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	-
Number of Cores/Bus Width	-
Speed	-
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	-
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	-
Operating Temperature	-
Security Features	-
Package / Case	-
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68302ag16c

Home Page:
www.freescale.com
email:
support@freescale.com
USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com
Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com
Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com
Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com
For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Once the processor is in the user state and executing instructions, only exception processing can change the privilege state. During exception processing, the current state of the S bit in the SR is saved and the S bit is asserted, putting the processor in the supervisor state. Therefore, when instruction execution resumes at the address specified to process the exception, the processor is in the supervisor privilege state. The transition from the supervisor to user state can be accomplished by any of four instructions: return from exception (RTE), move to status register (MOVE to SR), AND immediate to status register (ANDI to SR), and exclusive OR immediate to status register (EORI to SR).

2.4 EXCEPTION PROCESSING

The processing of an exception occurs in four steps, with variations for different exception causes. During the first step, a temporary copy of the SR is made, and the SR is set for exception processing. During the second step, the exception vector is determined; during the third step, the current processor context is saved. During the fourth step, a new context is obtained, and the processor switches to instruction processing.

2.4.1 Exception Vectors

Exception vectors are memory locations from which the processor fetches the address of a routine to handle that exception. All exception vectors are two words long except for the reset vector, which is four words. All exception vectors lie in the supervisor data space except for the reset vector, which is in the supervisor program space. A vector number is an 8-bit number which, when multiplied by four, gives the offset of the exception vector. Vector numbers are generated internally or externally, depending on the cause of the exception. In the case of interrupts, during the interrupt acknowledge bus cycle, a peripheral may provide an 8-bit vector number to the processor on data bus lines D7–D0. Alternatively, the peripheral may assert autovector ($\overline{\text{AVEC}}$) instead of data transfer acknowledge ($\overline{\text{DTACK}}$) to request an autovector for that priority level of interrupt. The exception vector assignments for the M68000 processor are shown in Table 2-5.

Table 2-5. M68000 Exception Vector Assignment

Vector Number	Decimal	Address Hex	Space	Assignment
0	0	000	SP	Reset: Initial SSP[2]
1	4	004	SP	Reset: Initial PC[2]
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator

Table 3-9. SCR Register Bits

Bit	Name	Section(s)
IPA	Interrupt Priority Active	3.8.2
HWT	Hardware Watchdog Timeout	3.8.2, 3.8.6
WPV	Write Protect Violation	3.8.2
ADC	Address Decode Conflict	3.8.2
ERRE	External RISC Request Enable	3.9
VGE	Vector Generation Enable	3.8.4
WPVE	Write Protect Violation Enable	3.8.3
RMCST	Read-Modify-Write Cycle Special Treatment	3.8.3
EMWS	External Master Wait State	3.8.3, 3.8.4
ADCE	Address Decode Conflict Enable	3.8.3
BCLM	Bus Clear Mask	3.8.2, 3.8.3, 3.8.5
FRZW	Freeze Watchdog Timer Enable	3.8.8
FRZ1	Freeze Timer 1 Enable	3.8.8
FRZ2	Freeze Timer 2 Enable	3.8.8
SAM	Synchronous Access Mode	3.8.3, 3.8.4
HWDEN	Hardware Watchdog Enable	3.8.6
HWDCN	Hardware Watchdog Count	3.8.6
LPREC	Low-Power Recovery	3.8.7
LPP16	Low-Power Clock Prescale Divide by 16	3.8.7
LPEN	Low-Power Enable	3.8.7
LPCD	Low-Power Clock Divider Selects	3.8.7

3.8.2 System Status Bits

The eight most significant bits of the SCR are used to report events recognized by the system control logic. On recognition of an event, this logic sets the corresponding bit in the SCR. The bits may be read at any time. A bit is reset by one and is left unchanged by zero. More than one bit may be reset at a time.

After system reset (simultaneous assertion of $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$), these bits are cleared.

IPA—Interrupt Priority Active

This bit is set when the M68000 core has an unmasked interrupt request. When bus clear mask (BCLM) is set, $\overline{\text{BCLR}}$ and the internal bus clear to the IDMA are asserted.

NOTE

If BCLM is set, an interrupt handler will normally clear IPA at the end of the interrupt routine to allow an alternate bus master to regain the bus; however, if BCLM is cleared, no additional action need be taken in the interrupt handler.

In the case of nested interrupts, the user may wish to clear the IPA bit only at the end of the original lower priority interrupt routine to keep $\overline{\text{BCLR}}$ asserted until it completes. To guarantee that

tools are 1) the ready bit in the transmit buffer descriptor, 2) the ENT bit, 3) the STOP TRANSMIT command, 4) the RESTART TRANSMIT command, and 5) the FRZ bit in the SCM (UART mode only).

ENR—Enable Receiver

When ENR is set, the receiver is enabled. When it is cleared, the receiver is disabled, and any data in the receive FIFO is lost. If ENR is cleared during data reception, the receiver aborts the current character. ENR may be set or cleared regardless of whether serial clocks are present. To restart reception, the ENTER HUNT MODE command should be issued before ENR is set again.

ENT—Enable Transmitter

When ENT is set, the transmitter is enabled; when ENT is cleared, the transmitter is disabled. If ENT is cleared, the transmitter will abort any data transmission, clear the transmit data FIFO and shift register, and force the TXD line high (idle). Data already in the transmit shift register will not be transmitted. ENT may be set or cleared regardless of whether serial clocks are present.

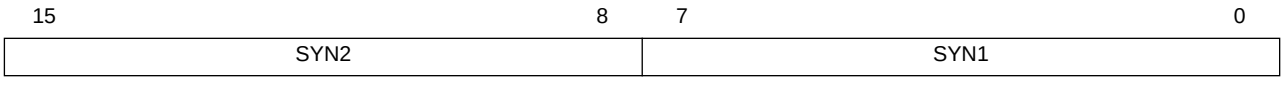
The STOP TRANSMIT command additionally aborts the current frame and would normally be given to the channel before clearing ENT. The command does not clear ENT automatically. In a similar manner, to restart transmission, the user should issue the RESTART TRANSMIT command and then set ENT. The command register is described in 4.3 Command Set. The specific actions taken with each command vary somewhat according to protocol and are discussed in each protocol section.

MODE1–MODE0—Channel Mode

00 = HDLC
01 = Asynchronous (UART and DDCMP)
10 = Synchronous DDCMP and V.110
11 = BISYNC and Promiscuous Transparent

4.5.4 SCC Data Synchronization Register (DSR)

Each DSR is a 16-bit, memory-mapped, read-write register. DSR specifies the pattern used in the frame synchronization procedure of the SCC in the synchronous protocols. In the UART protocol it is used to configure fractional stop bit transmission. After reset, the DSR defaults to $7E7E (two FLAGs); thus, no additional programming is necessary for the HDLC protocol. For BISYNC, DDCMP, and V.110, the contents of the DSR should be written before the channel is enabled. Note that for the DDCMP, SYN1 must equal SYN2 must equal DSYN1 for proper operation.



NOTE

The DSR register has no relationship to the RS-232 signal “data set ready,” which is also abbreviated DSR.

4.5.6.2 Maximum Receive Buffer Length Register (MRBLR)

Each SCC has one MRBLR that is used to define the receive buffer length for that SCC. The MRBLR defines the maximum number of bytes that the IMP will write to a receive buffer on that SCC before moving to the next buffer. The IMP may write fewer bytes to the buffer than MRBLR if a condition such as an error or end of frame occurs, but it will never write more bytes than the MRBLR value. Thus, buffers supplied by the user for use by the IMP should always be of size MRBLR (or greater) in length.

The transmit buffers for an SCC are not affected in any way by the value programmed into MRBLR. Transmit buffers may be individually chosen to have varying lengths, as needed. The number of bytes to be transmitted is chosen by programming the data length field in the Tx BD.

NOTE

MRBLR was not intended to be changed dynamically while an SCC is operating. However, if it is modified in a single bus cycle with one 16-bit move (NOT two 8-bit back-to-back bus cycles), then a dynamic change in receive buffer length can be successfully achieved, which occurs when the CP moves control to the next Rx BD in the table. Thus, a change to MRBLR will not have an immediate effect. To guarantee the exact Rx BD on which the change will occur, the user should change MRBLR only while the SCC receiver is disabled (see 4.5.6 SCC Parameter RAM Memory Map).

NOTE

The MRBLR value should be greater than zero in all modes. In the HDLC and transparent modes, the MRBLR should have an even value.

4.5.6.3 Receiver Buffer Descriptor Number (RBD#)

The RBD# for each SCC channel defines the next BD to which the receiver will move data when it is in the IDLE state or defines the current BD during frame processing. The RBD# is the BD offset from the SCC base in the Rx BD table. For Rx BD 0, RBD# = $00; for Rx BD 1, RBD# = $08, etc. Upon reset, the CP main controller sets this register to zero. The user can change this register only after the ENR bit is clear and after the ENTER HUNT MODE command has been issued. In most applications, this parameter will never need to be modified by the user.

4.5.6.4 Transmit Buffer Descriptor Number (TBD#)

The TBD# for each SCC channel defines the next BD from which the transmitter will move data when it is in the IDLE state or defines the current BD during frame transmission. The TBD# is the BD offset from the SCC base in the Tx BD table. For Tx BD 0, TBD# = $40; for Tx BD 1, TBD# = $48, etc. Upon reset, the CP main controller sets this register to $40. The user can change this register only after the STOP TRANSMIT command has been issued. In most applications, this parameter will never need to be modified by the user.

I—Interrupt

0 = No interrupt is generated after this buffer has been filled.
1 = The RX bit in the UART event register will be set when this buffer has been completely filled by the CP, indicating the need for the M68000 core to process the buffer. The RX bit can cause an interrupt.

The following bits contain status information written by the CP after it has finished receiving data in the associated data buffer.

C—Control Character

0 = This buffer does not contain a control character.
1 = This buffer contains a user-defined control character in the last byte location.

A—Address

0 = The buffer contains data only.
1 = When working in nonautomatic multidrop mode (UM1–UM0 = 01), this bit indicates that the first byte of this buffer contains an address byte. The address comparison should be implemented in software. In automatic multidrop mode, this bit indicates that the BD contains a message received immediately following an address recognized in UADDR1 or UADDR2. This address is not written into the receive buffer.

M—Address Match

This bit is meaningful only if the A bit (bit 10) is set and UM1–UM0 = 11 in the UART mode register. Following an address match, this bit defines which address character matched the user-defined address character, enabling the UART to receive the data.

0 = The address-matched user-defined UADDR2
1 = The address-matched user-defined UADDR1

ID—Buffer Closed on Reception of Idles

The buffer was closed due to the reception of the programmable number of consecutive IDLE sequences (defined in MAX_IDL).

Bits 7–6, 2—Reserved for future use.

BR—Break Received

A break sequence was received while receiving data into this buffer.

FR—Framing Error

A character with a framing error was received and is located in the last byte of this buffer. A framing error is detected by the UART controller when no stop bit is detected in the receive data string.

PR—Parity Error

A character with a parity error was received and is located in the last byte of this buffer.

OV—Overrun

A receiver overrun occurred during message reception.

ceeds the length of the data buffer, the HDLC controller will fetch the next BD in the table and, if it is empty, will continue to transfer the rest of the frame to this BD's associated data buffer.

During this process, the HDLC controller will check for a frame that is too long. When the frame ends, the CRC field is checked against the recalculated value and is written to the data buffer starting with the first address byte. The data length written to the last BD in the HDLC frame is the length of the entire frame. This enables HDLC protocols that “lose” frames to correctly recognize the frame-too-long condition. The HDLC controller then sets the last buffer in frame bit, writes the frame status bits into the BD, and clears the empty bit. The HDLC controller next generates a maskable interrupt, indicating that a frame has been received and is in memory. The HDLC controller then waits for a new frame. Back-to-back frames may be received with only a single shared flag between frames. Also, flags that share a zero will be recognized as two consecutive flags.

4.5.12.3 HDLC Memory Map

When configured to operate in HDLC mode, the IMP overlays the structure shown in Table 4-7 onto the protocol-specific area of that SCC parameter RAM. Refer to 2.8 MC68302 Memory Map for the placement of the three SCC parameter RAM areas and to Table 4-2 for the other parameter RAM values.

Table 4-8. HDLC-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	RCRC_L RCRC_H C_MASK_L C_MASK_H TCRC_L TCRC_H	Word Word Word Word Word Word	Temp Receive CRC Low Temp Receive CRC High Constant ($F0B8 16-Bit CRC, $DEBB 32-Bit CRC) Constant ($XXXX 16-Bit CRC, $20E3 32-Bit CRC) Temp Transmit CRC Low Temp Transmit CRC High
SCC Base + A8 # SCC Base + AA # SCC Base + AC # SCC Base + AE # SCC Base + B0 #	DISFC CRCEC ABTSC NMARC RETRC	Word Word Word Word Word	Discard Frame Counter CRC Error Counter Abort Sequence Counter Nonmatching Address Received Counter Frame Retransmission Counter
SCC Base + B2 # SCC Base + B4	MFLR MAX_cnt	Word Word	Max Frame Length Register Max_Length Counter
SCC Base + B6 # SCC Base + B8 # SCC Base + BA # SCC Base + BC # SCC Base + BE #	HMASK HADDR1 HADDR2 HADDR3 HADDR4	Word Word Word Word Word	User-Defined Frame Address Mask User-Defined Frame Address User-Defined Frame Address User-Defined Frame Address User-Defined Frame Address

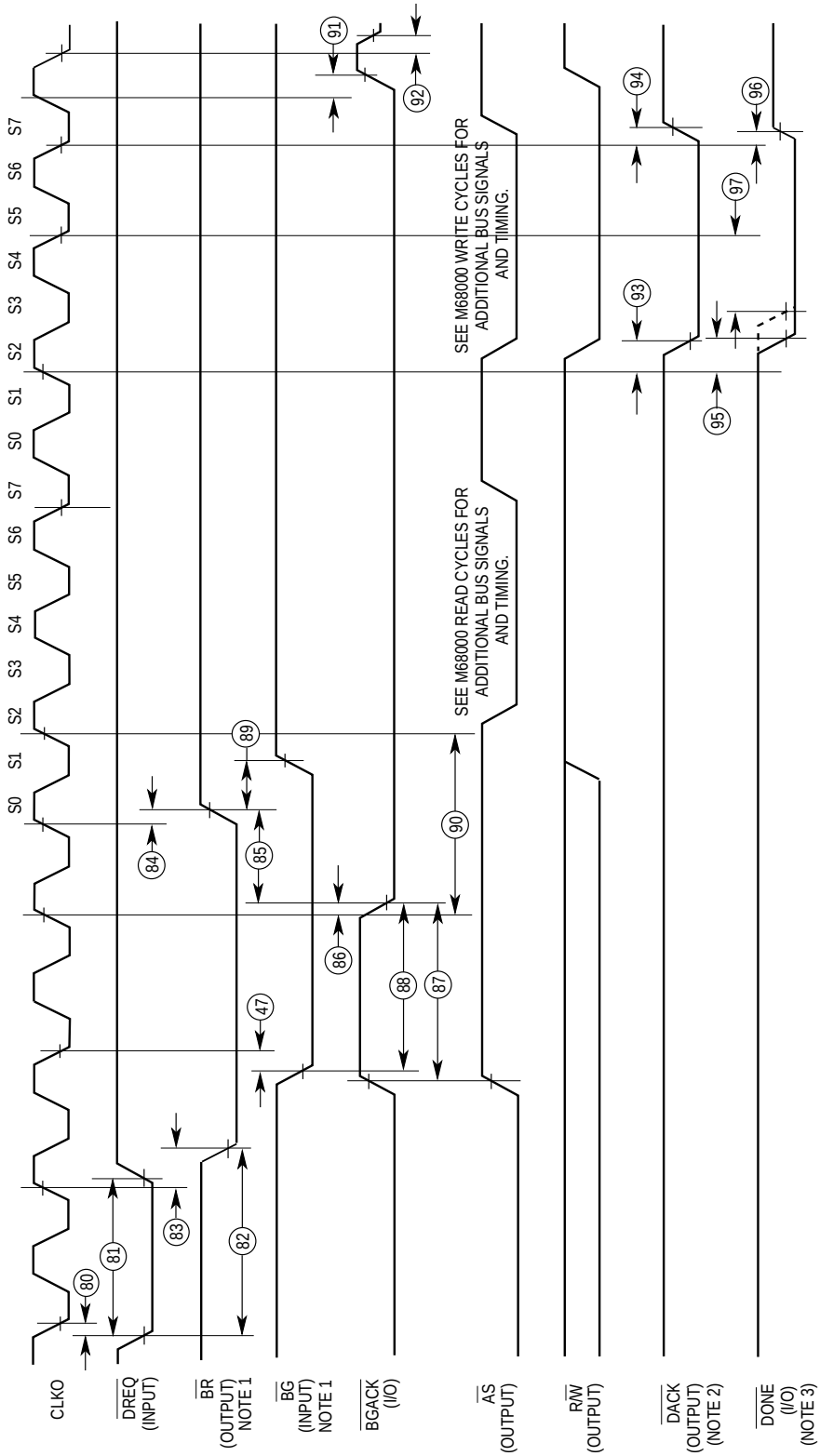
Initialized by the user (M68000 core).

NOTE

An incorrect initialization of C_MASK may be used to “force” receive CRC errors for software testing purposes. The transmit CRC will not be affected.

4.5.12.4 HDLC Programming Model

The M68000 core configures each SCC to operate in one of four protocols by the MODE1–MODE0 bits in the SCC mode register (SCM). MODE1–MODE0 = 00 selects HDLC mode.



NOTES:
1. BR and BG shown above are only active in disable CPU mode; otherwise, they do not apply to the diagram.
2. Assumes the ECO bit in the CMR = 1.
3. For the case when DONE is an input, assumes ECO bit in the CMR = 1.

Figure 6-6. DMA Timing Diagram (IDMA)

6.13 AC ELECTRICAL SPECIFICATIONS—CHIP-SELECT TIMING INTERNAL MASTER (see Figure 6-14)

			16.67 MHz		20 MHz		25 MHz		
Num.	Characteristic	Symbol	Min	Max	Min	Max	Min	Max	Unit
150	Clock High to $\overline{CS}$, $\overline{IACK}$ Low (see Note 2)	$t_{CHCSIAKL}$	0	40	0	35	0	27	ns
151	Clock Low to $\overline{CS}$, $\overline{IACK}$ High (see Note 2)	$t_{CLCSIAKH}$	0	40	0	35	0	27	ns
152	$\overline{CS}$ Width Negated	t_{CSH}	60	—	50	—	40	—	ns
153	Clock High to $\overline{DTACK}$ Low (0 Wait State)	t_{CHDTKL}	—	45	—	40	—	30	ns
154	Clock Low to $\overline{DTACK}$ Low (1–6 Wait States)	t_{CLDTKL}	—	30	—	25	—	20	ns
155	Clock Low to $\overline{DTACK}$ High	t_{CLDTKH}	—	40	—	35	—	27	ns
156	Clock High to $\overline{BERR}$ Low (see Note 1)	t_{CHBERL}	—	40	—	35	—	27	ns
157	Clock Low to $\overline{BERR}$ High Impedance (see Note 1)	t_{CLBERH}	—	40	—	35	—	27	ns
158	$\overline{DTACK}$ High to $\overline{DTACK}$ High Impedance	$t_{DTKHDTKZ}$	—	15	—	15	—	10	ns
171	Input Data Hold Time from S6 Low	t_{IDHCL}	5	—	5	—	5	—	ns
172	$\overline{CS}$ Negated to Data-Out Invalid (Write)	t_{CSNDOI}	10	—	10	—	7	—	ns
173	Address, FC Valid to $\overline{CS}$ Asserted	t_{AFVCSA}	15	—	15	—	15	—	ns
174	$\overline{CS}$ Negated to Address, FC Invalid	t_{CSNAFI}	15	—	15	—	12	—	ns
175	$\overline{CS}$ Low Time (0 Wait States)	t_{CSLT}	120	—	100	—	80	—	ns
176	$\overline{CS}$ Negated to R/$\overline{W}$ Invalid	t_{CSNRWI}	10	—	10	—	7	—	ns
177	$\overline{CS}$ Asserted to R/$\overline{W}$ Low (Write)	t_{CSARWL}	—	10	—	10	—	8	ns
178	$\overline{CS}$ Negated to Data-In Invalid (Hold Time on Read)	t_{CSNDII}	0	—	0	—	0	—	ns

NOTE:
1. This specification is valid only when the ADCE or WPVE bits in the SCR are set.
2. For loading capacitance less than or equal to 50 pF, subtract 4 ns from the maximum value given.
3. Since $\overline{AS}$ and $\overline{CS}$ are asserted/negated on the same CLKO edges, no $\overline{AS}$ to $\overline{CS}$ relative timings can be specified. However, $\overline{CS}$ timings are given relative to a number of other signals, in the same manner as $\overline{AS}$. See Figure 6-2 and Figure 6-3 for diagrams.

6.17 AC ELECTRICAL SPECIFICATIONS—TIMERS

(see Figure 6-18)

			16.67 MHz		20 MHz		25 MHz		
Num.	Characteristic	Symbol	Min	Max	Min	Max	Min	Max	Unit
200	Timer Input Capture Pulse Width	t_{TPW}	50	—	42	—	34	—	ns
201	TIN Clock Low Pulse Width	t_{TICLT}	50	—	42	—	34	—	ns
202	TIN Clock High Pulse Width and Input Capture High Pulse Width	t_{TICHT}	2	—	2	—	2	—	clk
203	TIN Clock Cycle Time	t_{cyc}	3	—	3	—	3	—	clk
204	Clock High to TOUT Valid	t_{CHTOV}	—	35	—	30	—	24	ns
205	$\overline{FRZ}$ Input Setup Time (to Clock High) (see Note 1)	t_{FRZSU}	20	—	20	—	14	—	ns
206	$\overline{FRZ}$ Input Hold Time (from Clock High)	t_{FRZHT}	10	—	10	—	7	—	ns

NOTES:
1. $\overline{FRZ}$ should be negated during total system reset.
2. The TIN specs above do not apply to the use of TIN1 as a baud rate generator input clock. In such a case, specifications 1–3 may be used.

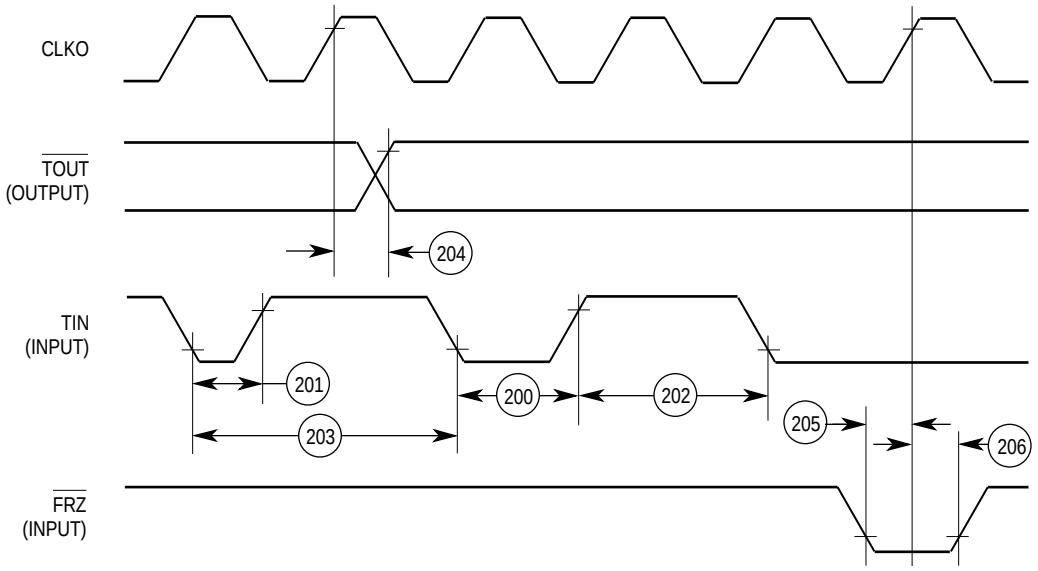


Figure 6-18. Timers Timing Diagram

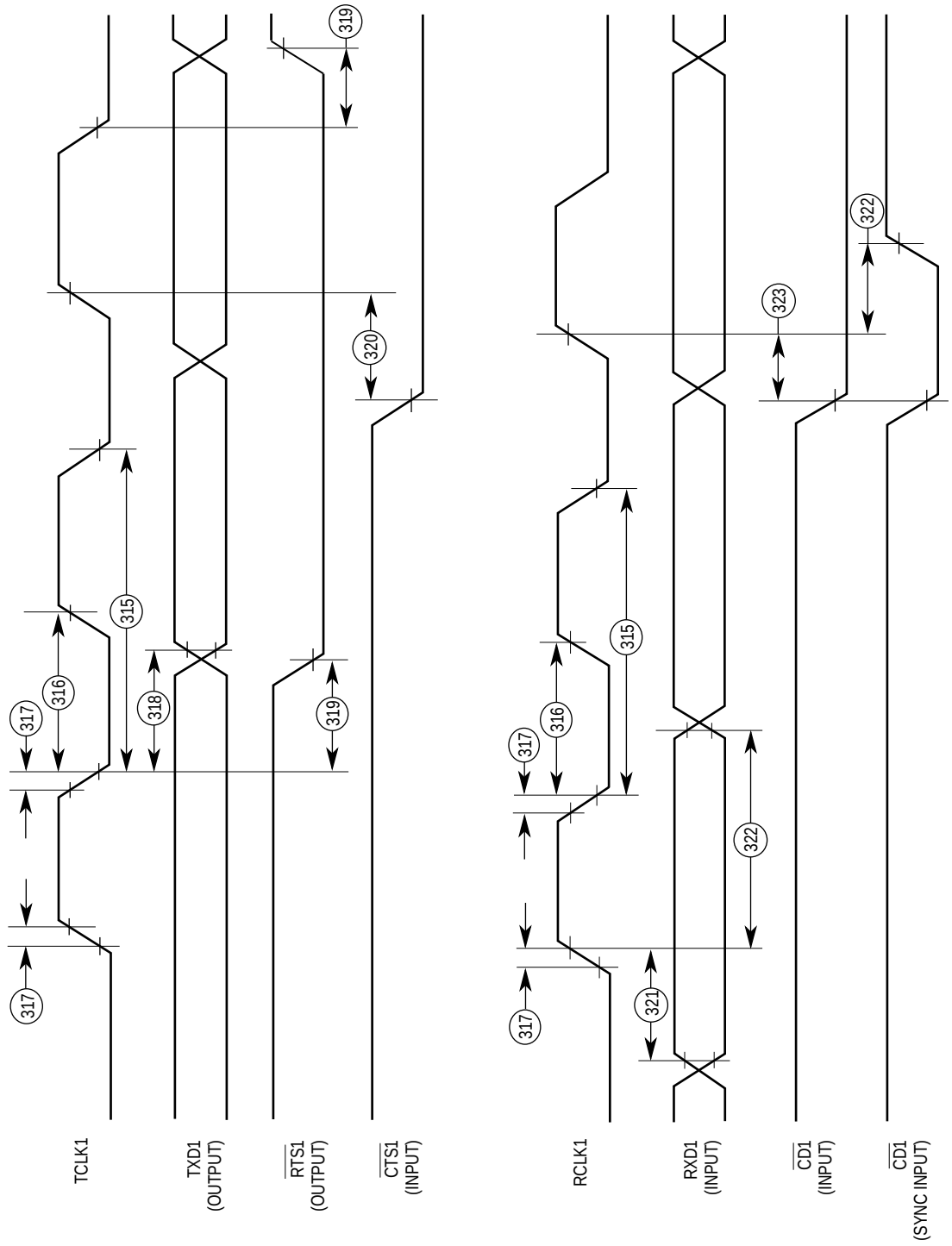
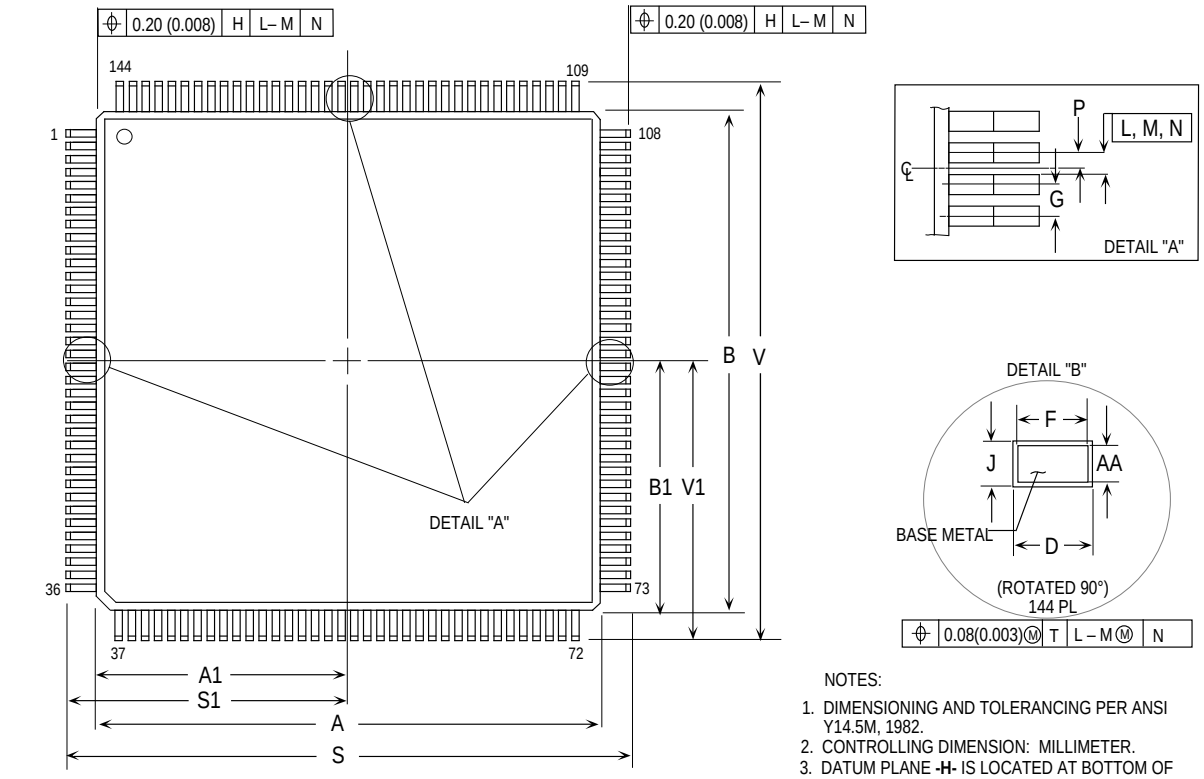
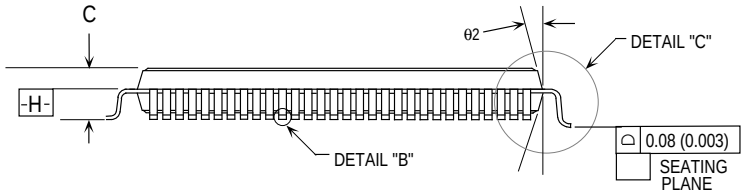


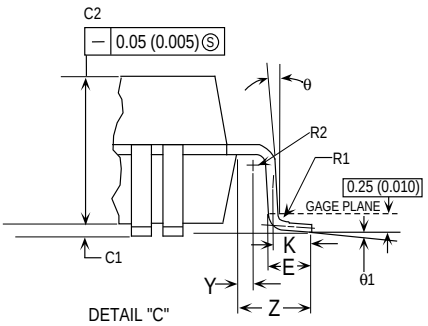
Figure 6-24. NMSI Timing Diagram

7.2.3 Thin Surface Mount (TQFP)

CASE 918-02
144 TQFP







NOTES:

1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE **-H-** IS LOCATED AT BOTTOM OF LEAD AND COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS **-L-**, **-M-**, AND **-N-** TO BE DETERMINED AT DATUM PLANE **-H-**.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE **-T-**.
6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 (0.010 PER SIDE. DIMENSIONS A AND B DO NOT INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM LINE **-H-**.
7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE D DIMENSION TO EXCEED 0.35 (0.014).

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	20.00 BSC		0.790 BSC	
A1	10.00 BSC		0.394 BSC	
B	20.00 BSC		0.790 BSC	
B1	10.00 BSC		0.394 BSC	
C	1.40	1.60	0.055	0.063
C1	0.05	0.15	0.002	0.006
C2	1.35	1.45	0.053	0.057
D	0.17	0.27	0.007	0.011
E	0.45	0.75	0.018	0.030
F	0.17	0.23	0.007	0.009
G	0.50 BSC.		0.20 BSC.	
J	0.09	0.20	0.004	0.008
K	0.50 REF		0.020 REF	
P	0.25 BSC		0.010 BSC	
R1	0.13	0.20	0.005	0.008
R2	0.13	0.20	0.005	0.008
S	22.00 BSC		0.866 BSC	
S1	11.00 BSC		0.433 BSC	
V	22.00 BSC		0.866 BSC	
V1	11.00 BSC		0.433 BSC	
Y	0.25 REF		0.010 REF	
Z	1.00 REF		0.039 REF	
AA	0.09	0.16	0.004	0.006
θ	0°		0°	
θ1	0°	7°	0°	7°
θ2	11°	13°	11°	13°

er register (SAPR). The pointer to the destination is located in the destination address pointer register (DAPR). The byte count register (BCR) specifies the number of bytes to be transferred and is decremented once for each byte transferred. Note that the six SDMA channels on the MC68302 have a higher priority than the IDMA (unless the IDMA is performing a maximum rate burst).

D.5.2 IDMA Software Initialization

Information that describes the data block to be moved, the transfer methods, and the control options is loaded into the channel mode register (CMR). See Table D-2 for these required selections. Loading the SAPR, DAPR, and BCR are also part of the initialization procedure. If the functions of the $\overline{\text{DREQ}}$, $\overline{\text{DACK}}$, and $\overline{\text{DONE}}$ pins are required, then their corresponding bits must be enabled in the port A control register (PACNT).

D.5.3 IDMA Bus Arbitration Signals

Whenever the IDMA wants to transfer data, it must arbitrate for the external bus. Two internal signals are used: IDMA bus request ($\overline{\text{IDBR}}$) and IDMA bus grant ($\overline{\text{IDBG}}$). When no external resource controls the bus ($\overline{\text{BGACK}}$ is negated) nor any internal resource (such as the SCCs) requires the use of the bus, then the IDMA can be granted the bus when it asserts (internally) $\overline{\text{IDBR}}$. Obtaining bus mastership occurs when 1) $\overline{\text{AS}}$ and $\overline{\text{BGACK}}$ are high (negated) and 2) external $\overline{\text{BR}}$ and $\overline{\text{BG}}$ are not indicating an external master wants the bus. To indicate that the IDMA has taken control of the bus, $\overline{\text{BGACK}}$ is asserted while $\overline{\text{BR}}$ and $\overline{\text{BG}}$ are not asserted. The IDMA function code bits in the function code register (FCR) can be further used to distinguish between the source and destination accesses.

D.5.4 Triggering External IDMA Transfers

The data request ($\overline{\text{DREQ}}$) input signal should be asserted by a peripheral when it requires service. The MC68302 responds with data acknowledge ($\overline{\text{DACK}}$) output signal which asserts when the IDMA is moving data from/to the peripheral device. A third signal, $\overline{\text{DONE}}$, indicates when the last DMA transfer cycle is in progress. The bidirectional $\overline{\text{DONE}}$ signal is asserted by the IDMA when the BCR has decremented to zero or can be asserted by the peripheral to terminate the data transfer.

D.5.5 Performing Internally Generated IDMA Transfers

The procedure to move blocks of data from one memory space to another can be achieved by first loading the SAPR, DAPR, BCR, and CMR. The DMA transfer sequence begins by setting the STR bit (see Table D-2) in the CMR. The transfer stops when the BCR has decremented to zero, the external $\overline{\text{DONE}}$ pin is asserted externally, or the software clears the STR bit in the CMR. The percentage of bus usage can be controlled by the BT bits to keep the IDMA from exceeding 12.5, 25, 50, or 75 percent of the available bus bandwidth. The transfer can also be interrupted by the $\overline{\text{BCLR}}$ signal, which can be activated through the SDMA channels, or by an interrupt (see 3.8.3 System Control Bits).

for a given protocol. For HDLC and transparent, this is every 16 bits; whereas, in the other protocols, it is every 8 bits on receive and every 16 bits on transmit.

This application does not address the issue of choosing whether a master-slave arrangement is the best approach for your application, but rather looks at the design issues that need to be addressed once the approach seems reasonable. (Most applications usually find the master-slave approach shown in Figure D-19 quite acceptable for data rates up to 64 kbps on each of the nine SCCs. Any comments on rates beyond this value tend to be less general and more application dependent.)

D.7.1 Synchronous vs. Asynchronous Accesses

When a device is in slave mode, there are two different ways it can be accessed: synchronously and asynchronously.

When the MC68302 enters slave mode, it is initialized with asynchronous accesses. The term “asynchronous” refers to the fact that the master does not need to supply signals to the slave on particular system clock edges, as required in synchronous. However, asynchronous accesses are always and only three waits states for both reads and writes. In general, this is the suggested method of accessing the MC68302 in slave mode. These accesses are much simpler than in synchronous timing. Any performance loss is minimal since, once the MC68302 is initialized, accesses to it are rare. The SDMA channels, which are unaffected by this choice, can still operate with zero wait states and comprise most of the slave MC68302 activity.

If the SAM bit in the SCR is set, the following accesses to the slave MC68302 will be synchronous. Writes will be zero wait states, and reads can be either zero or one wait state, based on the EMWS bit in the SCR. The EMWS bit is almost always required to meet synchronous setup times. Synchronous accesses should be used when one master MC68302 is accessing another slave MC68302 without any intervening glue (like buffers). In this case, the synchronous timings can be met with proper clocking.

D.7.2 Clocking

The master clock out (CLKO) line should be connected to the EXTAL input of the slaves, allowing the master MC68302 to perform zero wait state writes and one wait state reads from the slave MC68302 at full 16.67-MHz operation in synchronous mode. The guaranteed propagation delay between EXTAL and CLKO is 2-11 ns at 16.67-MHz operation. The typical delay is about 5 ns.

If a single external clock is required to drive the inputs to all the MC68302 EXTAL pins, the skew between EXTAL and CLKO must be watched by the designer. In the case of synchronous accesses, this will certainly slow the overall frequency that can be designed.

D.7.3 Programming the Base Address Registers (BARs)

The next issue is how to program the three BARs on each of the MC68302s considering they have the same address ($0F2). Figure D-19 shows an easy method of programming the BAR. The master MC68302 first programs its BAR at $0000F2. For each slave, it writes the slave's parallel I/O line with a zero and writes to $8000F2 to program the slave's BAR. This

SCCs on the MC68302 can do this very efficiently because of their sophisticated DMA capability, and very little MC68000 core intervention is required.

Third, some applications require the switching of data without interfering with the protocol encoding itself. For instance, in a multiplexer, data from a high-speed time-multiplexed serial stream is multiplexed into multiple low-speed data streams. In this case, the idea is to switch the data path but not alter the protocol encoded on that data path.

Finally, some applications require a special protocol that does not fall under the category of HDLC, UART, etc. In some instances, transparent mode can be used; however, care should be taken to understand the capabilities of transparent mode before trying this. The most important issue is how this new protocol recognizes its frames on the receiving end. Transparent mode on the MC68302 was designed to work well receiving continuous streams of data (no gaps in the data exist over time). This is different from receiving transparent frames; although there is some support for this, it is limited on the MC68302.

D.8.3 Physical Interface to Accompany Transparent Mode

Before discussing the details of transparent mode timing, we need to chose the physical interface to go with the transparent mode. The timings associated with transparent mode differ based on the physical interface chosen.

The MC68302 supports the following four physical interfaces:

- Nonmultiplexed Serial Interface—NMSI
- Pulse Code Modulation Highway—PCM
- Interchip Digital Link—IDL
- General Circuit Interface—GCI

You will probably chose either an NMSI or PCM highway interface, unless you are designing an ISDN-based system. If you are designing an ISDN-based system, you will probably choose either an IDL or GCI interface. The following paragraphs discuss all the interfaces, but special attention is given to the NMSI.

NOTE

The following discussion assumes some knowledge of the interfaces. For more applications information on these interfaces, refer to 4.4 Serial Channels Physical Interface**.**

The most commonly used physical interface on the MC68302 is the nonmultiplexed serial interface (NMSI). The NMSI consists of seven basic modem (RS-232) signals: $\overline{\text{TXD}}$, $\overline{\text{TCLK}}$, $\overline{\text{RXD}}$, $\overline{\text{RCLK}}$, $\overline{\text{RTS}}$, $\overline{\text{CTS}}$, and $\overline{\text{CD}}$. Each of the three SCCs can have its own set of these signals, as shown in Figure D-21.

of the physical interface configuration. Similarly, all bits in the receive buffer will be filled with real transparent data (full packing is always performed), regardless of the physical interface configuration.

If no data is available to transmit, transparent mode will transmit ones. The decision of whether to set the last (L) bit in the Tx BD is left to the user. If multiple buffers are to be sent back-to-back with no gaps in between, the L bit should be cleared in all buffers except for the last buffer. In this case, failure to provide buffers in time will result in a transmit underrun. If the L bit is set, the frame will end without error, and the transmission of ones will resume.

The transmit byte count and buffer alignment need not be even, but the SDMA channel will always read words on an even-byte boundary, even if it has to discard one of the two bytes. For example, if a transmit buffer begins on an odd-byte boundary and is 10 bytes in length (worst case), six word reads will result, even though only 10 bytes will be transmitted.

The receive buffer length (stored in MRBLR) and starting address must be even. All transfers to memory will be of word length and, unless an error occurs, a buffer will not be closed until it contains MRBLR/two words (the byte count will be equal to MRBLR). This raises an important point. Data received will only be transmitted to memory every 16 clocks. If a non-multiple of 16 bits is sent in a frame, the residue bits will not be transmitted to memory until additional bits arrive, and it will be impossible to demarcate frames unless their length is predetermined. (If a SYNC character is received with the data, the BISYNC mode can be used to receive an odd number of bytes with odd-length receive buffers and pointers allowed. (For more detailed information, refer to D.8.6 Other NMSI Modes**.)**

When the enable transmitter (ENT) bit is set, the process of polling the Tx BD begins by the RISC. The frequency of this polling is determined by the SCC's transmit clock. If the clock is stopped, no polling will occur. When the ready bit of the first Tx BD is set, the RISC initiates the SDMA activity of filling up the transmit FIFO with three words of data. Once the FIFO is full, the $\overline{\text{RTS}}$ signal is asserted, and the physical interface signals take control to determine the exact timing of the transmitted data. Once the physical interface says “go”, typically one final $FF is transmitted before data begins; however, whether $FF is transmitted depends on the mode chosen.

When the enable receiver (ENR) bit is set and 16 bits of valid data (as defined by the physical interface signals) have been clocked into the receiver, the RISC checks to see if the first receive buffer is available and, if the buffer is available, begins moving the data to it. The receive FIFO is three words deep, but a single open entry in the FIFO causes an SDMA service request. There are three types of receive errors: overrun (receive FIFO overflow), busy (new data arrived without a receive buffer being available), and $\overline{\text{CD}}$ lost (which is not possible in any example configuration discussed in this appendix). These errors are reported in the SCC event register (SCCE) or the Rx BD.

Whenever a buffer has been transmitted with the interrupt (I) bit set in the Tx BD, the TX event in the SCCE register will be set. This TX bit can cause an interrupt if the corresponding bit in the SCCM is set. Similarly, whenever a buffer has been received with the interrupt (I) bit set in the Rx BD, the RX event in the SCCE register will be set. Also, whenever a word of data is written to the receive buffer, the RCH bit is set in the SCCE.

Figure D-25 shows how $\overline{CTS}$ can be used in the NMSI transmit case. NTSYN and EXSYN are set to enable transparent mode. Instead of software operation for $\overline{CTS}$ and $\overline{CD}$, normal (automatic) operation is chosen. $\overline{RTS}$ is asserted when the transmit FIFO is full. From then on, data is held off until $\overline{CTS}$ is sampled low. From that sample point, there is a 3.5 TCLK delay before the first bit of the data buffer is transmitted. Ones are transmitted until the first bit of the data buffer is transmitted.

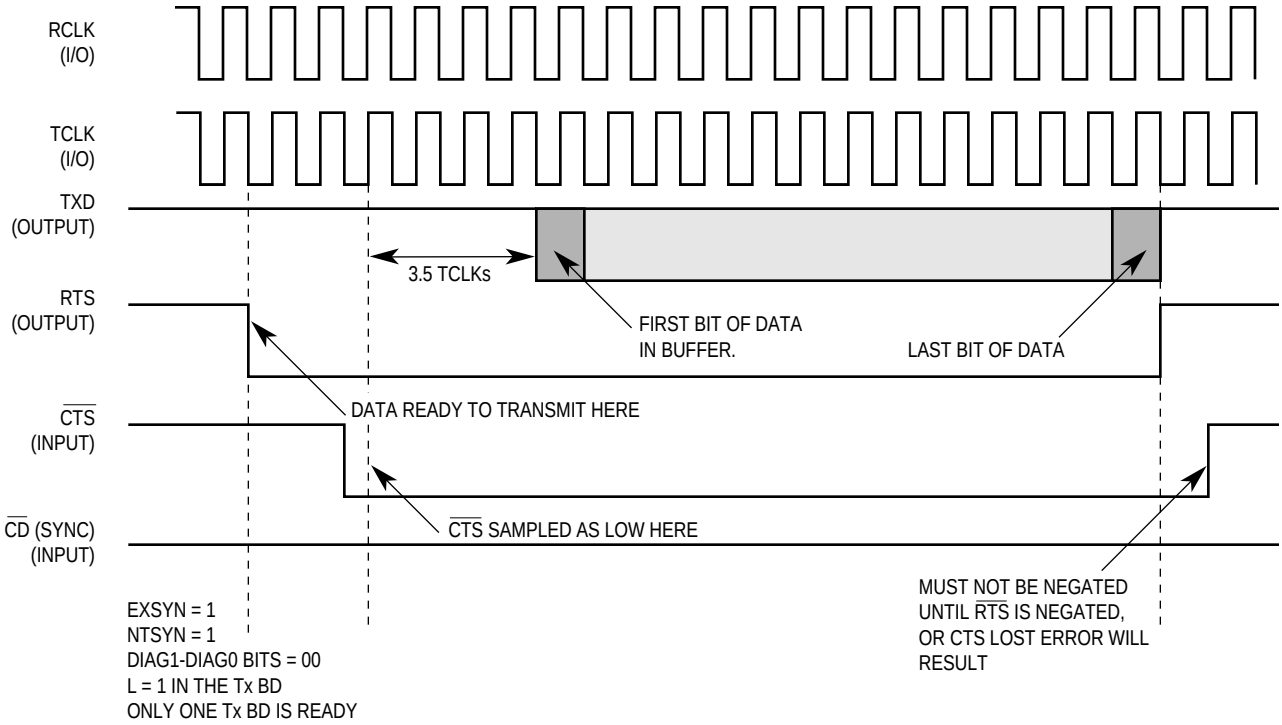


Figure D-25. Using $\overline{CTS}$ In the NMSI Transmit Case

In the case shown in Figure D-25, it is important that $\overline{CTS}$ not go high for the duration of the buffer transmission. If multiple buffers are all ready with their L bits cleared, transmission of frames will continue back-to-back. If $\overline{CTS}$ negated during any of these buffers, transmission will cease, and that buffer will report a $\overline{CTS}$ lost condition. Ones will be transmitted at that time. Once a restart transmit command is given, transmission of the next buffer can begin once $\overline{CTS}$ is reasserted.

Once $\overline{CTS}$ deasserts after $\overline{RTS}$, the $\overline{RTS}$-$\overline{CTS}$ protocol can begin again as soon as the next buffer is made ready, but a minimum of 17 idle bits will occur between frames, regardless of how soon $\overline{CTS}$ is reasserted. Remember that when EXSYN is set, $\overline{CD}$ (sync) must be low for transmission to begin. In this case, it is grounded; whereas, in the following case, EXSYN is actively switching.

Figure D-26 shows how $\overline{CD}$ (sync) can be used to control transmission. EXSYN and NTSYN are once again set to enable transparent mode, and the L bit is set. Since software operation mode (DIAG1 = 1 and DIAG0 = 1) is chosen, the $\overline{CTS}$ pin value is ignored. Once $\overline{CD}$ (sync) is latched low, data begins transmission in 6.5 TCLKs. Notice that the rising edge of $\overline{CD}$ (sync) and subsequent falling edges of $\overline{CD}$ (sync) (not shown) have no effect, since synchronization has already been achieved.

UM1, UM0—UART Mode 1-0

00 = Normal UART operation.
01 = Nonautomatic multidrop mode. No automatic address recognition is performed.
10 = DDCMP protocol is implemented over the asynchronous channel.
11 = Automatic multidrop mode. The IMP automatically checks the value of the incoming address character and accepts the data following it only if the address matches the 8-bit value in either UADDR1 or UADDR2.

FRZ—Freeze Transmission

0 = Normal operation.
1 = The IMP stops transmitting after transmitting any data already transferred to the transmit FIFO.

CL—Character Length

0 = 7-bit character length. On receive, bit 7 is written to memory as a zero.
1 = 8-bit character length.

RTSM—RTS Mode

0 = $\overline{\text{RTS}}$ is asserted whenever the transmitter is enabled and there are characters to transmit.
1 = $\overline{\text{RTS}}$ is asserted whenever the transmitter is enabled.

SL—Stop Length

0 = One stop bit.
1 = Two stop bits.

DIAG1, DIAG0—Diagnostic Mode

00 = Normal operation.
01 = Loopback mode.
10 = Automatic echo.
11 = Software operation.

ENR—Enable Receiver

0 = Receiver is disabled.
1 = Receiver is enabled.

ENT—Enable Transmitter

0 = Transmitter is disabled.
1 = Transmitter is enabled.

MODE1, MODE0—Channel Mode

00 = HDLC.
01 = Asynchronous (UART and DDCMP).
10 = Synchronous DDCMP and V.110.
11 = BISYNC and Promiscuous (Transparent).

23. **SCCs, Diagnostics**
When an unexplained SCC problem arises, three important debugging techniques can be used to help isolate the problem area. First, to see if the problem is related to the external M68000 bus, try configuring the transmit and/or receive buffers to be located in the internal dual-port RAM. In this way, the external bus will not be required for transmission or reception. Second, to see if the problem is related to the serial interface pins, try setting the DIAG1–DIAG0 bits in the SCM register to loopback mode or software operation. These techniques allow transmission and reception to be tested without proper settings of the serial interface control pins. Loopback mode is also a good test mechanism to build into the application board test code. Third, to see if the problem is related to rise times, noise, or glitches on external serial clocks, try using the SCC internal baud rate generator or try generating a clock externally with another baud rate generator or timer and routing that clock externally to the SCC.

24. **UART, MAX_IDL**
One common problem with UART mode is the setting of the MAX_IDL value. The maximum value is $0000, and the minimum value is $0001. A large value causes the receive buffer not to close until the entire idle period is complete. At slow serial data rates, this time can be several minutes.

25. **Write Zeroes into Reserved Locations**
Zero fill all reserved locations to prevent unexpected behavior.

26. **Event Registers, Read-Modify-Write**
Do not use instructions that execute a read-modify-write cycle (e.g., ANDI, ORI, BSET, BCLR) to clear bits in the event registers. A bit in such a register is cleared by writing a one, rather than a zero, to the bit. The use of instructions that execute a read-modify-write cycle inadvertently clears all bits in the event register. The clearing of all bits in an event register could result in “lost” events. For example, when an interrupt handler intentionally or unintentionally clears all bits of an event register upon exit of the handler, all events occurring between the time of the entrance and exit of the interrupt handler are lost. A proper handling of event registers is to clear particular bits near the entrance of an interrupt handler. A proper instruction to use is the MOVE instruction (e.g., “MOVE #$01, SCCE” clears bit 0 of the SCCE register).

27. **Microcode, RAM**
If microcode from RAM is used (i.e., one of the packages available from Motorola), the microcode must be downloaded into the MC68302 dual-port RAM prior to setting location $0F8 to $0001. If this sequence is not followed, the CP will not function properly, regardless of which protocol is selected.