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/mc68mh360zq25lr2

QMCSUPPLEMENT/AD
8/97

QMC Supplement to
MC68360 and MPC860 User's Manuals

TABLES

works transparently, not participating in any QMC protocol functions. The SCC only performs the parallel-to-serial conversion and adds elasticity through its FIFO memory. The CPM, with its special enhanced microcode and additional dedicated hardware for framing and masking support, does all of the protocol processing for each of the 64 channels. Note that it is executed without intervention from the on-board CPU. Figure 1-1 illustrates the QMC's multichannel capability. Note that each SCC can support up to 64 channels from the TDM; however, there are limitations depending on the device used. This is summarized in Section 1.3, "QMC Features."

Each SCC can work in QMC mode, either alone or together in any combination. The larger FIFO of SCC1 yields the best performance and is therefore recommended for QMC operation. One TDM connection can be routed to one or more SCCs operating in QMC mode, with each SCC operating on different time slots. It is possible to use both TDMs for QMC with combined routing to one SCC or to separate SCCs. When using two TDMs connected to one SCC, restrictions such as using common clocks and sync inputs apply; it is also important to avoid collisions by separating the serial interface (SI) routing.

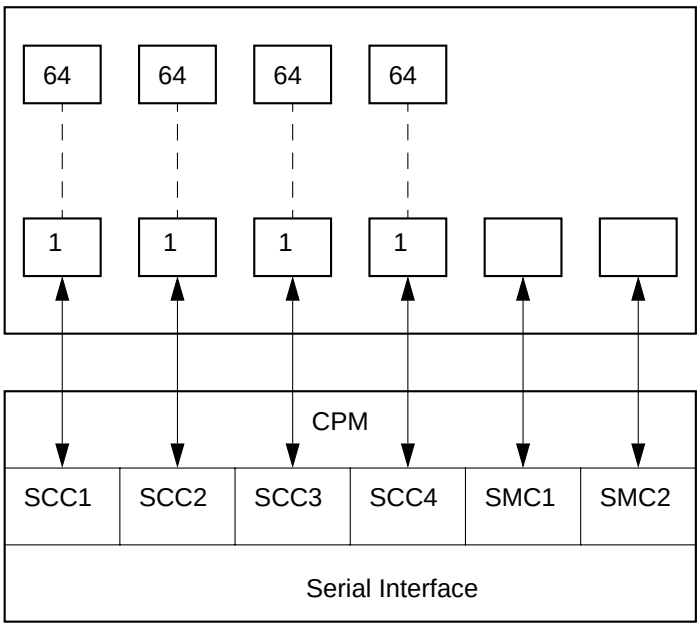


Figure 1-1. QMC Channel Addressing Capability

1.6 QMC Serial Routing and Example Applications

The QMC protocol provides multiple logical channels from a single SCC. The SCC channel dedicated to operate the QMC protocol should have all the relevant bits or time slots routed to it. Individual logical channels are selected by a combination of signals routed through the TDM and tables within the QMC protocol. Contrasting a non-QMC example application with QMC implementations highlights benefits of the multichannel protocol.

Figure 1-2 shows an Ethernet-to-BRI bridge using an MC68EN360, a non-QMC device. The configuration shows the Ethernet routed via an NMSI interface to SCC1. The ISDN BRI is routed via the TSA over an IDL2[1] interface to SCC2–SCC4 for the 2 B + D (B1, B2, and D) channels. The first byte of the frame (B1) is routed to SCC2, the second byte (B2) to SCC3, and then the next two bits (the D channel) to SCC4. In this example, SMC2 is used to connect to a PC over RS232. The internal routing is illustrated in Figure 1-3. Note that three SCCs are required to implement the ISDN BRI. This uses all the MC68EN360's serial channels without efficient use of the CPM bandwidth.

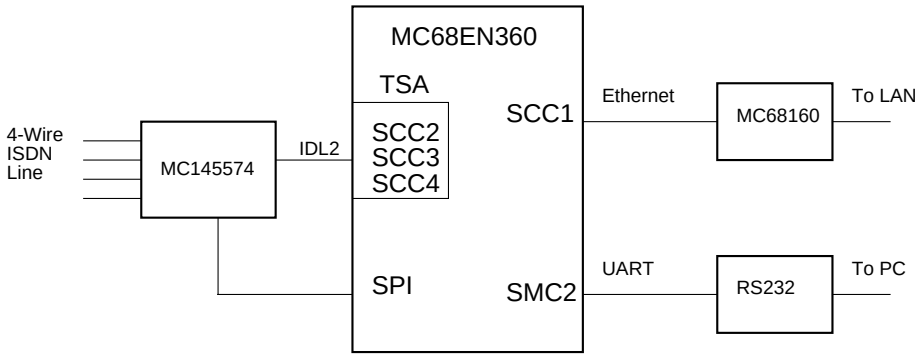


Figure 1-2. Ethernet-to-BRI Bridge Using MC68EN360

[1]The IDL2 interface is a full duplex ISDN interface used to interface to a physical layer device, such as the Motorola ISDN S/T transceiver MC145474.

Table 2-1. Global Multichannel Parameters (Continued)

Offset to SCC Base	Name	Width (Bits)	Description
18	**Rx_S_PTR**	16	Rx time-slot assignment table pointer (default = SCC base + 20 in normal mode)—This global QMC parameter defines the start value of the TSATRx table, which must be present only once per SCC global area. Other SCCs may access this location.
1A	**TxPTR**	16	TxPTR (initialize to SCC Base + 60)—This global parameter is a RISC variable that points to the current transmitter time slot. The host must initialize it to the starting location of TSATTx. The RISC processor increments this pointer whenever it completes the processing of a transmitter time slot.
1C	**C_MASK32**	32	CRC constant (0xDEBB20E3)—Required to calculate 32-bit CRC-CCITT. C_MASK32 is written by the host during QMC initialization. It is used for 32-bit CRC-CCITT calculation if HDLC mode of operation is chosen for a selected channel. (This is a programmable option. For each HDLC channel, one of two CRCs can be chosen, as programmed in CHAMR.) For more information, see Section 2.4.1, “Channel-Specific HDLC Parameters,” and Table 2-5. This entry must have a correct value if at least one HDLC channel is used; otherwise, it can be cleared (0).
20	**TSATRx**	32 Entries x 16	Time slot assignment table Rx—Host-initialized, 16-bit-wide table with 32 entries that define mapping of logical channels to time slots for the QMC receiver. The QMC protocol looks at chunks of 8 bits regardless of whether they come from one physical time slot of the TDM or whatever other combination of bits the TSA transfers to the SCC. These 8 bits are referred to as a time slot in the assignment table. It is recommended but not required to route all bits from the TDM to the SCC and to do all enabling and masking in the time-slot assignment table. See Figure 2-3.
60	**TSATTx**	32 Entries x 16	Time slot assignment table Tx—Maps a specific logical channel to each physical time slot. Time slot assignment table Tx is a host-initialized, 16-bit table with 32 entries that define the mapping of channels to time slots for the QMC transmitter. The QMC protocol looks at chunks of 8 bits regardless if they go to one physical time slot of the TDM or whatever other combination of bits are transferred from the SCC to the TDM through the TSA. These 8 bits are referred to as a time slot in the assignment table. It is recommended but not required to route all bits from the TDM to the SCC and to do all enabling and masking in the time slot assignment table. See Figure 2-3.
A0	**C_MASK16**	16	CRC constant (0xF0B8)—Required to calculate 16-bit CRC-CCITT. This constant is written by the host during QMC initialization. It is used for 16-bit CRC-CCITT calculation if HDLC mode of operation is chosen for a selected channel. (This is a programmable option. For each HDLC channel, one of two CRCs can be chosen, as programmed in CHAMR.) For more information, see Section 2.4.1, “Channel-Specific HDLC Parameters,” and Table 2-5. This entry must have a correct value if at least one HDLC channel is used; otherwise, it can be cleared (0).
A4	TEMP_RBA	32	Temporary receive buffer address
A8	TEMP_CRC	32	Temporary cyclic redundancy check

NOTE

The area between SCC base + 20 and SCC base + 9F is normally used for TSA tables. The mapping above is ideal for 32-channel support. The exact mapping of the TSA tables is determined by the programming of Rx_S_PTR and Tx_S_PTR, and is not fixed. For 64-channel support it is suggested to use common Rx and Tx parameters. The TSA table will be common and have 64 entries starting at SCC base + 20; see Figure 2-4. Alternatively, another SCC's parameter RAM may be used, as determined by Rx_S_PTR and Tx_S_PTR; see Figure 2-6 for more information. However implemented, the TSA tables may reside anywhere in internal memory.

Figure 2-3 shows a general time slot assignment table for 32 16-bit time slots. The fields will be used to either transmit or receive channels.

Time Slot 0	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)
Time Slot 1	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)
	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)
					32 x 16
	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)
	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)
Time Slot 30	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)
Time Slot 31	V	W	Mask(0:1)	Channel Pointer	Mask(2:7)

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

Figure 2-3. Time Slot Assignment Table

Table 2-5 describes the channel mode register's fields for HDLC operation. Boldfaced parameters must be initialized by the user.

Table 2-5. CHAMR Field Descriptions (HDLC)

Field	Name	Description
0	**MODE**	Mode—Each channel has a programmable option whether to use transparent mode or HDLC mode. 0 Transparent mode 1 HDLC mode
1	—	0
2	**IDLM**	Idle mode. 0 Idle mode is disabled. No idle patterns are transmitted between frames. After transmitting the NOF + 1 flags, the transmitter starts the data of the frame. If between frames and the frame buffer is not ready, the transmitter sends flags until it can start transmitting the data. The NOF shall be greater or equal to the PAD setting; see Section 5.2, "Transmit Buffer Descriptor." If NOF = 0, this is identical to flag sharing in HDLC mode. For a high CPM load or with long bus latencies, the QMC protocol may insert additional flags. 1 Idle mode enabled. At least one idle pattern is transmitted between adjacent frames. If between frames and the frame buffer is not ready, the transmitter sends idle characters. When data is ready, the NOF + 1 flags are sent followed by the data frame. If in IDLE mode and NOF = 1, the following sequence is transmitted: .....init value, FF, FF, flag, flag, data,..... The init value before the idle will be 1's, in this case it is assumed the transmitter was uninitialized. An uninitialized SCC transmits 1s in every position.
3	**ENT**	Enable transmit. 0 Disable transmitter. If this bit is cleared and the channel's transmitter is routed to a certain time slot (within TSATTx, see Figure 2-3) the transmitter sends 1's on this time slot. 1 The transmit portion of the channel is enabled and data is sent according to protocol and to other control settings. Note that there is no ENR bit in the QMC protocol. To enable the receiver, the ZDSTATE and RSTATE parameters shall be set to their initial values.
4–6	—	Reserved
7	**POL**	Enable polling. This bit enables the transmitter to poll the transmit buffer descriptors. 0 The CPM does not check the ready bit (R) in the transmit buffer descriptor. 1 The CPM checks the ready bit (R) in the transmit buffer descriptor. The user can use this bit to prevent unnecessary external bus cycles when checking the ready bit (R) in the buffer descriptor. This bit should always be set by the software at the beginning of a transmit sequence of one or more frames. This bit is cleared (0) by the RISC processor when no more buffers are ready in the transmit queue when it finds a buffer descriptor with the R bit cleared (0), i. e., at the end of a frame or at the end of a multiframe transmission. In order to prevent deadlock the software should always prepare the new BD, or multiple BDs, and set (1) the ready bit in the BD, before setting (1) the POL bit. Note that as this bit is automatically cleared by the CPM; the user should not attempt to clear this bit in software.
8	**CRC**	This bit selects the type of CRC when using the HDLC channel mode. 0 16-bit CCITT-CRC is selected for this channel. 1 32-bit CCITT-CRC is selected.
9	—	0

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.

0	1	2	3	4	5	6	7
				IQOV	GINT	GUN	GOV

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

Figure 4-3. SCCM Register

4.3 Interrupt Table Entry

The interrupt table contains information about channel-specific events. Its flags are shown in Figure 4-4. Note that some bits have no meaning when operating in transparent mode. For more detailed description on which bits are used in HDLC and transparent operation, refer to Section 2.4, “Channel-Specific Parameters.” Table 4-2 describes the fields of an interrupt table entry.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
V	W	NID	IDL	—	CHANNEL NUMBER					MRF	UN	RXF	BSY	TXB	RXB
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.

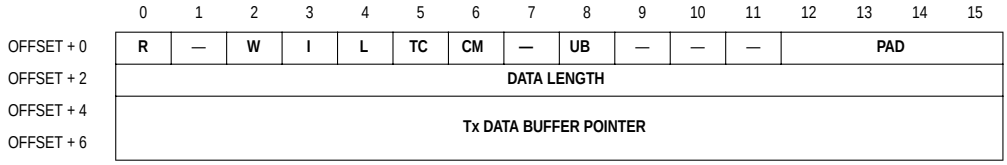
Figure 4-4. Interrupt Table Entry

Table 4-2. Interrupt Table Entry Field Descriptions

Field	Name	Description
0	V	Valid bit 0 = Entry is not valid. 1 = Valid entry containing interrupt information. Upon generating a new entry, the RISC processor sets this bit. The V bit is cleared by the host immediately after it reads the interrupt flags in this entry (before processing the interrupt). The V bits in the queue are host-initialized. During the initialization procedure, the host must clear those bits in all queue entries.
1	W	Wrap bit 0 = This is not the last entry in the circular interrupt table. 1 = This is the last circular interrupt table entry. The next event’s entry is written/read (by RISC/host) from the address contained in INTBASE. During initialization, the host must clear all W bits in the queue except the last one which must be set. The length of the queue is left to the user and can be a maximum of 64 Kbytes.
2	NID	Not idle 0 = No NID event has occurred. 1 = A pattern which is not an idle pattern was identified. NID interrupts are not generated in transparent mode.

5.2 Transmit Buffer Descriptor

Figure 5-3 shows the transmit buffer descriptor.



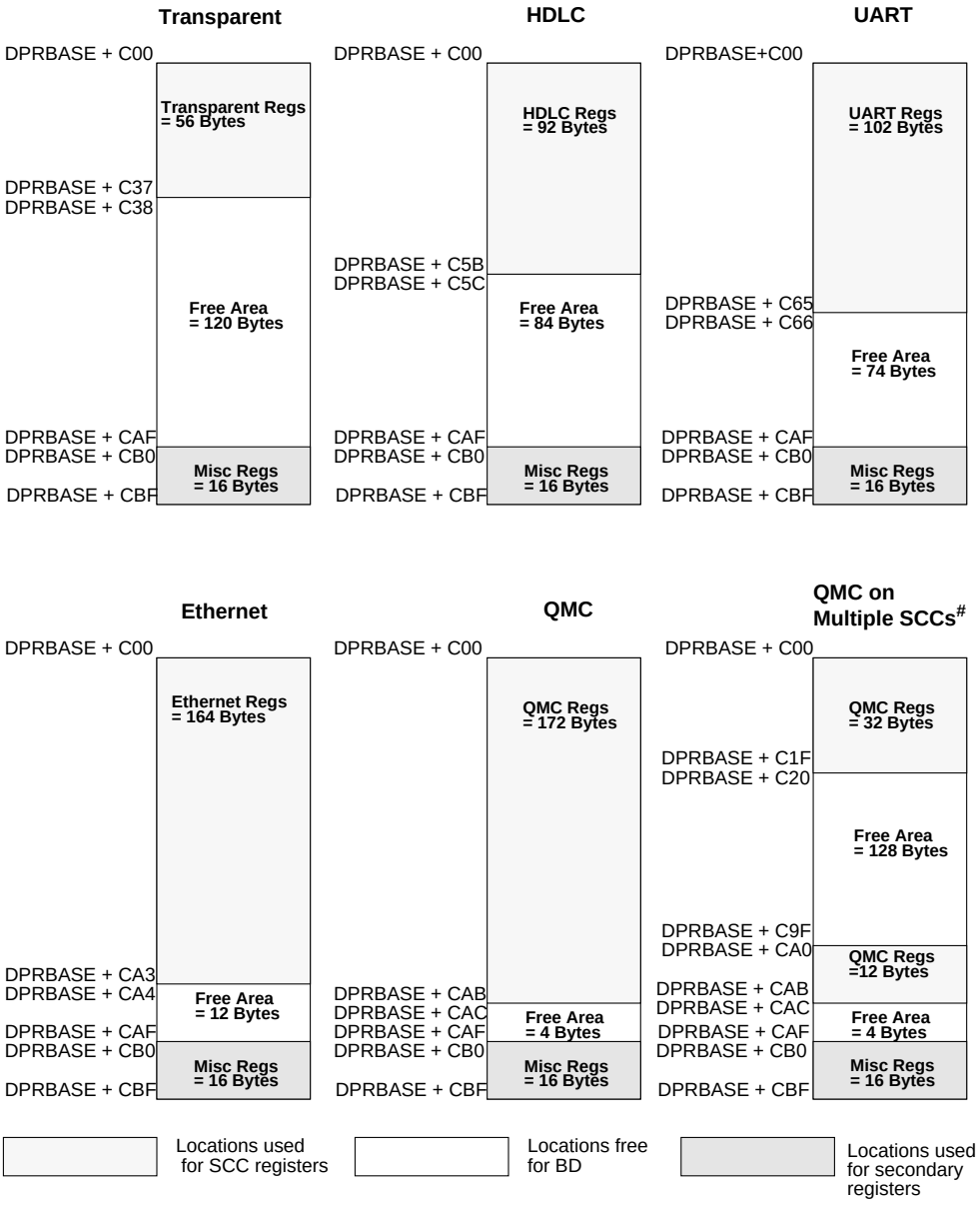
Notes: Entries in boldface must be initialized by the user.
For the 68360, the bit numbering is reversed. See Appendix A for more information.

Figure 5-3. Transmit Buffer Descriptor (TxBD)

Table 5-2 describes the individual fields of a transmit buffer descriptor. Boldfaced entries must be initialized by the user.

Table 5-2. Transmit Buffer Descriptor (TxBD) Field Descriptions

Field	Name	Description
0	**R**	Ready 0 The data buffer associated with this buffer descriptor is not ready for transmission. The user can manipulate this buffer descriptor or its associated data buffer. The CPM clears this bit after the buffer has been transmitted or after an error condition is encountered. 1 The data buffer, which has been prepared for transmission by the user, has not been transmitted or is being transmitted. If R = 1, the user cannot write to fields of this buffer descriptor.
1	—	—
2	**W**	Wrap (final buffer descriptor in table) 0 This is not the last buffer descriptor in the TxBD table. 1 This is the last descriptor in the Tx buffer descriptor table. After this buffer is used, the CPM transmits data from the first buffer descriptor in the table (the buffer descriptor pointed to by TBASE). The number of TxBDs in this table is programmable and is determined only by the wrap bit and the overall space constraints of the dual-ported RAM.
3	**I**	Interrupt 0 No interrupt is generated after this buffer has been serviced. 1 TXB in the circular interrupt table entry is set when the controller services this buffer. This bit can cause an interrupt (if enabled).
4	**L**	Last 0 This is not the last buffer in the frame. 1 This is the last buffer in the current frame.
5	**TC**	Tx CRC (HDLC mode only). This bit is valid only when L = 1; otherwise, it is ignored. 0 Transmit the closing flag after the last data byte. This setting can be used for testing purposes to send an erroneous CRC after the data. 1 Transmit the CRC sequence after the last data byte.



 # —TSATRx & TSATTx are mapped to an alternative SCC's parameter RAM.

Figure 5-7. MC68MH360 SCC1 Parameter RAM Usage

If the QMC operates with a full 64 channels, no space is left in the lower 4-Kbyte area. In this case, the only free areas are in the RAM pages, each 256-bytes large. Depending on the functions, channels and protocols used, some areas remain free for buffer descriptors. Also, if a particular function is not enabled, its parameter RAM area may also be used.

If fewer than 64 logical channels are used or if physical channels are concatenated to super channels, space is freed in the dual-ported RAM. Each unused logical channel creates a 64-byte hole in the dual-ported RAM. This area is free for buffer descriptors for any SCC. QMC channels can also use this space instead of external memory for buffer descriptors, reducing the load on the external bus.

Figure 5-11 shows the internal memory map for the MPC860MH. Figure 5-12 to Figure 5-15 show a more detailed memory map for each SCC, showing the parameter RAM usage for different functions.

- RAM page one is dedicated to SCC1, I^2C, IDMA1 and for miscellaneous storage.
- RAM page two is dedicated for SCC2, SPI, IDMA2 and RISC timers
- RAM page three is dedicated for SCC3, SMC1 and DSP1 operations.
- RAM page four is dedicated for SCC4, SMC2 and DSP2 operations.

Table 5-4 shows the functions available for various protocols on each SCC for the MPC860MH.

Table 5-4. MPC860MH Functions Available

	Function Available?	Transparent	HDLC	UART	Ethernet	QMC	Shared QMC
SCC1	Yes	Misc, I^2C, IDMA1	Misc, I^2C, IDMA1	Misc, I^2C, IDMA1	Misc, IDMA1	Misc, IDMA1	Misc, IDMA1
	No				I^2C	I^2C	I^2C
SCC2	Yes	SPI, Timer, IDMA2	SPI, Timer, IDMA2	SPI, Timer, IDMA2	Timer, IDMA2	Timer, IDMA2	Timer, IDMA2
	No				SPI	SPI	SPI
SCC3	Yes	SMC1, DSP1	SMC1, DSP1	SMC1, DSP1	DSP1	DSP1	DSP1
	No				SMC1	SMC1	SMC1
SCC4	Yes	SMC2, DSP2	SMC2, DSP2	SMC2, DSP2	DSP2	DSP2	DSP2
	No				SMC2	SMC2	SMC2

Figure 5-12 to Figure 5-15 show that not all the functions available on each SCC can be used simultaneously due to overlaps of the register locations stored in the parameter RAM.

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 */
```

```
      pvec = (int *) (getvbr() + (((vecblk << 5) + 0x1E) * 4));/*    calculate
      vector address */
      *pvec = (int) SCC1esr;        /* init int vector */
      pdpr->TRR4 = 8;               /* init timer ref */
      pdpr->TMR4.PS = 0;            /* init prescalar */
      pdpr->TMR4.OM = 1;            /* select toggle */
      pdpr->TMR4.FRR = 1;           /* restart after ref reac */
      pdpr->TMR4.ICLK = 1;          /* master clock */
      pdpr->TGCR.RST4 = 1;          /* enable timer */
      pdpr->PAPAR |= 0x8000         /* enable TOUT4 pin */
      pdpr->PADIR |= 0x8000;        /* pin 15 is an output */
      pdpr->TRR3 = 174;             /* init timer ref */
      pdpr->TMR3.PS = 0;            /* init prescalar */
      pdpr->TMR3.OM = 0;            /* select pulse */
      pdpr->TMR3.FRR = 1;           /* restart after ref reac */
      pdpr->TMR3.ICLK = 3;          /* external clock */
      pdpr->TGCR.RST3 = 1;          /* enable timer */
      pdpr->PAPAR |= 0x3000;/* enable TOUT3 & TIN3 */
      pdpr->PADIR |= 0x2000;        /* pin 13 is an  output */
/* SDCR has been initialized by the debugger */
      pdpr->SIMODE.CRTa = 1;        /* common syncs & clocks */
      pdpr->SIMODE.RFSDa = 1;       /* receive frame sync 1 clk dly */
      pdpr->SIMODE.TFSDa = 1;       /* xmit frame sync 1 clk dly */
      pdpr->SICR.SC1 = 1;           /* SCC1 is TDM */
      pdpr->PAPAR |= 0x100;         /* enable L1RCLKa */
/* PAPAR, PADIR & PAODR are cleared at reset */
      pdpr->PCPAR = 0x800;          /* enable L1RSYNCa */
/* Port C registers are cleared at reset */
      pdpr->SIGMR = 4;              /* enable TDMa */
      pdpr->SIRAM[0] = 0x8042;      /* init 1st receive element */
      pdpr->SIRAM[1] = 0x8042;      /* init 2nd receive element */
      pdpr->SIRAM[2] = 0x8042;      /* init 3rd receive element */
      pdpr->SIRAM[3] = 0x8043;      /* init 4th receive element */
      pdpr->SIRAM[64] = 0x8042;     /* init 1st xmit element */
      pdpr->SIRAM[65] = 0x8042;     /* init 2nd xmit element */
      pdpr->SIRAM[66] = 0x8042;     /* init 3rd xmit element */
      pdpr->SIRAM[67] = 0x8043;     /* init 4th xmit element */
```

Example #1

A device is operating at 25 MHz. SCC1 runs 1x10-Mbps Ethernet in half duplex, SCC2 runs 1 x 2-Mbps HDLC, SCC3 runs 1 x 64-Kbps HDLC, SCC4 runs 1 x 9.6-Kbps UART and SMC1 runs 1 x 38-Kbps SMC UART. The following equation applies:

$$\left(\frac{10}{22}\right)+\left(\frac{2}{8}\right)+\left(\frac{0.064}{2.4}\right)+\left(\frac{0.0096}{2.4}\right)+\left(\frac{0.038}{0.22}\right) = 0.96 \qquad \textbf{(<1)}$$

This yields a percentage CPM utilization of 96% meaning the device can handle these protocols at this frequency. Note the 9.6-Kbps UART link only requires 0.4% of the CPM bandwidth, implying that in any configuration where there is free bandwidth that it will be possible to run a low-rate UART link.

Example #2

A device operating at 25 MHz is required to run 24 channels at 64 Kbps in QMC mode with an additional 2 HDLC channels operating at 128 Kbps each. The following equation applies:

$$\left(\frac{2\times 0.128}{8}\right)+\left(\frac{24\times 0.064}{2.1}\right) = 0.76 \qquad \textbf{(<1)}$$

Example #3

The last example shows an application with 32 QMC channels and one additional 2-Mbps HDLC channel. The following equation applies:

$$\left(\frac{2}{8}\right)+\left(\frac{32\times 0.064}{2.1}\right) = 1.22 \qquad \textbf{(will not work)}$$

Since the result above is greater than 1, this configuration may not work at 25 MHz. If a 33-MHz operation is used, CPM utilization will drop below 1, allowing the combination to be supported. The following equation applies:

$$1.22\times\left(\frac{25}{33}\right) = 0.92 \qquad \textbf{(<1)}$$

In general, a channel combination will work if the combined load is less than 1. The equations are scalable to frequency with the exception of the nonlinear Ethernet protocol. Taking Ethernet into account is difficult. Designers will need to benchmark performance for results near 1.

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.

C.3.3.2 U-Interface Configuration

Do the following for U-interface configuration:

- IDL2 with time slot assigner (TSA enabled in reg. OR6[5–7]; TSA selection in reg. OR0 to OR5)
- Slave mode (DCL & FSC are input) - (pin M/$\overline{S}$ to GND)
- FREQREF enabled at 2.048 MHz (reg. OR8[4] = 1)

C.3.3.3 QUICC32 Configuration

Do the following for QUICC32 configuration:

- SCC3 using the QMC protocol for handling the different channels of the multiplexed IDL2 bus. (D channels are HDLC encoded/decoded and B-channels can be configured for transparent or HDLC framing)
- SCC1 can be configured for Ethernet, HDLC, transparent, or UART.
- SCC2 and SCC4 can be configured for HDLC, transparent, or UART.
- The SPI is connected to the SCP port of each S/T or U interface for handling configuration and control information.
- For the U interface, the SPI/SCP connection can be replaced by a connection of the 8-bit parallel port of the U transceiver to the processor bus of the QUICC.
- One I/O signal can be dedicated for handling the $\overline{SCPEN}$ signal of each S/T or U interface.
- One interrupt signal can be dedicated for handling the $\overline{IRQ}$ signal of each S/T or U interface.

INDEX