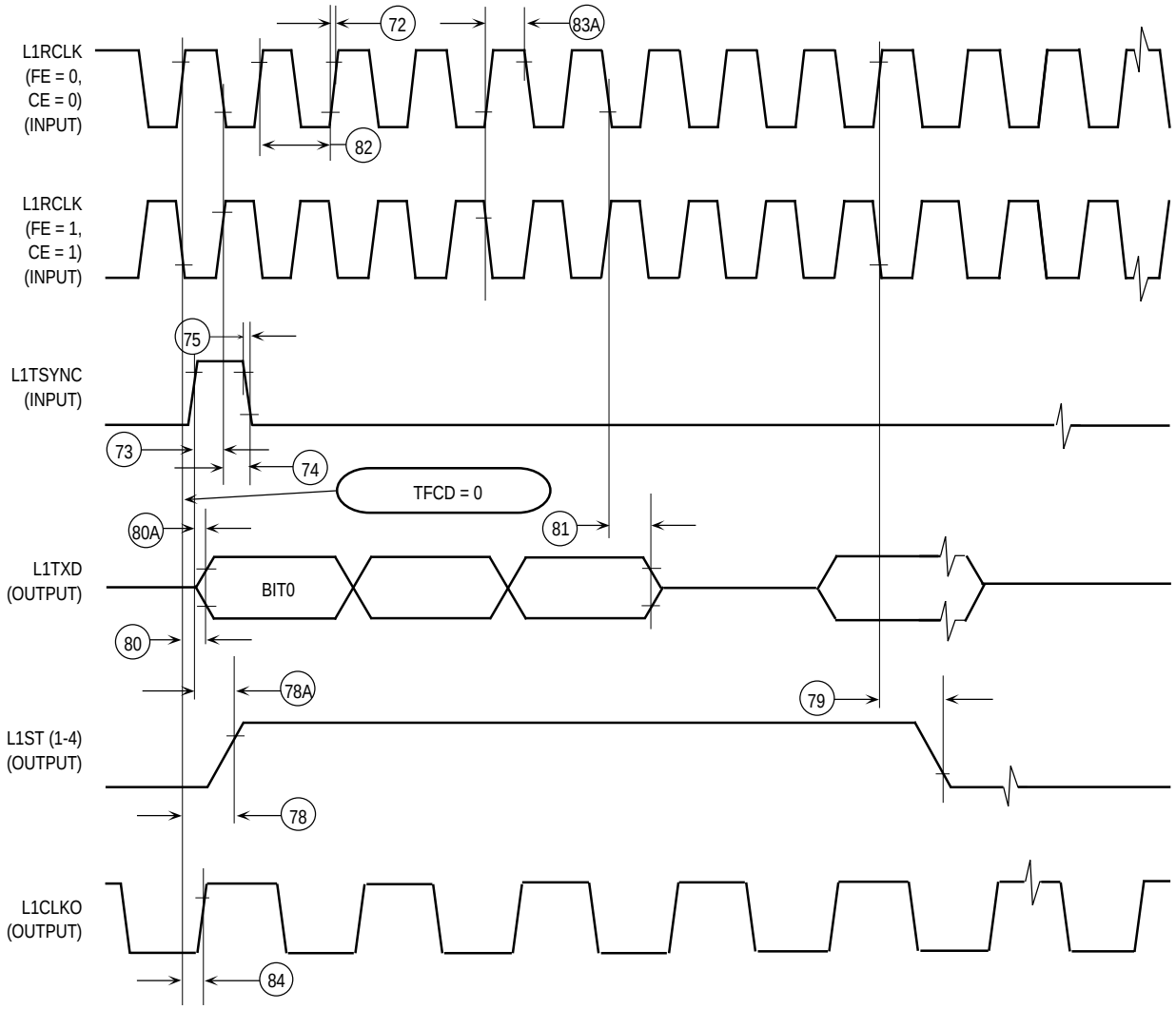


Figure 10-63. SI Transmit Timing with Double Speed Clocking (DSC = 1)

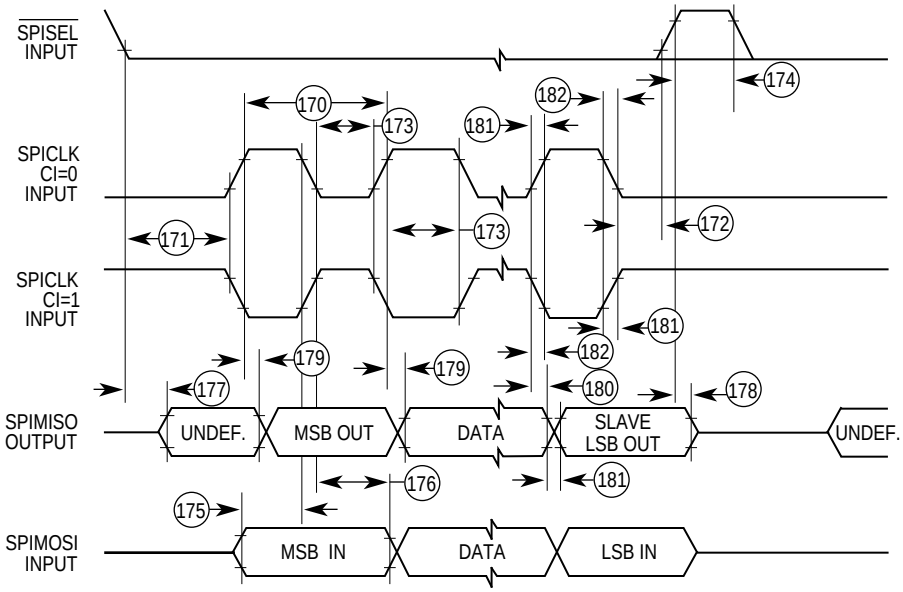


Figure 10-78. SPI Slave (CP = 1)

11.2 PIN ASSIGNMENT—240-LEAD QUAD FLAT PACK (QFP)