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	M68000
Number of Cores/Bus Width	1 Core, 8/16-Bit
Speed	16MHz
Co-Processors/DSP	Communications; RISC CPM
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	3.3V
Operating Temperature	-40°C ~ 85°C (TA)
Security Features	-
Package / Case	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68302cag16vcr2

3.2.5.3 Interrupt Mask Register (IMR)

Each bit in the 16-bit IMR corresponds to an INRQ interrupt source. The user masks an interrupt source by clearing the corresponding bit in the IMR. When a masked INRQ interrupt occurs, the corresponding bit in the IPR is set, but the IMR bit prevents the interrupt request from reaching the M68000 core. If an INRQ source is requesting interrupt service when the user clears the IMR bit, the request to the core will cease, but the IPR bit remains set. If the IMR bit is then set later by the user, the pending interrupt request will once again request interrupt service and will be processed by the core according to its assigned priority. The IMR, which can be read by the user at any time, is cleared by reset.

It is not possible to mask the ERR INRQ source in the IMR. Bit 0 of the IMR is undefined.

NOTE

If a bit in the IMR is masked at the same time that the interrupt at level 4 is causing the M68000 core to begin the interrupt acknowledge cycle, then the interrupt is not processed, and one of two possible cases will occur. First, if other unmasked interrupts are pending at level 4, then the interrupt controller will acknowledge the interrupt with a vector from the next highest priority unmasked interrupt source. Second, if no other interrupts are pending at level 4, then the interrupt controller will acknowledge the interrupt with the error vector (00000 binary).

To avoid handling the error vector, the user can raise the interrupt mask in the M68000 core status register (SR) to 4 before masking the interrupt source and then lower the level back to its original value. Also, if the interrupt source has multiple events (e.g., SCC1), then the interrupts for that peripheral can be masked within the peripheral mask register.

NOTE

To clear bits that were set by multiple interrupt events, the user should clear all the unmasked events in the corresponding on-chip peripheral's event register.

15	14	13	12	11	10	9	8
PB11	PB10	SCC1	SDMA	IDMA	SCC2	TIMER1	SCC3

7	6	5	4	3	2	1	0
PB9	TIMER2	SCP	TIMER3	SMC1	SMC2	PB8	—

address match exists within its address space and, therefore, whether to assert the chip-select line.

111 = Not supported; reserved. Chip select will not assert if this value is chosen.
110 = Value may be used.
•
•
•
000 = Value may be used.

After system reset, the FC field in BR3–BR0 defaults to supervisor program space (FC = 110) to select a ROM device containing the reset vector. Because of the priority mechanism and the EN bit, only the $\overline{CS0}$ line is active after a system reset.

NOTE

The FC bits can be masked and ignored by the chip-select logic using CFC in the OR.

Bits 12–2—Base Address

These bits are used to set the starting address of a particular address space. The address compare logic uses only A23–A13 to cause an address match within its block size. The base address should be located on a block boundary. For example, if the block size is 64k bytes, then the base address should be a multiple of 64k.

After system reset, the base address defaults to zero to select a ROM device on which the reset vector resides. All base address values default to zero on system reset, but, because of the priority mechanism, only $\overline{CS0}$ will be active.

NOTE

All address bits can be masked and ignored by the chip-select logic through the base address mask in the OR.

RW—Read/Write

0 = The chip-select line is asserted for read operations only.
1 = The chip-select line is asserted for write operations only.

After system reset, this bit defaults to zero (read-only operation).

NOTE

This bit can be masked and ignored by the read-write compare logic, as determined by MRW in the OR. The line is then asserted for both read and write cycles.

On write protect violation cycles (RW = 0 and MRW = 1), $\overline{BERR}$ will be generated if WPVE is set, and WPV will be set.

If the write protect mechanism is used by an external master, the R/$\overline{W}$ low to $\overline{AS}$ asserted timing should be 16 ns minimum.

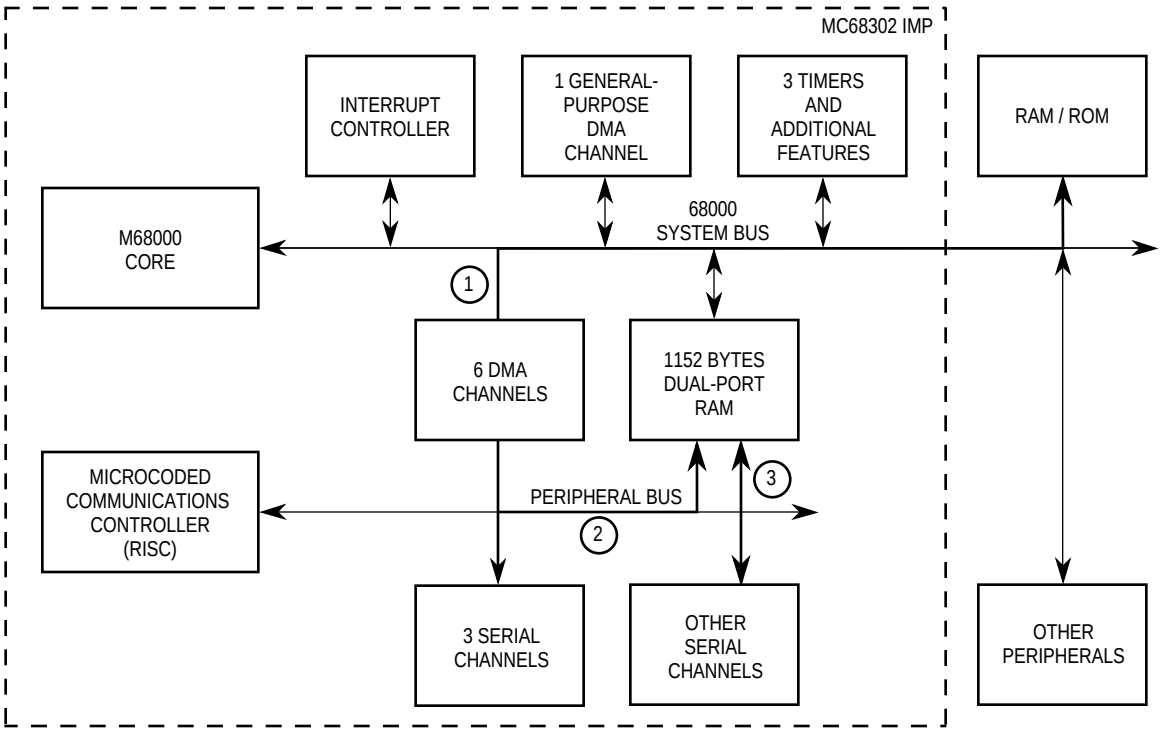


Figure 4-2. Three Serial Data Flow Paths

The SDMA channels implement bus-cycle-stealing data transfers controlled by microcode in the CP main controller. Having no user-accessible registers associated with them, the channels are effectively controlled by the choice of SCC configuration options.

When one SDMA channel needs to transfer data to or from external memory, it will request the M68000 bus with the internal signal SDBR, wait for SDBG, and then only assert the external signal $\overline{\text{BGACK}}$ (see 3.8.5 Bus Arbitration Logic). It remains the bus master for only one bus cycle. The six SDMA channels have priority over the IDMA controller. If the IDMA is bus master when an SDMA channel needs to transfer over the M68000 bus, the SDMA will steal a cycle from the IDMA with no arbitration overhead while $\overline{\text{BGACK}}$ remains continuously low and $\overline{\text{BCLR}}$ remains high. Each SDMA channel may be programmed with a separate function code, if desired. The SDMA channel will read 16 bits at a time. It will write 8 bits at a time except during the HDLC or transparent protocols where it writes 16 bits at a time. Each bus cycle is a standard M68000-type bus cycle. The chip select and wait state generation logic on the MC68302 may be used with the SDMA channels.

NOTE

When external buffer memory is used, the M68000 bus arbitration delay must be less than what would cause the SCC internal FIFOs to overrun or underrun. This aspect is discussed in more detail in 4.5 Serial Communication Controllers (SCCs) and in Appendix A SCC Performance.

L1CLK	IDL clock; input to the IMP.
L1TXD	IDL transmit data; output from the IMP. Valid only for the bits that are supported by the IDL; three-stated otherwise.
L1RXD	IDL receive data; input to the IMP. Valid for the 20 bits of the IDL; ignored for other signals that may be present.
L1SY1	IDL SYNC signal; input to the IMP. This signal indicates that the 20 clock periods following the pulse designate the IDL frame.
L1RQ	Request permission to transmit on the D channel; output from the IMP.
L1GR	Grant permission to transmit on the D channel; input to the IMP.
SDS1	Serial data strobe 1
SDS2	Serial data strobe 2

NOTE

The IDL bus signals, L1TXD and L1RXD, require pull-up resistors in order to ensure proper operation with transceivers.

In addition to the 144-kbps ISDN 2B + D channels, IDL provides channels for maintenance and auxiliary bandwidth. The IDL bus has five channels:

B1	64-kbps Bearer Channel
B2	64-kbps Bearer Channel
D	16-kbps Signaling Channel
M	8-kbps Maintenance Channel (not required by IDL)
A	8-kbps Auxiliary Channel (not required by IDL)

The IMP supports all five channels of the IDL bus. The following table shows where each channel can be routed. The two B channels can be concatenated and routed to the same SCC channel.

IDL Channel	Serial Controllers
D	SCC1, SCC2, SCC3
B1	SCC1, SCC2, SCC3
B2	SCC1, SCC2, SCC3
M	SMC1
A	SMC2

The IMP supports the request-grant method for contention detection on the D channel. When the IMP has data to transmit on the D channel, it asserts L1RQ. The physical layer device monitors the physical layer bus for activity on the D channel and indicates that the channel is free by asserting L1GR. The IMP samples the L1GR signal when L1SY1 is asserted. If L1GR is high (active), the IMP transmits the first zero of the opening flag in the first bit of the D channel. If a collision is detected on the D channel, the physical layer device negates L1GR. The IMP then stops its transmission and retransmits the frame when L1GR is asserted again. This is handled automatically for the first two buffers of the frame.

The monitor channel is used to transfer data between layer-1 devices and the control unit (i.e., the M68000 core). The command/indication channel is used to control activation/deactivation procedures or for the switching of test loops by the control unit.

The IMP supports all five channels of the GCI channel 0. The following table shows where each channel can be routed. The two B channels can be concatenated and routed to the same SCC channel.

GCI Channel 0	Serial Controllers
D	SCC1, SCC2, SCC3
B1	SCC1, SCC2, SCC3
B2	SCC1, SCC2, SCC3
M	SMC1
C/I	SMC2

The GCI interface supports the CCITT I.460 recommendation for data rate adaptation. The GCI interface can access each bit of the B channel as an 8-kbps channel. The mask register (SIMASK) for the B channels specifies which bits are supported by the GCI interface. The receiver will receive only the bits that are enabled by SIMASK; the transmitter will transmit only the bits that are enabled by SIMASK and will not drive the L1TXD pin otherwise (L1TXD in GCI mode is an open-drain output).

The IMP supports contention detection on the D channel. When the IMP has data to transmit on the D channel, it checks bit 4 of the SCIT C/I channel 2. The physical layer device monitors the physical layer bus for activity on the D channel and indicates with this bit that the channel is free. If a collision is detected on the D channel, the physical layer device sets bit 4 of C/I channel 2 to logic high. The IMP then aborts its transmission and retransmits the frame when this bit is asserted again. This procedure is handled automatically for the first two buffers of a frame. The L1GR line may also be used for access to the S interface D channel. This signal is checked by the IMP, and the physical layer device should indicate that the S interface D channel is free by asserting L1GR.

In the deactivated state, the clock pulse is disabled, and the data line is a logic one. The layer-1 device activates the IMP by enabling the clock pulses and by an indication in the channel 0 C/I channel. The IMP will then report to the M68000 core by a maskable interrupt that a valid indication is in the SMC2 receive buffer descriptor.

When the M68000 core activates the line, it sets SETZ in the serial interface mode (SIMODE) register, causing the data output from L1TXD to become a logic zero. Code 0 (command timing TIM) will be transmitted on channel 0 C/I channel to the layer-1 device until the SETZ is reset. The physical layer device will resume transmitting the clock pulses and will give an indication in the channel 0 C/I channel. The M68000 core should reset SETZ to enable data output.

4.4.3 PCM Highway Mode

In PCM highway mode, one, two, or all three SCCs can be multiplexed together to support various time-division multiplexed interfaces. PCM highway supports the standard T1 and CEPT interfaces as well as user-defined interfaces. In this mode, the NMSI1 pins have new names and functions (see Table 4-2).

4.5.11.6 UART Address Recognition

In multidrop systems, more than two stations may be present on a network, with each having a specific address. Figure 4-18 shows two examples of such a configuration. Frames comprised of many characters may be broadcast, with the first character acting as a destination address. To achieve this, the UART frame is extended by one bit, called the address bit, to distinguish between an address character and the normal data characters. The UART can be configured to operate in a multidrop environment in which two modes are supported:

Automatic Multidrop Mode—The IMP automatically checks the incoming address character and accepts the data following it only if the address matches one of two 8-bit preset values. In this mode, UM1–UM0 = 11 in the UART mode register.

Nonautomatic Multidrop Mode—The IMP receives all characters. An address character is always written to a new buffer (it may be followed by data characters in the same buffer). In this mode, UM1–UM0 = 01 in the UART mode register.

Each UART controller has two 8-bit address registers (UADDR1 and UADDR2) for address recognition. In the automatic mode, the incoming address is checked against the lower order byte of the UART address registers. Upon an address match, the address match (M) bit in the BD is set/cleared to indicate which address character was matched. The data following it is written to the same data buffer.

NOTE

For 7-bit characters, the eighth bit (bit 7) in UADDR1 and UADDR2 should be zero.

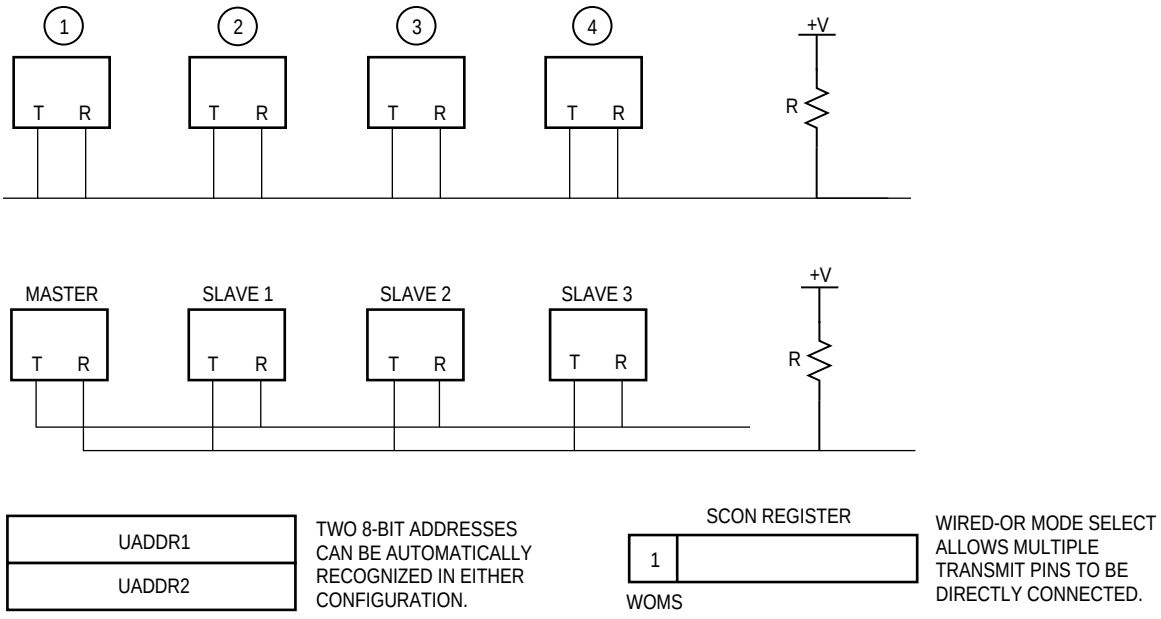


Figure 4-18. Two Configurations of UART Multidrop Operation

ENC—Data Encoding Format

0 = Non-return to zero (NRZ). A one is a high level; a zero is a low level.
1 = Non-return to zero inverted (NRZI). A one is represented by no change in the level; a zero is represented by a change in the level. The receiver decodes NRZI, but a clock must be supplied. The transmitter encodes NRZI. During an idle condition, with the FLG bit cleared, the line will be forced to a high state.

COMMON SCC MODE BITS—See 4.5.3 SCC Mode Register (SCM) for a description of the DIAG1, DIAG0, ENR, ENT, MODE1, and MODE0 bits.

4.5.12.10 HDLC Receive Buffer Descriptor (Rx BD)

The HDLC controller uses the Rx BD to report information about the received data for each buffer. The Rx BD is shown in Figure 4-26.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFFSET + 0	E	X	W	I	L	F	—	—	—	—	LG	NO	AB	CR	OV	CD
OFFSET + 2	DATA LENGTH															
OFFSET + 4	RX BUFFER POINTER (24-bits used, upper 8 bits must be 0)															
OFFSET +6																

Figure 4-26. HDLC Receive Buffer Descriptor

An example of the HDLC receive process is shown in Figure 4-27. This shows the resulting state of the Rx BDs after receipt of a complete frame spanning two receive buffers and a second frame with an unexpected abort sequence. The example assumes that MRBLR = 8 in the SCC parameter RAM.

The first word of the Rx BD contains control and status bits. Bits 15–10 are written by the user before the buffer is linked to the Rx BD table, and bits 5–0 are set by the CP following frame reception. Bit 15 is set by the M68000 core when the buffer is available to the HDLC controller; it is cleared by the HDLC controller when the buffer is full.

E—Empty

0 = The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The M68000 core is free to examine or write to any fields of the BD.
1 = The data buffer associated with the BD is empty. This bit signifies that the BD and its associated buffer are available to the HDLC controller. The M68000 core should not write to any fields of this BD after it sets this bit. The empty bit will remain set while the HDLC controller is currently filling the buffer with received data.

X—External Buffer

0 = The buffer associated with this BD is in internal dual-port RAM.
1 = The buffer associated with this BD is in external memory.

IDL—IDLE Sequence Status Changed

A change in the status of the serial line was detected on the HDLC channel. The SCC status register may be read to determine the current status.

TXE—Tx Error

An error ($\overline{\text{CTS}}$ lost or underrun) occurred on the transmitter channel.

RXF—Rx Frame

A complete frame has been received on the HDLC channel. This bit is set no sooner than two receive clocks after receipt of the last bit of the closing flag.

BSY—Busy Condition

A frame was received and discarded due to lack of buffers.

TXB—Tx Buffer

A buffer has been transmitted on the HDLC channel. This bit is set no sooner than when the second-to-last bit of the closing flag begins its transmission, if the buffer is the last in the frame. Otherwise, it is set after the last byte of the buffer has been written to the transmit FIFO.

RXB—Rx Buffer

A buffer has been received on the HDLC channel that was not a complete frame. This bit will only be set if the I bit in the Tx BD was set.

4.5.12.13 HDLC Mask Register

The SCC mask register (SCCM) is referred to as the HDLC mask register when the SCC is operating as an HDLC controller. It is an 8-bit read-write register that has the same bit formats as the HDLC event register. If a bit in the HDLC mask register is a one, the corresponding interrupt in the event register will be enabled. If the bit is zero, the corresponding interrupt in the event register will be masked. This register is cleared upon reset.

4.5.13 BISYNC Controller

The byte-oriented binary synchronous communication (BISYNC) protocol was originated by IBM for use in networking products. The three classes of BISYNC frames are transparent, non-transparent with header, and non-transparent without header (see Figure 4-30). The transparent mode in BISYNC allows full binary data to be transmitted, with any possible character pattern allowed. Each class of frame starts with a standard two octet synchronization pattern and ends with a block check code (BCC). The end of text character (ETX) is used to separate the text and BCC fields.

NON TRANSPARENT WITH HEADER

SYN1	SYN2	SOH	HEADER	STX	TEXT	ETX	BCC

NON TRANSPARENT WITHOUT HEADER

SYN1	SYN2	STX	TEXT	ETX	BCC

TRANSPARENT

SYN1	SYN2	DLE	STX	TRANSPARENT TEXT	DLE	ETX	BCC

Figure 4-30. Typical BISYNC Frames

The bulk of the frame is divided into fields whose meaning depends on the frame type. The BCC is either a 16-bit CRC (CRC-16) format if 8-bit characters are used or a longitudinal check (a sum check) in combination with vertical redundancy check (parity) if 7-bit characters are used. In transparent operation, to allow the BISYNC control characters to be present in the frame as valid text data, a special character (DLE) is defined, which informs the receiver that the character following the DLE is a text character, not a control character (from the control character table). If a DLE is transmitted as valid data, it must be preceded by a DLE character. This procedure is sometimes called byte-stuffing.

The physical layer of the BISYNC communications link must provide a means of synchronizing the receiver and transmitter, which is usually accomplished by sending at least one pair of synchronization characters prior to every frame.

BISYNC has the unusual property that a transmit underrun need not be an error. If an underrun occurs, the synchronization pattern is transmitted until data is once again ready to transmit. The receiver discards the additional synchronization characters as they are received, provided the V bit is set in the BISYNC-BISYNC SYNC register. In non-transparent operation, all synchronization characters (SYNCs) are discarded if the V bit is set in the BISYNC-BISYNC SYNC register. In transparent operation, all DLE-SYNC pairs are discarded. (Note that correct operation in this case assumes that, on the transmit side, the underrun does not occur between the DLE and its following character, a failure mode prevented in the MC68302.)

By appropriately setting the SCC mode register, any of the SCC channels may be configured to function as a BISYNC controller. The BISYNC controller handles the basic functions of the BISYNC protocol in normal mode and in transparent mode.

The SCC in BISYNC mode can work with IDL, GCI (IOM2), PCM highway, or NMSI interfaces. When the SCC in BISYNC mode is used with a modem interface (NMSI), the SCC outputs are connected directly to the external pins. The modem interface uses seven dedicated pins: transmit data (TXD), receive data (RXD) receive clock (RCLK), transmit clock (TCLK), carrier detect ($\overline{CD}$), clear to send ($\overline{CTS}$), and request to send ($\overline{RTS}$). Other modem lines can be supported using the parallel I/O pins.

The BISYNC controller consists of separate transmit and receive sections whose operations are asynchronous with the M68000 core and may be either synchronous or asynchronous with respect to the other SCCs. Each clock can be supplied from either the internal baud

Memory Map for the placement of the three SCC parameter RAM areas and Table 4-5 for the other parameter RAM values.

Table 4-9. BISYNC Specific Parameter RAM

Address	Name	Width	Description
SCC Base + 9C SCC Base + 9E SCC Base + A0 # SCC Base + A2 SCC Base + A4 # SCC Base + A6 SCC Base + A8	RCRC CRCC PRCRC TCRC PTCRC RES RES	Word Word Word Word Word Word Word	Temp Receive CRC CRC Constant Preset Receiver CRC 16/LRC Temp Transmit CRC Preset Transmitter CRC 16/LRC Reserved Reserved
SCC Base + AA # SCC Base + AC # SCC Base + AE #	PAREC BSYNC BDLE	Word Word Word	Receive Parity Error Counter BISYNC SYNC Character BISYNC DLE Character
SCC Base + B0 # SCC Base + B2 # SCC Base + B4 # SCC Base + B6 # SCC Base + B8 # SCC Base + BA # SCC Base + BC # SCC Base + BE #	CHARACTER1 CHARACTER2 CHARACTER3 CHARACTER4 CHARACTER5 CHARACTER6 CHARACTER7 CHARACTER8	Word Word Word Word Word Word Word Word	CONTROL Character 1 CONTROL Character 2 CONTROL Character 3 CONTROL Character 4 CONTROL Character 5 CONTROL Character 6 CONTROL Character 7 CONTROL Character 8

Initialized by the user (M68000 core).

The M68000 core configures each SCC to operate in one of four protocols by the MODE1–MODE0 bits in the SCC mode register. MODE1–MODE0 = 11 selects the BISYNC mode of operation. The SYN1–SYN2 synchronization characters are programmed in the data synchronization register (see 4.5.4 SCC Data Synchronization Register (DSR)).

The BISYNC controller uses the same basic data structure as the other protocol controllers. Receive and transmit errors are reported through their respective BDs. The status of the line is reflected in the SCC status register, and a maskable interrupt is generated upon each status change.

There are two basic ways of handling the BISYNC channels. First, data may be inspected on a per-byte basis, with the BISYNC controller interrupting the M68000 core upon receipt of every byte of data. Second, the BISYNC controller may be operated so that software is only necessary for handling the first two to three bytes of data; subsequent data (until the end of the block) can be handled by the BISYNC controller without interrupting the M68000 core. See 4.5.13.14 Programming the BISYNC Controllers for more information.

4.5.13.4 BISYNC Command Set

The following commands are issued to the command register.

STOP TRANSMIT Command

After a hardware or software reset and the enabling of the channel using the SCC mode register, the channel is in the transmit enable mode and starts polling the first BD in the table approximately every eight transmit clocks.

The STOP TRANSMIT command aborts transmission after the contents of the FIFO are transmitted (up to three bytes) without waiting until the end of the buffer is reached. The TBD# is not advanced. SYNC characters consisting of SYNC-SYNC or DLE-SYNC pairs (according to the transmitter mode) will be continually transmitted until transmission is re-enabled by issuing the RESTART TRANSMIT command. The STOP TRANSMIT com-

mand may be used when it is necessary to abort transmission and transmit an EOT control sequence. The EOT sequence should be the first buffer presented to the BISYNC controller for transmission after re-enabling transmission.

The STOP TRANSMIT command must be issued before the SCC mode register is used to disable the transmitter if the transmitter is to be re-enabled at a later time.

NOTE

The BISYNC controller will remain in the transparent or normal mode after receiving the STOP TRANSMIT or RESTART TRANSMIT commands.

RESTART TRANSMIT Command

The RESTART TRANSMIT command is used to begin or resume transmission from the current Tx BD number (TBD#) in the channel's Tx BD table. When this command is received by the channel, it will start polling the ready bit in this BD. This command is expected by the BISYNC controller after a STOP TRANSMIT command, after the STOP TRANSMIT command and the disabling of the channel in its mode register, or after a transmitter error (underrun or CTS lost) occurs.

If the transmitter is being re-enabled, the RESTART TRANSMIT command must be used and should be followed by the enabling of the transmitter in the SCC mode register.

RESET BCS CALCULATION Command

The RESET BCS CALCULATION command resets the receive BCS accumulator immediately. For example, it may be used to reset the BCS after recognizing a control character, signifying that a new block is commencing (such as SOH).

ENTER HUNT MODE Command

After a hardware or software reset and the enabling of the channel in the SCC mode register, the channel is in the receive enable mode and will use the first BD in the table.

The ENTER HUNT MODE command is used to force the BISYNC controller to abort reception of the current block, generate an RX interrupt (if enabled) as the buffer is closed, and enter the hunt mode. In hunt mode, the BISYNC controller continually scans the input data stream for the SYN1–SYN2 sequence as programmed in the data synchronization register. After receiving the command, the current receive buffer is closed, and the BCS is reset. Message reception continues using the next BD.

If an enabled receiver has been disabled (by clearing ENR in the SCC mode register), the ENTER HUNT MODE command must be given to the channel before setting ENR again.

4.5.13.5 BISYNC Control Character Recognition

The BISYNC controller can recognize special control characters. These characters are used to “customize” the BISYNC protocol implemented by the BISYNC controller and may be used to aid its operation in a DMA-oriented environment. Their main use is for receive buffers longer than one byte. In single-byte buffers, each byte can easily be inspected, and control character recognition should be disabled.

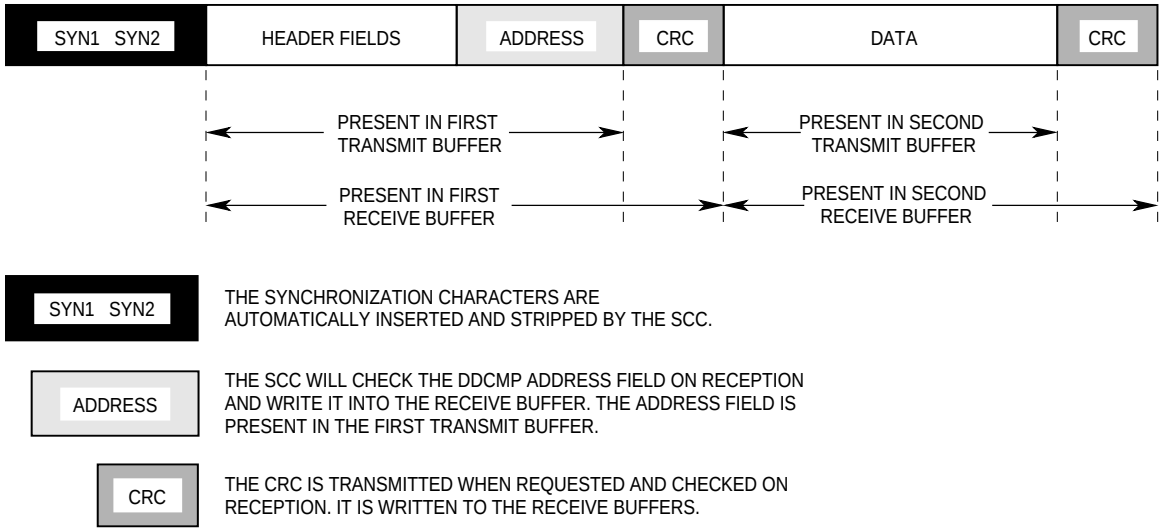


Figure 4-35. DDCMP Transmission/Reception Summary

When a BD has been completely transmitted, the transmit CRC (TC) bit is checked in the BD. If set, the DDCMP controller appends one of the block checks: CRC1, CRC2, or CRC3 for the header field, data message, or control messages, respectively. Next, the DDCMP controller writes the buffer's status bits into the BD and clears the ready bit in the BD. It then proceeds to the next BD in the table. When the last bit (L) is set and the TC bit is set in that BD, the DDCMP controller appends the CRC2 block check to the data field. This bit is also used for transmitting CRC3 in control messages. Next, it writes the buffer status bits into the BD and clears the ready bit. Finally, on synchronous links, either SYN1–SYN2 pairs or IDLEs (as programmed in the DDCMP mode register) are transmitted. When the end of the current BD has been reached and the last bit is not set (working in multibuffer mode or sending back-to-back messages), only the status bits are written. In either case, when a BD has been completely transmitted, an interrupt is issued if the interrupt (I) bit in the BD is set and the event is not masked in the DDCMP mask register. The appropriate setting of the I bit in each BD allows the user to be interrupted after transmission of each buffer, a specific buffer, or each message.

4.5.14.2 DDCMP Channel Frame Reception Processing.

The DDCMP receiver is also designed to work with almost no intervention from the M68000 core (see Figure 4-35).

The DDCMP receiver performs automatic SYN1–SYN2 synchronization on synchronous links and start/stop synchronization on asynchronous links. Automatic message synchronization is achieved by searching for the special starting characters SOH, ENQ, or DLE and making address comparisons with a mask. When the M68000 core enables the DDCMP receiver on synchronous links, it enters hunt mode. In this mode, as data is shifted into the receiver shift register one bit at a time, the contents of the register are compared to the SYN1–SYN2 fields of the data synchronization register (see 4.5.4 SCC Data Synchronization Register (DSR)). If the two are not equal, the next bit is shifted in, and the comparison is repeated. When the registers match, hunt mode is terminated, and character assembly

DSOH—DDCMP SOH Register

The 8-bit DSOH register is used to synchronize data messages by the DDCMP controller. When the DDCMP controller is not in hunt mode (byte synchronization is now established), it searches for the SOH character to start processing data messages. The DDCMP controller transfers the header and the data fields of the message to the buffer, checks the header and data CRCs, counts the data field up to the value contained in the header byte count field, and compares the header address field against the user-defined addresses. The DSOH register is a memory-mapped read-write register.

DENQ—DDCMP ENQ Register

The 8-bit DENQ register is used to synchronize control messages by the DDCMP controller. When the DDCMP controller is not in hunt mode (byte synchronization is established), it searches for the ENQ character to start processing control messages. The DDCMP controller transfers the message to the buffer, checks the CRC, and compares the message address field against the user-defined addresses. The DENQ register is a memory-mapped read-write register.

DDLE—DDCMP DLE Register

The 8-bit DDLE register is used to synchronize maintenance messages by the DDCMP controller. When the DDCMP controller is not in hunt mode (byte synchronization is established), it searches for the DLE character to start processing the maintenance messages. The DDCMP controller transfers the header and the data fields of the message to the buffer, checks the header and data CRCs, counts the data field up to the value contained in the header byte count field, and compares the header address field against the user-defined addresses. The DDLE register is a memory-mapped read-write register.

4.5.14.7 DDCMP Address Recognition.

Each DDCMP controller has five 16-bit registers to support address recognition: one mask register and four address registers (DMASK, DADDR1, DADDR2, DADDR3, and DADDR4). The DDCMP controller reads the message address from the receiver, masks it with the user-defined DMASK bits, and then checks the result against the four address register values. A one in DMASK indicates a bit position where a comparison should take place; a zero masks the comparison. For 8-bit address comparison, the high byte of DMASK should be zero.

4.5.14.8 DDCMP Error-Handling Procedure

The DDCMP controller reports message reception and transmission errors using the channel BDs, the error counters, and the DDCMP event register. The modem interface lines can also be directly monitored with the SCC status register.

Transmission errors:

1. Transmitter Underrun. When this error occurs, the channel terminates buffer transmission, closes the buffer, sets the underrun (UN) bit in the BD, and generates the transmit error (TXE) interrupt (if enabled). The channel will resume transmission after the reception of the RESTART TRANSMIT command. The FIFO size is three bytes.

The channel will enter hunt mode immediately. It is possible that a SYN1–SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this “header” will have a CRC error.

NOTE

This error can occur only on asynchronous links.

6. Parity Error. When a parity error occurs, the channel writes the received character to the buffer, closes the buffer, sets the parity error (PR) bit in the BD, and generates the RBK interrupt (if enabled).
   The channel will enter hunt mode immediately. It is possible that a SYN1–SYN2-(SOH,DLE,ENQ) sequence in data will be incorrectly interpreted as the start of the next header, but this “header” will have a CRC error.

NOTE

This error can occur only on asynchronous links.

Error Counters

The CP maintains four 16-bit (modulo 2**16) error counters for each DDCMP controller. They can be initialized by the user when the channel is disabled. The counters are as follows:

—CRC1EC—CRC1 Error Counter
—CRC2EC—CRC2/CRC3 Error Counter
—NMARC — Nonmatching Address Received Counter (updated only when the frame is error-free)
—DISMC — Discarded Messages (received messages when there are no free buffers and the frame is error-free)

4.5.14.9 DDCMP Mode Register

Each SCC mode register is a 16-bit, memory- mapped, read-write register that controls the SCC operation. The term DDCMP mode register refers to the protocol-specific bits (15–6) of the SCC mode register when that SCC is configured for DDCMP. The read-write DDCMP mode register is cleared by reset.

15	14	13	12	11	10	9	8	7	6	5 0
NOS3	NOS2	NOS1	NOS0	—	V.110	—	—	SYNF	ENC	COMMON SCC MODE BITS

NOS3–NOS0—Minimum Number of SYN1—SYN2 Pairs between or before Messages (1 to 16 SYNC Pairs)

If NOS3–NOS0 = 0000, then 1 SYNC pair will be transmitted; if NOS3–NOS0 = 1111, then 16 SYNC pairs will be transmitted.

NOTE

With appropriate programming of the transmit BD (TC = 1 and L = 0), it is possible to transmit back-to-back messages.

Each module operates completely independent of its environment. Independence from the hardware environment is achieved by the fact that each module is individually configurable and relocatable anywhere in system memory. Calls are used for requesting resources or services from the host operating system (memory management and message passing), which creates independence from the firmware. Modules may be used with any combination of other modules, including those added by the user (see Figure B-1).

The chip driver module features are as follows:

- All software modules use the driver module as their interface to the MC68302
- Illustrates initialization of the MC68302 and interrupt handling
- Provides complete configuration of the MC68302 on system start (or restart)
- Provides services for the MC68302 serial ports

  —- Supports linked lists of frames for both transmit and receive
  — Provides error handling and recovery for both transmit and receive
  — Supports all three serial ports in different protocol configurations
  — Provides internal loopback (frames not sent to MC68302)
- Provides timer services
- Can be configured to send “trace” messages to a system log file
- Supports up to 24 serial ports

in the buffer or when certain events, such as an error or an end-of-frame, are detected. Whatever the reason, the buffer is then said to be closed, and additional data will be stored using the next BD. Whenever the CP needs to begin using a BD because new data is arriving, it will check the empty bit of that BD. If the current BD is not empty, it will report a busy error. However, the CP will not move from the current BD until it becomes empty. When the CP sees the “wrap” bit set in a BD, the CP goes back to the beginning of the BD table after use of this BD is complete. After using a BD, the CP sets its empty bit to not-empty; thus, the CP will never use a BD twice. The BD must be processed by the M68000 core before being used again.

D.3.3 New Pointers

To control the buffering of the SCCs, the three BD pointers to be used by software are defined. Two pointers are used for the transmit BDs, and one pointer is used for the receive BDs (see Figure D-5).

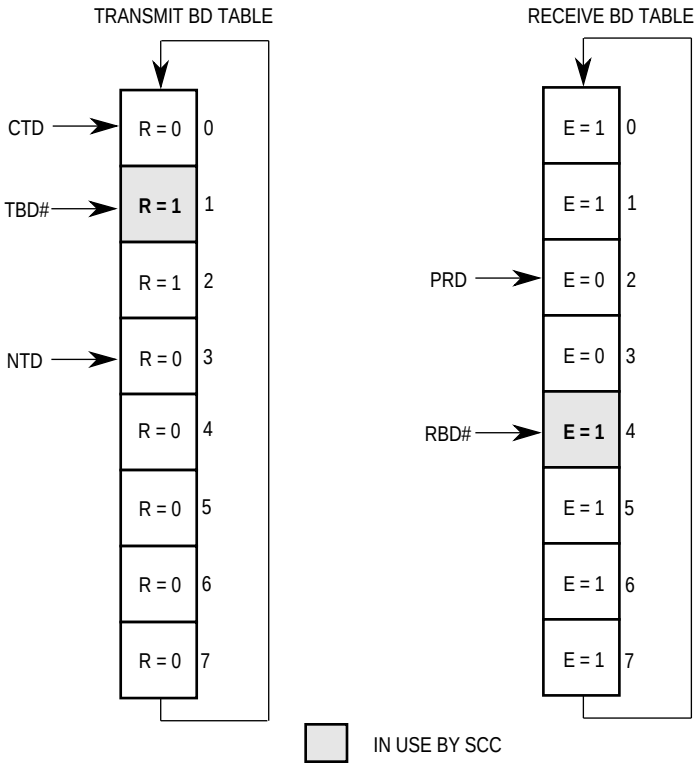


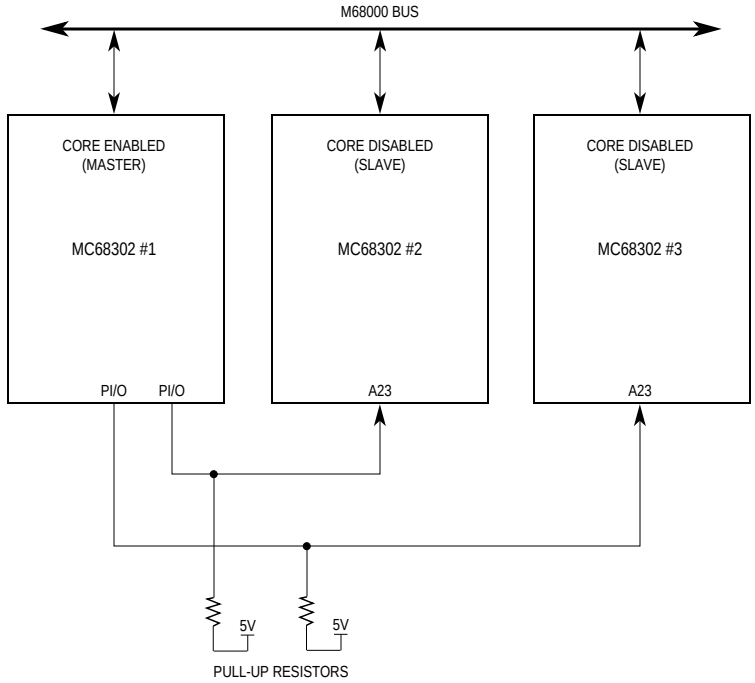
Figure D-5. Pointer during Execution

New transmit data (NTD) shows the next transmit BD that will be receiving data. This is the first pointer to move in the transmit process.

Confirm transmit data (CTD) shows the next transmit BD that will be confirmed. To confirm a buffer, check for errors after transmission and then mark the BD as available for USQ. This is the last pointer to move in the transmit process.

fully functioning MC68302s, each having an isolated bus and the ability to send data and messages between them (e.g., through a shared RAM). However, another approach is possible.

By using the MC68302 “disable CPU logic” feature, enabled with the DISCPU pin, the MC68302 can be converted into an intelligent slave peripheral that no longer has its M68000 core operating. The SDMA channels and IDMA channel request the bus externally through the bus request ($\overline{BR}$) pin. (When not in slave mode, these channels request the bus internally to the on-chip bus arbiter, with no external indication visible.) A typical slave mode example is shown in Figure D-19. A single master MC68302 (i.e., one with the M68000 core enabled) can access and control one or more slave MC68302s. (i.e., ones with the M68000 core disabled.)



NOTE: A23 is not used by the slaves.

Figure D-19. Typical Slave Mode Example

Use of the “disable CPU logic” feature in a multiple MC68302 system depends mainly on the amount of protocol processing required by the M68000 core. If the data rates are high and the amount of protocol processing required on each channel is significant, the M68000 core may be the limiting factor in communications performance. Thus, further increases in serial rates will not yield additional packets/sec performance. In such a case, a faster processor (such as the MC68020/MC68030) could be used to control all three MC68302 devices in slave mode.

The bus utilization of the SDMA channels on the three MC68302 devices is not usually a significant factor. For instance, if three SCC channels are running full duplex at 64 kbps, the respective SDMA channels consume less than 1 percent of the M68000 bus. You can calculate this figure for your design by determining how often a bus cycle to memory is required

D.8.5 Transparent Mode with the NMSI Physical Interface

NMSI has two independent data signals, TXD and RXD, and two independent clocking signals, TCLK and RCLK. TCLK and RCLK may be individually chosen to be generated internally or externally to the MC68302.

NMSI also has three control signals: $\overline{\text{RTS}}$, $\overline{\text{CTS}}$, and $\overline{\text{CD}}$. First, let's discuss their properties in general. The SCC forces $\overline{\text{RTS}}$ low when it is ready to transmit data, but the SCC waits until it sees $\overline{\text{CTS}}$ is low before doing this. After the frame has been transmitted, the $\overline{\text{RTS}}$ signal is negated (high). The $\overline{\text{CTS}}$ signal should stay low during the entire time $\overline{\text{RTS}}$ is low, or transmission is aborted and a $\overline{\text{CTS}}$ lost error is indicated in the transmit buffer descriptor (Tx BD). On the receiving side, the CD signal going low tells the MC68302 to gate data into this SCC. Once low, $\overline{\text{CD}}$ should remain low for the entire frame, or reception is terminated and a $\overline{\text{CD}}$ lost error is signaled in the receive buffer descriptor (Rx BD).

Sometimes the $\overline{\text{CTS}}$ and $\overline{\text{CD}}$ input functions described above are not appropriate for an application. In this case, the software operation mode in the SCC mode register (SCM) can be chosen by programming the DIAG1-DIAG0 bits. In the software operation mode, as far as the SCC is concerned, $\overline{\text{CTS}}$ and $\overline{\text{CD}}$ are always low. However, the real value of the $\overline{\text{CTS}}$ and $\overline{\text{CD}}$ lines externally can be read in the SCCS register once the transmitter and receiver are enabled, and changes in these lines can generate interrupts via the SCCE register. Software operation mode does not affect $\overline{\text{RTS}}$ because, since $\overline{\text{RTS}}$ is an output, $\overline{\text{RTS}}$ can always be ignored by the external logic.

In totally transparent mode (and also BISYNC mode), the $\overline{\text{CD}}$ signal can become a synchronization input. When discussing the $\overline{\text{CD}}$ signal during totally transparent mode in this document, $\overline{\text{CD}}$ will be referred to as "$\overline{\text{CD}}$ (sync)". The totally transparent mode is initiated by setting the EXSYN bit in the SCM. With EXSYN set, a high-to-low transition on $\overline{\text{CD}}$ (sync) defines the start of *both transmission and reception* of transparent mode frames. Subsequent high and low transitions of $\overline{\text{CD}}$ (sync) have *no* effect on the reception of data. The only way to reinitiate the SYNC process is to issue an ENTER HUNT MODE command to the channel, and then force another high-to-low transition on $\overline{\text{CD}}$ (sync).

Figure D-23 shows the simplest NMSI transmit case. NTSYN and EXSYN are set to enable transparent mode, and the L bit is set. Software operation mode (DIAG1 = 1 and DIAG0 = 1) is chosen to eliminate using $\overline{\text{CTS}}$ to control transmission. However, since EXSYN = 1, $\overline{\text{CD}}$ becomes $\overline{\text{CD}}$ (sync), and transmission cannot begin until $\overline{\text{CD}}$ (sync) is low (which can be accomplished by grounding $\overline{\text{CD}}$ (sync)). Thus, there will only be a 1-bit delay between $\overline{\text{RTS}}$ being asserted by the transmitter and actual data being transmitted. Since the L bit is set, $\overline{\text{RTS}}$ is negated after the last byte in the frame.

In the preceding example, if multiple buffers had been ready with their L bits cleared, $\overline{\text{RTS}}$ would have remained asserted, and the next buffer's data would have begun immediately. If multiple buffers had been ready with their L bits set, $\overline{\text{RTS}}$ would have been asserted again after a delay of at least 17 idle bits on the line (the exact number of bits is load dependent).

The PCM highway interface has three $\overline{RTS}$ signals. One of these signals is asserted when an SCC wants to transmit over the PCM Highway just like in NMSI mode), and stays continuously asserted until the entire frame is transmitted (regardless of how many time slots the transmission takes). Which $\overline{RTS}$ signal asserts depends on which SCC is transmitting; there is one $\overline{RTS}$ signal for each SCC. Notice, however, that there is no $\overline{CTS}$ signal, so there is nothing to hold off the transfer. If the $\overline{RTS}$ signals are not needed, they can be ignored or reassigned as parallel I/O lines.

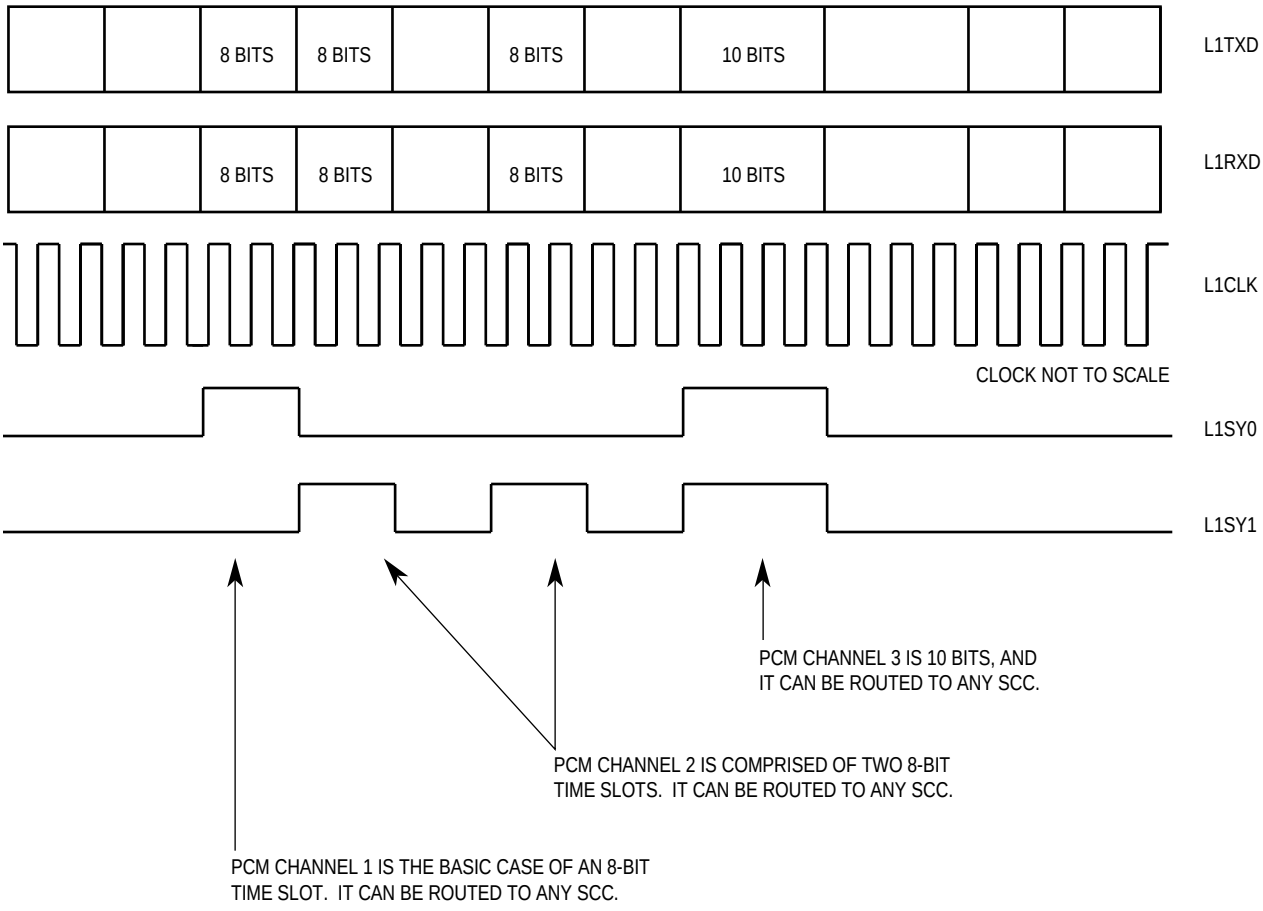


Figure D-29. Routing Channels in PCM Envelope Mode

What other signals are missing from PCM mode? First, there is no $\overline{CD}$ signal for the receiver. The receiver is enabled whenever the ENR bit is set. However, you could say there is a $\overline{CD}$ (sync) of sorts that is implemented with the L1SY1 and L1SY0 pins. Two pins are used since not only is the timing important, but also the selection of the PCM channel as well.

The way transparent mode works with a PCM highway interface is very similar to the operation of the gated clocks example discussed previously. Whether or not a time slot environment is present, PCM mode gives greater control over what intervals transparent data can be transmitted and received. However, in PCM mode, the clocks are gated by the physical interface on the MC68302 as opposed to external hardware.