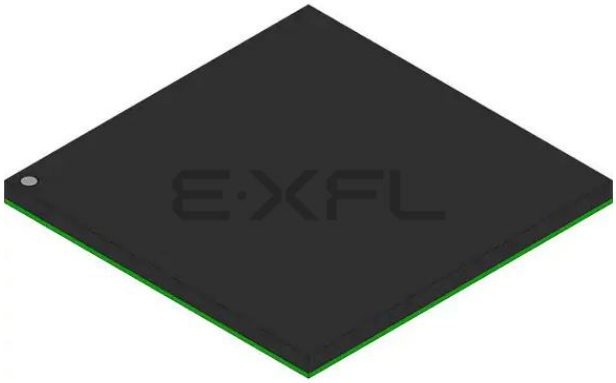


Welcome to E-XFL.COM

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	Active
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/pro/item?MUrl=&PartUrl=mc68mh360zq25vl

CONTENTS

Acronyms and Abbreviations

Table i contains acronyms and abbreviations that are used in this document.

Table i. Acronyms and Abbreviated Terms

Term	Meaning
BD	Buffer descriptor
bps	Bits per second
BRI	Basic rate interface
BRG	Baud rate generator
CPM	Communications processor module
CR	Command register
DCL	Data clock signal
FSC	Frame sync signal
GSM	Global system for mobile communications
GOV	Global receiver overrun (global error)
GUN	Global transmitting underrun (global error)
HDLC	High-level data link control
I^2C	Interprocessor-integrated controller channel
MSC	Multi-subchannel microcode
NMSI	Nonmultiplexed serial interface
QMC	QUICC multichannel controller
QUICC	QUad integrated communication controller
RCCR	RISC controller configuration register
RxBD	Receive buffer descriptor
SCC	Serial communication controller
SCCE	SCC event register
SI	Serial interface routing
SS-7	Signaling system 7
TDM	Time-division multiplexing
TSA	Time slot assigner
TSO	Time slot zero
TxBD	Transmit buffer descriptor

Chapter 1
Overview

This chapter gives an overview of the QMC protocol including some example applications.

1.1 The QMC (QUICC Multichannel Controller)

The QMC protocol emulates up to 64 logical channels within one SCC (serial communication controller) using the same time-division-multiplexed (TDM) physical interface. This multichannel protocol is implemented using the CPM ROM space and additional hardware; it is not a downloadable microcode.

The standard QUICC family members (MC68360[1], MPC860[2], etc.) work in TDM applications but can only support one logical channel per SCC. The parts currently supporting the QMC protocol are a superset to the following devices:

- MC68MH360 is a superset of the MC68EN360[3]
- MPC860MH is a superset of the MPC860EN
- MPC860DH is a superset of the MPC860DE

The QMC parts are pin-compatible with their respective family members. With minor adjustments, they can be used in identical applications such as primary rate ISDN support.

1.2 Introduction

Ideal for E1/T1 applications, the QMC protocol can multiplex any 64-channel combination of subgroups to one TDM interface.

Each of the channels can be separately programmed either to perform HDLC formatting/deformatting or to act as a transparent channel.

Both of the SI serial interfaces (for example, TDM_a or TDM_b) can be dedicated to the QMC protocol. The SI transfers the whole frame to an SCC[4]. Using the CPM RISC, the SCC

[1]MC68360 is trademarked as the QUICC.
[2]MPC860 is trademarked as the PowerQUICC.
[3]On the MC68MH360, protocol support for Centronics and BISYNC have been removed to create space for the QMC microcode.
[4]This is the normal operating mode; however, it is possible to split the TDM stream over several SCCs.

Table 2-9. RSTATE Field Descriptions for 860MH (HDLC)

Field	Name	Description
0–1	—	0
2	—	1
3	**MOT**	Motorola/Intel bit 0 = The bus format is Intel format (little-endian). 1 = The system bus is considered to be organized in Motorola format (big-endian).
4	—	0
5–7	**AT[1–3]**	Address type—This field contains the address type for the transmitter DMA channel for data buffers in external memory (transmit buffers). Address types are needed by the memory controller to decode a correct memory cycle and activate the correct handshaking.

2.4.2 Channel-Specific Transparent Parameters

Table 2-10 describes the channel-specific transparent parameters. Boldfaced parameters must be initialized by the user.

Table 2-10. Channel-Specific Transparent Parameters

Offset	Name	Width	Description
00	**TBASE**	16	Tx buffer descriptor base address—Defines the offset of the channel's transmit BD table relative to MCBASE, host-initialized. See Figure 2-2.
02	**CHAMR**	16	Channel mode register. See Section 2.4.2.1, "CHAMR—Channel Mode Register (Transparent Mode)."
04	**TSTATE**	32	Tx internal state —TSTATE defines the internal Tx state. Host-initialized to 0x3800_0000—FC = 8, Motorola mode for MH360. Host-initialized to 0x3000_0000—AT = 0, Motorola mode for 860MH. Initialize before enabling the channel. See Section 2.4.2.2, "TSTATE—Tx Internal State (Transparent Mode)."
08		32	Tx internal data pointer—Points to current absolute address of channel.
0C	**TBPTR**	16	Tx buffer descriptor pointer (host-initialized to TBASE before enabling the channel or after a fatal error before reinitializing the channel)—Contains the offset of current BD relative to MCBASE. See Table 2-1. MCBASE + TBPTR gives the address for the BD in use.
0E		16	Tx internal byte count—Number of remaining bytes
10	TUPACK	32	(Tx temp) Unpack 4 bytes from 1 long word
14	**ZISTATE**	32	Zero-insertion machine state (host-initialized to 0x0000_0100)—Contains the previous state of the zero-insertion state machine.
18	RES	32	
1C	**INTMSK**	16	Channel's interrupt mask flags. See Figure 2-9.
1E	BDFlags	16	Temp
20	**RBASE**	16	Receive buffer descriptor base offset—Defines the offset of the channel's 64-Kbyte receive BD table relative to MCBASE. Host-initialized. See also Figure 2-2.

2.4.2.3 INTMSK—Interrupt Mask (Transparent Mode)

Each event defined in the interrupt circular queue entry maps directly to a bit in INTMSK as shown in Figure 2-13. There is one mask bit for each event—UN (bit 11), BSY (bit 13), TXB (bit 14) and RXB (bit 15). Bits that do not map to an event are reserved. Reserved bits must be set to zero.

- 0 = No interrupt request is generated and no new entry is written in the circular interrupt table.
- 1 = Interrupts are enabled.

This register is initialized by the host before operation.

INTERRUPT TABLE ENTRY:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
V	W	RES	RES	–	CHANNEL NUMBER					RES	UN	RES	BSY	TXB	RXB
RESET: 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INTMSK:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RESERVED		RESERVED		RESERVED						RES	INTERRUPT MASK BITS				
Reset: 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Note: For the 68360, the bit numbering is reversed. See Appendix A for more information.

Figure 2-13. INTMSK and Interrupt Table Entry (Transparent Mode)

2.4.2.4 TRNSYNC—Transparent Synchronization

In transparent mode, the TRNSYNC register controls the synchronization for single time slots or superchannel applications.

Note

This register has no meaning if the SYNC bit in the channel mode register (CHAMR) is cleared (0).

When sending a transparent message over several time slots, it is necessary to know in which time slot the first byte of data appears.

The TRNSYNC word-length register is divided into two parts with the high byte controlling the first received time slot and the low byte controlling the transmitter synchronization.

Take the example of a superchannel of several time slots:

TSn, TSn + 1, TSn + 2 ..... TSn + x

The algorithm for the receiver byte in decimal is:

(TSn + 1) * 2

The algorithm for the transmit byte in decimal is:

(TSn + x + 1) * 2

The result from these calculations is a decimal value programmed into TRNSYNC.

NOTE

Note that TS*n* is not necessarily the first time slot in the frame. For example, if a superchannel is produced from TS2, TS4, and TS6, the message may be arranged with TS4 holding the first byte, then TS6, and the final byte held in TS2 of the following frame.

The following nine cases in Figure 2-14, named C1 to C9, show different scenarios ranging from a single time slot per logical channel to a superchannel using several time slots. In this application, 24 time slots are routed to this SCC from the SI RAM. After time slot 23, the frame starts with 0 again. The arrow in all the figures illustrates the starting position.

C1 is for a single byte in TS7, so TSn = 7

Rx Byte: (7+1) * 2 = 16
As x = 0, TSn + x = TSn = 7, so
TX Byte: (7 + 1) * 2 = 16

C2 is a single byte in TS23, so TSn = 23. Note that time slot after 23 is 0, so in the calculations below 23 + 1 = 0.

Rx Byte: (23 + 1) * 2 = 0
As x = 0, TSn + x = TSn = 23, so
TX Byte: (23 + 1) * 2 = 0

C3 is a 2-byte pattern TS7, TS8, so TSn = 7

Rx Byte: (7 + 1) * 2 = 16
As x = 1, TSn + x = 8, so
TX Byte: (8 + 1) * 2 = 18

C4 is a 2-byte pattern TS8, TS7, so TSn = 8

Rx Byte: (8+1) * 2 = 16
As x = 1, TSn + x = 7, so
TX Byte: (7 + 1) * 2 = 16

C5 is a 2-byte pattern TS19, TS23, so TSn = 19

Rx Byte: (19 + 1) * 2 = 40
As x = 1, TSn + x = 23, so
TX Byte: (23 + 1) * 2 = 0

C6 is a 2-byte pattern TS23, TS19, so TSn = 23

Rx Byte: (23 + 1) * 2 = 0
As x = 1, TSn + x = 19, so;
TX Byte: (19 + 1) * 2 = 40

C7 is a 4-byte pattern TS20, TS23, TS8, TS9 & TS19, so TSn = 20

Rx Byte: (20 + 1) * 2 = 42
As x = 5, TSn + x = 19, so
TX Byte: (19 + 1) * 2 = 40

C8 is a 4-byte pattern TS8, TS9, TS19, TS20 & TS23, so TSn = 8

Rx Byte: (8 + 1) * 2 = 18
As x = 5, TSn + x = 23, so
TX Byte: (23 + 1) * 2 = 0

C9 is a 4-byte pattern TS19, TS20, TS23, TS8 & TS9, so TSn = 19

Rx Byte: (19 + 1) * 2 = 40
As x = 5, TSn + x = 9, so
TX Byte: (9 + 1) * 2 = 20

NOTE

Case C1 and C2 can be used as described above. A more elegant solution for single time slot applications is to have the SYNC bit cleared (0) in the channel mode register. Both scenarios produce the same result.

For the 860MH, RSTATE should be initialized to 0x3100_0000 before enabling the channel —AT = 1. Note that for the 860MH, bit 4 should always be zero as only bits 5–7 map to AT[1–3]. Table 2-15 describes the RSTATE fields for the 860MH with boldfaced parameters to be initialized by the user.

Table 2-15. RSTATE Field Descriptions for 860MH (Transparent Mode)

Field	Name	Description
0–1	—	0
2	—	1
3	**MOT**	Motorola/Intel bit 0 = The bus format is Intel format (little-endian). 1 = The system bus is considered to be organized in Motorola format (big-endian).
4	—	0
5–7	**AT[1–3]**	Address type—This field contains the address type for the transmitter DMA channel for data buffers in external memory (transmit buffers). Address types are needed by the memory controller to decode a correct memory cycle and activate the correct handshaking.

Chapter 3
QMC Commands

The host issues commands to the QMC by writing to the command register (CR). The QMC commands are similar to those of standard QUICC HDLC protocol. The CR format for QMC is shown in Figure 3-1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RST	QMC OPCODE			1	1	1	0	0	CHANNEL NUMBER					–	FLG

Note: For the 68360, the bit numbering is reversed. See Appendix A for more information.

Figure 3-1. Command Register (CR)

3.1 Transmit Commands

STOP TRANSMIT <channel> (QMC opcode = 001)

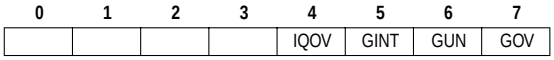
The stop transmit command disables the transmission of data on the selected channel and clears the POL bit in the CHAMR register. Upon asserting this command in the middle of a frame, the RISC processor sends an ABORT indication (7F) followed by IDLEs or FLAGs, depending on the mode, on the selected channel. If this command is issued between frames, the RISC processor continues sending IDLEs or FLAGs (depending on the IDLM mode bit in the CHAMR register) in this channel.

The Tx buffer descriptor pointer (TBPTR) is not advanced to the next buffer; see Table 2-4 and Section 2.2, “Global Multichannel Parameters.”

Set (1) the POL bit to start transmission or to continue after a stop command.

Only after transmission start for a deactivated channel, which is identified by a cleared (0) V bit in the time slot assignment table or a cleared (0) ENT bit, is it necessary to initialize ZISTATE and TSTATE before setting (1) the POL bit.

To deactivate a channel, clear the V bit in the TSA table and the ENT bit in the channel mode register (CHAMR).

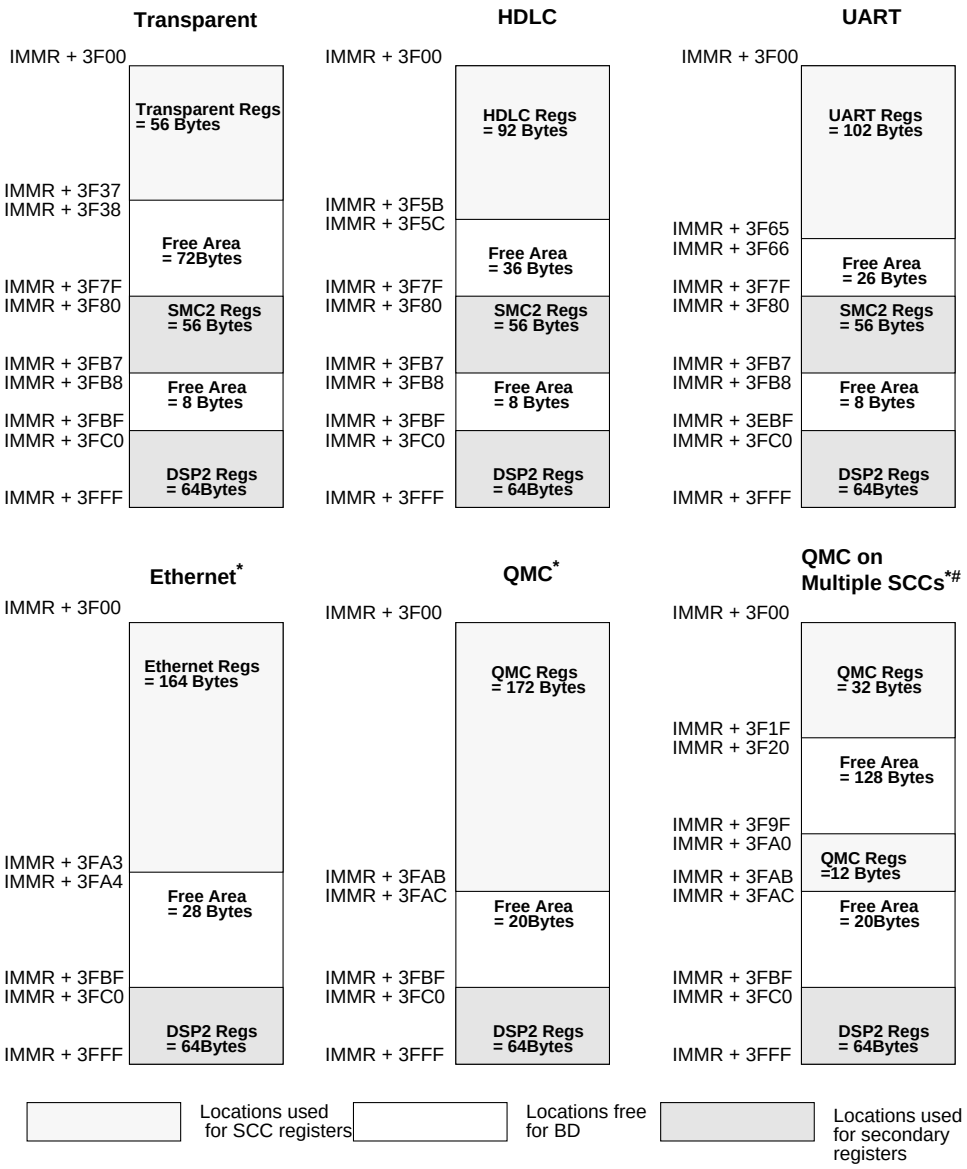


Note: For the 68360, the bit numbering is reversed. See Appendix A for more information.

Figure 4-2. SCC Event Register

Table 4-1. SCC Event Register Field Descriptions

Field	Name	Description
0–3	—	Reserved
4	IQOV	Interrupt table (interrupt queue) overflow 0 No interrupt table overflow has occurred. 1 An overflow condition in the circular interrupt table occurs (and an interrupt request is generated). This condition occurs if the RISC processor attempts to write a new interrupt entry into an entry that was not handled by the host. Such an entry is identified by V = 1. This entry is cleared by the software immediately after entering the interrupt routine. When this occurs, the last interrupt is lost and not overwritten on the first entry.
5	GINT	Global interrupt 0 No global interrupt has occurred. 1 This flag indicates that at least one new entry in the circular interrupt table has been generated by the QMC. The host clears GINT by writing a 1 to its location in SCCE. After clearing it, the host reads the next entry from the circular interrupt table, and starts processing a specific channel's exception. The user must make sure that no more valid interrupts are pending in the interrupt table after clearing the GINT bit, before performing the RTE to avoid deadlock. This procedure ensures that no pending interrupts exist in the queue.
6	GUN	Global transmitter underrun 0 No global transmitter underrun has occurred. 1 This flag indicates that an underrun occurred in the SCC's transmitter FIFO. This error is fatal since it is unknown which channel(s) are affected. Following the assertion of the GUN bit in the SCCE, the QMC stops transmitting data on all channels. The TDM Tx line goes into idle mode. This error affects only the transmitter; the receiver continues to work. After initializing all the individual channels, the host may resume transmitting. If enabled in the SCCM, an interrupt request is generated when GUN is set. The host may clear GUN by writing 1 to its location in the SCCE.
7	GOV	Global receiver overrun 0 No global receiver overrun has occurred. 1 This flag indicates that an overrun occurred in the SCC's transmitter FIFO. This error is fatal since it is unknown which channel(s) are affected. Following the assertion of the GOV bit in the SCCE, the QMC stops receiving data on all channels. Data is no longer written to memory. This error affects only the receiver; the transmitter continues to work. After initializing all the individual channels, the host may resume receiving. If enabled in SCCM, an interrupt request is generated when GOV is set. The host may clear GOV by writing 1 to its location in the SCCE.



#—TSATRx & TSATTx are mapped to an alternative SCC's parameter RAM.
*** —SMC2 is not available in these configurations due to memory conflict.**

Figure 5-15. MPC860MH SCC4 Parameter RAM Usage

Step 16. Initialize channel-specific parameters for HDLC or transparent mode as follows. For more information on HDLC, see Section 2.4.1, “Channel-Specific HDLC Parameters,” and for transparent mode, see Section 2.4.2, “Channel-Specific Transparent Parameters.” Repeat for each of the enabled channels.

- TBASE: TxBD descriptors base address. Initialize to location of first TxBD = MCBASE+TBASE.
  RBASE: RxBD descriptors base address. Initialize to location of first RxBD = MCBASE+RBASE.

```
ch[0].RBASE = 0;              /* locate CH0 RxBDS at 0 */
ch[0].TBASE = 8;              /* locate CH0 TxBDS at 8 */
ch[1].RBASE = 0x10;           /* locate CH1 RxBDS 0x10 */
ch[1].TBASE = 0x18;           /* locate CH1 TxBDS 0x18 */
ch[2].RBASE = 0x20;           /* locate CH2 RxBDS 0x20 */
ch[2].TBASE = 0x28;           /* locate CH2 TxBDS 0x28 */
ch[3].RBASE = 0x30;           /* locate CH3 RxBDS 0x30 */
ch[3].TBASE = 0x38;           /* locate CH3 TXBDS 0x38 */
```

  Copy RBASE to RBPTR (Rx buffer descriptor pointer) and TBASE to TBPTR (Tx buffer descriptor pointer).

```
ch[x].TBPTR = ch[x].TBASE;
ch[x].RBPTR = ch[x].RBASE;
```

- TSTATE: Tx internal state. For the MH360, this is typically 0x3800_0000. For the 860MH, this is typically 0x3000_0000.

```
ch[x].TSTATE = 0x3800_0000;   /* setting for MH360 */
ch[x].TSTATE = 0x3000_0000;   /* setting for 860MH */
```

- RSTATE: Rx internal state. For the MH360, this is typically 0x3900_0000. For the 860MH, this is typically 0x3100_0000.

```
ch[x].RSTATE = 0x3900_0000;   /* setting for MH360 */
ch[x].RSTATE = 0x3100_0000;   /* setting for 860MH */
```

- ZISTATE: zero insertion should be initialized to 0x0000_0100.

```
ch[x].ZISTATE = 0x100;
```

- ZDSTATE: zero deletion machine state should be initialized to 0x1800_0080 for transparent mode or 0x0000_0080 for HDLC.

```
ch[x].ZDSTATE = 0x80;         /* set ZDZTATE for HDLC */
```

- INTMSK: channel’s interrupt mask flags. Bits should be set to enable the corresponding interrupts.

```
ch[x].INTMSK = 0xA;
```

```
struct  descs {
      rxbdq recvbd0;               /* receive buffer 0 */
      txbdq xmitbd0;               /* transmit buffer 0 */
      } chbd[4];                   /* 4 sets of chan descriptors */
static char *poem[6];              /* poem area */
short linecntr;                    /* line counter */
struct intrpten {
      unsigned V:1;                /* entry valid bit */
      unsigned W:1;                /* entry wrap bit */
      unsigned NID:1;              /* not-an-idle has occurred */
      unsigned IDL:1;              /* an idle has occurred */
      unsigned :1;
      unsigned CHNMBR:5;           /* channel number */
      unsigned MRF:1;              /* maximum frame length violation */
      unsigned UN:1;               /* Tx underrun */
      unsigned RXF:1;              /* receive frame */
      unsigned BSY:1;              /* frame discarded, no buffers */
      unsigned TXB:1;              /* buffer transmitted */
      unsigned RXB:1;              /* receive buffer closed */
      }  intrpt[10];               /* interrupt table */
short recvcnt,xmitcnt,othrcnt = 0;  /* interrupt counters */

main()
{
      void SCC1esr();              /* exception service rtn */
      int *pvec;                   /* exception vector pntr */
      char vecblk = 3;             /* vector number block */
      char intlvl = 4;             /* interrupt level */
      pdpr = (struct dprbase *) (getmbar() & 0xFFFFE000);/*  init  dual-ported
      RAM ptr */
      pdpr->CICR.VBA2_VBA0 = (unsigned) (vecblk);/* vecs at vec num 0x60-7F */
      pdpr->CICR.IRL2_IRL0 = (unsigned) (intlvl);/* CPM interrupts level 4 */
      pdpr->CICR.HP4_HP0 = 0x1F;    /* no int priority change */
/* SCdP is zero from reset */
      pdpr->CICR.SCcP = 1;         /* SCC2 to SCCC position */
      pdpr->CICR.SCbP = 2;         /* SCC3 to SCCB position */
      pdpr->CICR.SCaP = 3;         /* SCC4 to SCCA position */
```

Table 9-1. Time Slot Assignment Table Entry Fields for Receive (MSC) (Continued)

Field	Description
L	Last bit—Identifies the last subchannel in a time slot. 0 This is not the last subchannel in the time slot. 1 This is the last sub channel within an 8-bit time slot.The RISC processor handles the next time slot transferred from the TSA in the next request.
Rx channel pointer	This field of the TSATRx entry identifies the data channel that is routed to this time slot. The actual channel pointer is 11 bits long, and contains the starting address of the channel-specific parameter area (address of TBASE). The 5 most-significant bits are taken from TSATRx, and the 6 least-significant bits are always internally set to zero. For the MSC operation, the addressing range is 2 Kbytes.
Mask(0–7)	Mask bits—These 8 bits identify the valid bits in this time slot for subchanneling support. For 8-bit resolution, all mask bits should be set to 1. Any unmasked bit (1) is processed in the receiver for a valid time slot. Any masked bit (0) is ignored by the receiver for a valid channel and no bit counter is affected.

Table 9-2 describes the fields in the time slot assignment table for transmit (TSATTx) for MSC operation.

Table 9-2. Time Slot Assignment Table Entry Fields for Transmit (MSC)

Name	Description
V	Valid bit—The valid bit indicates whether this time slot is valid. 0 Logic 1 is transmitted. If the Tx signal of the TDM interface is programmed to be an open drain output (port B programming), other devices can transmit on nonvalid time slots. 1 Data is transmitted from its associated buffer in combination with the mask bit settings.
W	Wrap bit—The wrap bit identifies the last entry in TSATTx. 0 This is not the last time slot in the frame. 1 The RISC processor wraps around and handles time slot 0 or the first 8 bits of data in the SCC in the next request. The next request is identified by a frame synchronization pulse.
L	Last bit—Identifies the last subchannel in a time slot. 0 This is not the last subchannel in the time slot. 1 This is the last sub channel within an 8-bit time slot.The RISC processor handles the next time slot transferred to the TSA in the next request.
Tx channel pointer	This field of the TSATTx entry identifies a data channel which is routed to this time slot. The actual channel pointer is 11 bits long, and contains the starting address of the channel-specific parameter area (address of TBASE). The 5 most significant bits are taken from TSATTx channel pointer field, and the 6 least significant bits are always internally set to zero. For MSC protocol, the addressing range is 2 Kbytes.
Mask(0–7)	Mask bits—Identifies the valid bits in this time slot for subchanneling support. For 8-bit resolution, all mask bits should be set to 1. For a valid channel with an unmasked bit (1), the bit position is filled according to the protocol. A valid channel with a masked bit (0) transmits a logic high (1).

Figure A-2 shows the 68360 bit numbering for a time slot assignment table for 64-channel common Rx and Tx mapping.

Figure A-2. Time Slot Assignment Table for 64-Channel Common Rx and Tx Mapping

Appendix C
Connecting ISDN Multiple S/T or U Interfaces to QUICC32

Using IDL or GCI protocols, the MC145574 (S/T interface) and the MC145572 (U interface) can be gluelessly interfaced to members of the MC68302 family for low-cost, active-ISDN basic rate terminal applications.

For applications needing to support more than one basic rate interface (BRI), such as LAN/WAN bridges, PBX, line cards or multiple-line terminal adaptors, a system solution using multiple MC145574s or MC145572s can be built around a QUICC32 (MC68MH360).

The QUICC32 and the QMC (QUICC's multichannel controller) protocol are useful for such ISDN applications requiring several logical channels on one physical medium.

This appendix shows how multiple MC145574s or MC145572s can be connected to a QUICC32, describing the level-1 connections and explaining the data flow through the devices.

No software issues are addressed in this appendix.

C.1 The QMC Protocol

Based upon the IDL bus, the QMC protocol implemented on the QUICC32 generates a TDM (time-division multiplexing) bus with programmable time slots for each ISDN interface. With 32 time slots, each carrying 8 consecutive bits forming 64-Kbps channels, a 2-Mbps TDM line (roughly equivalent to a CEPT/E1 link) can be created.

Time slot zero (TS0) is dedicated to the first B1 channel, with TS1 assigned to the first B2 channel and TS2 to the first D channel. Even though only 2 bits are used for signaling, the D channel has 8 bits reserved on the TDM link since the QMC microcode must process data on 8-bit boundaries for correct delineation of channels. The unused 6 bits are masked in the QMC time slot assignment table.

Since the TDM line allows a maximum of 32 channels, the above process of routing channels to time slots (that is, the second B1 channel routed to TS3 and so on) can be repeated for up to 10 BRIs.

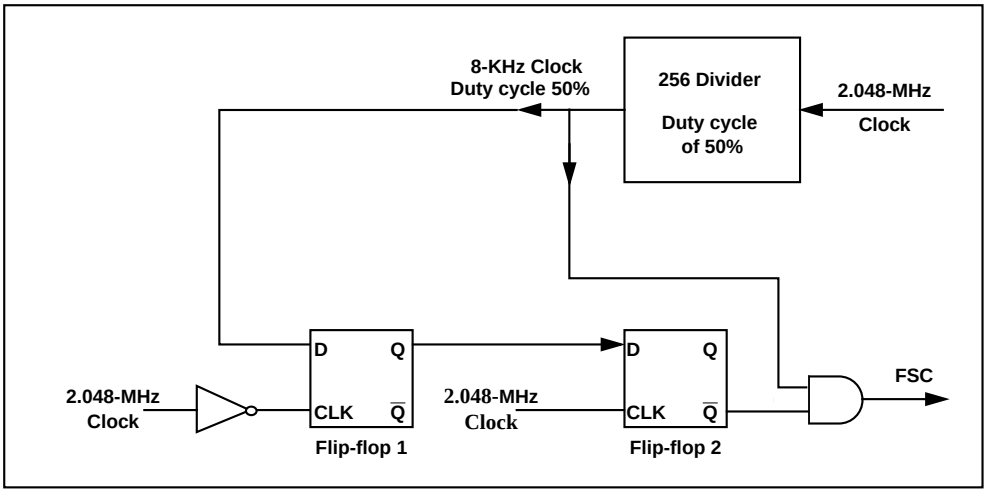


Figure C-4. FSC Generation from a 2.048-MHz Clock—Block Diagram

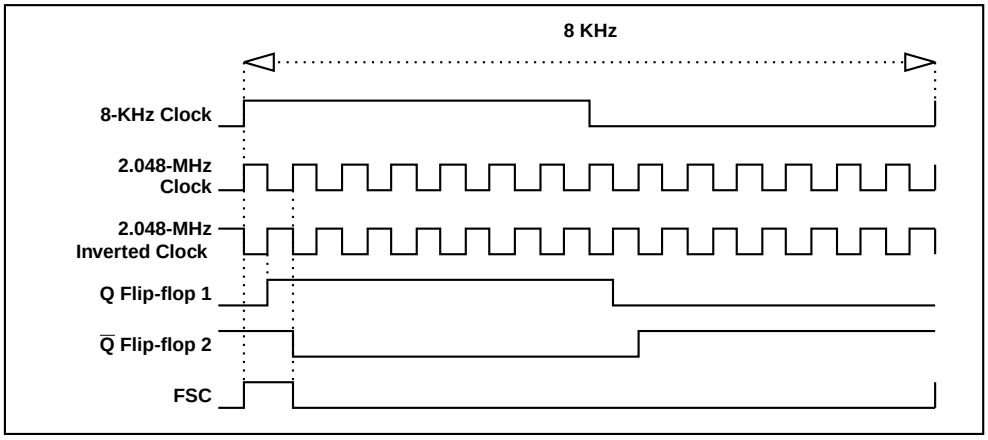


Figure C-5. FSC Generation from a 2.048-MHz Clock—Timing

The S/T interface includes elastic buffers allowing continued operation under any phase relationship between the IDL frame sync and the network. These buffers allow the frame sync to wander with respect to the network up to 60 µs peak-to-peak, exceeding the Q.502 requirements of 18 µs peak-to-peak over a 24-hour period.

Figure C-6 shows a block diagram of the connection between four S/T interfaces and the QUICC32. The diagram would be the same for up to 10 S/T interfaces.

INDEX

INDEX