PRELIMINARY

Am85C80

Combined SCSI Controller and Serial Communications Controller

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- **The Am85C80 is an Enhanced Serial Communications Controller (ESCC) and Small Computer Systems Interface (SCSI) Controller**
  - This chip replaces both the 85C30 Serial Communications Controller (SCC) and the 53C80 SCSI Controller in functionality, while combining the two chips into a single package
  - The sleep mode feature consumes very little power; this feature may be used in notebook computer and other battery operated portable systems

ESCC

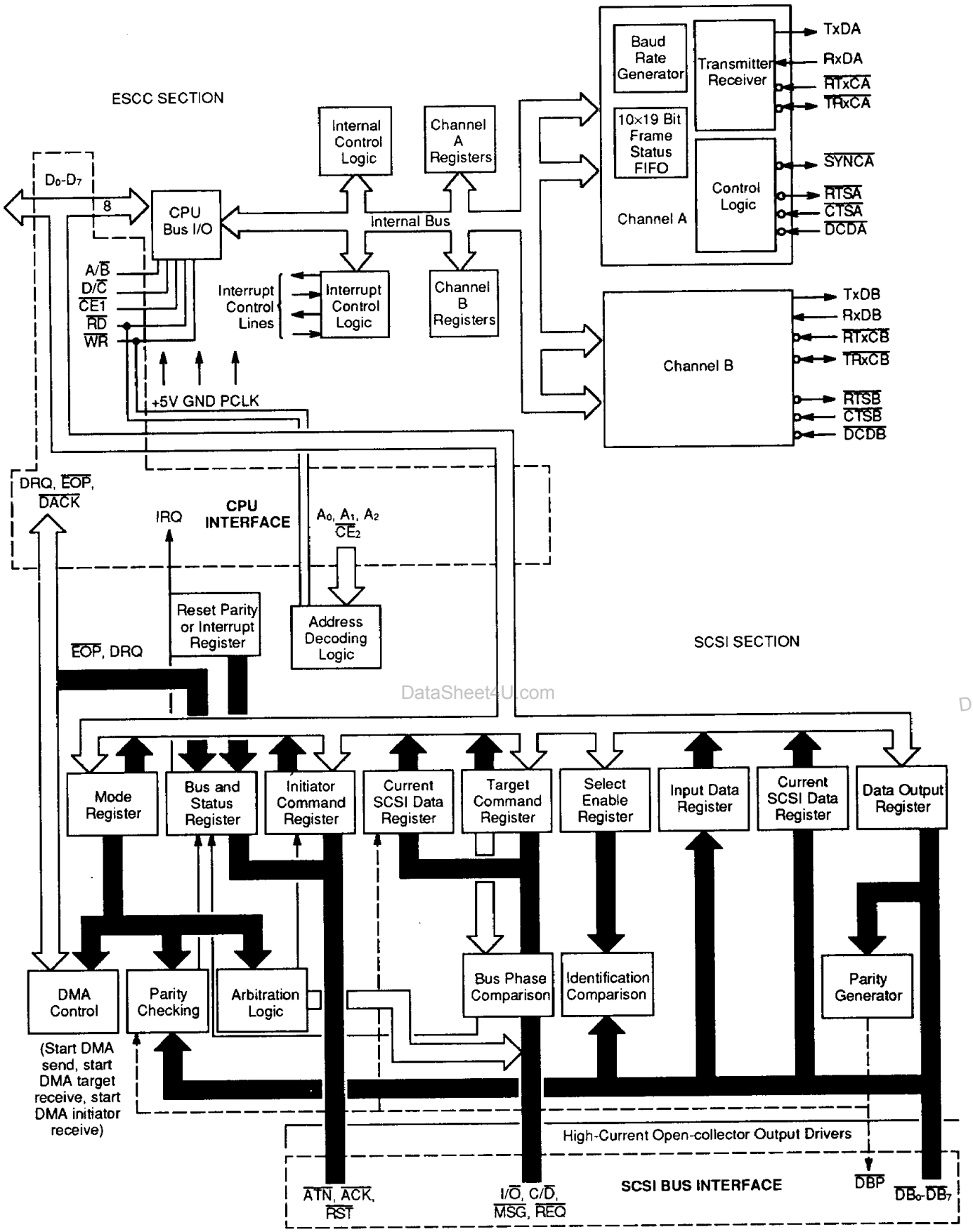
- **Fast Data Rate**
  - 8.192 MHz/2.048 Mb/s
  - 16.384 MHz/4.096 Mb/s
- **Low Power CMOS Technology**
- **Easily interfaced with most CPUs**
  - Compatible with non-multiplexed bus
- **Two Independent Full-duplex Serial Channels**
- **Asynchronous Mode Features**
  - Programmable stop bits, clock divider, character length and parity
  - Break detection/divider
  - Error detection for framing, overrun and parity
- **Synchronous Mode Features**
  - Supports IBM BISYNC, SDLC, SDLC Loop, HDLC and ADCCP Protocols
  - Programmable CRC generators and checkers
  - SDLC/HDLC support includes frame control, zero insertion and deletion, abort, and residue handling
  - External sync mode is supported on Channel A only
- **Enhanced ESCC functions support high-speed frame reception using DMA**
  - 14-bit frame byte counter
  - 10 × 19 SDLC/HDLC Frame Status FIFO
  - Independent Control on both channels
  - Enhanced operation does not allow special receive conditions to lock the three-byte DATA FIFO when the 10 × 19 FIFO is enabled
- **Local Loopback and Auto Echo Modes**
- **Internal or External Character Synchronization**
- **2 Mb/s FM Encoding Transmit and Receive capability using Internal DPLL for 16.384 MHz product**
- **Internal Synchronization between RxC to PCLK and TxC to PCLK**
  - This allows the user to eliminate external sychronization hardware required by the NMOS device when transmitting or receiving data at the maximum rate of 1/4 PCLK frequency

SCSI Interface

- **Asynchronous interface to 1.5 megabytes per second**
- **Supports Initiator and Target roles**
- **Parity generation with optional checking**
- **Supports Arbitration**
- **Direct control of all bus signals**
- **High-current outputs drive SCSI Bus directly**
- **Implements Glitch Eater circuitry that filters glitches upto 30 ns on the SCSI Bus**
- **CPU Interface Supports**
  - Memory or I/O-mapped interface
  - DMA or programmed I/O
  - Normal or Block mode DMA
  - Optional CPU interrupts

Publication # 12582 Rev. C Amendment /0
Issue Date: November 1991

BLOCK DIAGRAM



12582C-096B

Block Diagram

GENERAL DESCRIPTION

The Am85C80 is an Enhanced Serial Communications Controller and SCSI Interface Controller. The chip replaces both Am85C30 and Am53C80 in functionality, while combining the two chips into a single package.

ESCC

The Enhanced Serial Communications Controller (ESCC) is a high-speed, low-power, multi-protocol communications peripheral designed for use with 8- and 16-bit microprocessors. It has two independent, full duplex channels and functions as a serial-to-parallel, parallel-to-serial converter/controller. AMD's proprietary enhancements make the ESCC easier to interface and more effective in high-speed applications by reducing software overhead and eliminating the need for some external glue logic.

The ESCC is easy to use due to a variety of sophisticated internal functions, including on-chip baud rate generators, digital phase-locked loops and crystal oscillators which dramatically reduce the need for external logic. The device can generate and check CRC codes in any SYNC mode, and can be programmed to check data integrity in various modes. The ESCC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

This versatile device supports virtually any serial data transfer application such as laser beam printers, networks, modems, cassettes and tape drives. The ESCC is designed for non-multiplexed buses and is easily interfaced with most CPUs, such as 80188, 80186, 80286, 8080, Z80, 6800, 68000 and MULTIBUS.

Enhancements which allow the ESCC to be used more effectively in high-speed applications include:

- a 10 × 19 bit SDLC/HDLC frame status FIFO array
- a 14-bit SDLC/HDLC frame byte counter
- automatic SDLC/HDLC opening frame flag transmission
- TxD pin forced High in SDLC NRZI mode after closing flag
- automatic SDLC/HDLC Tx underrun/EOM flag reset
- automatic SDLC/HDLC Tx CRC generator reset/preset
- $\overline{RTS}$ synchronization to closing SDLC/HDLC flag
- $\overline{DTR/REQ}$ deactivation delay significantly reduced
- external PCLK to $\overline{RxC}$ or $\overline{TxC}$ synchronization requirement eliminated for PCLK divide-by-four operation
- 16 MHz operation

Other enhancements to improve the ESCC interface capabilities include:

- write data valid setup time to falling edge of $\overline{WR}$ requirement eliminated
- reduced $\overline{INT}$ response time
- reduced access recovery time (t_{RC}) to 3 PCLK best case (3 1/2 PCLK worst case)
- improved $\overline{Wait}$ timing
- write registers WR3, WR4, WR5, and WR10 made readable
- lower priority interrupt masking without $\overline{INTACK}$
- complete SDLC/HDLC CRC character reception
- Sleep mode feature saves all ESCC registers

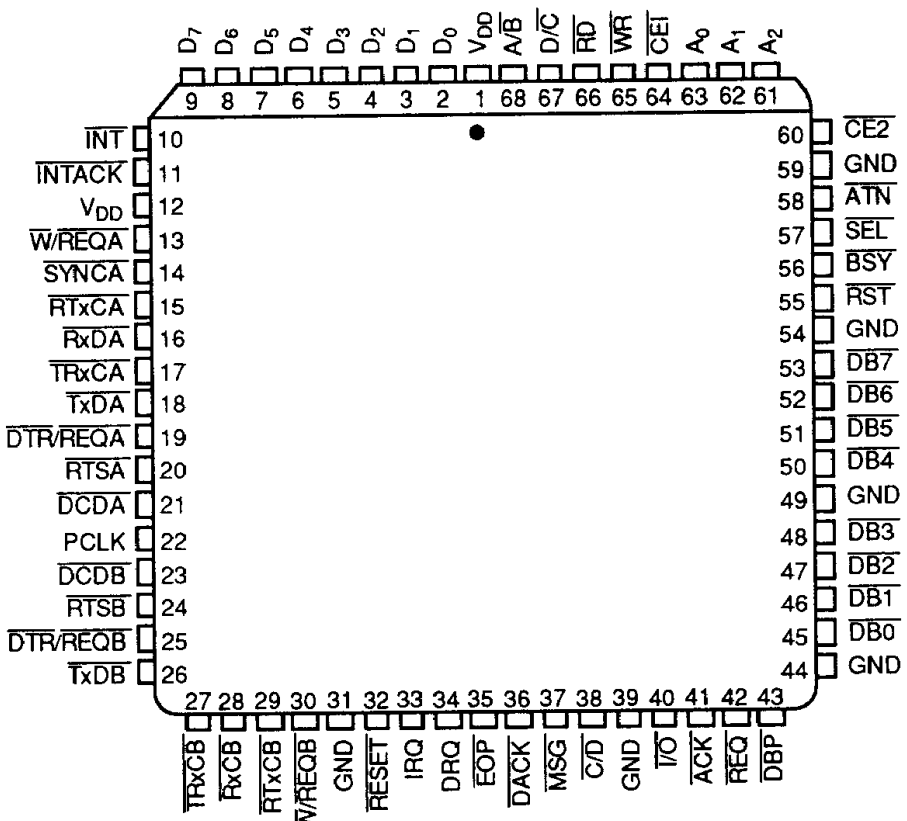
SCSI

The Small Computer Systems Interface (SCSI) Interface Controller is a CMOS device designed to accommodate the SCSI as defined by the ANSI X3T9.2 committee. The Am85C80 operates in both the Initiator and Target roles and can, therefore, be used in host adaptor, host port, disk drive, and formatter designs. This device supports Arbitration, including Reselection. Special high-current open-collector output drivers, capable of sinking 48 mA at 0.5 V, allow for direct connection to the SCSI Bus.

The Am85C80 communicates with the system microprocessor as a peripheral device. The chip is controlled by reading and writing several internal registers which may be addressed as standard or memory-mapped I/O. Minimal processor intervention is required for DMA transfers because the Am85C80 controls the necessary handshake signals. The Am85C80 interrupts the CPU when it detects a bus condition that requires attention. Normal and Block mode DMA is provided to match many popular DMA controllers.

CONNECTION DIAGRAM
Top View

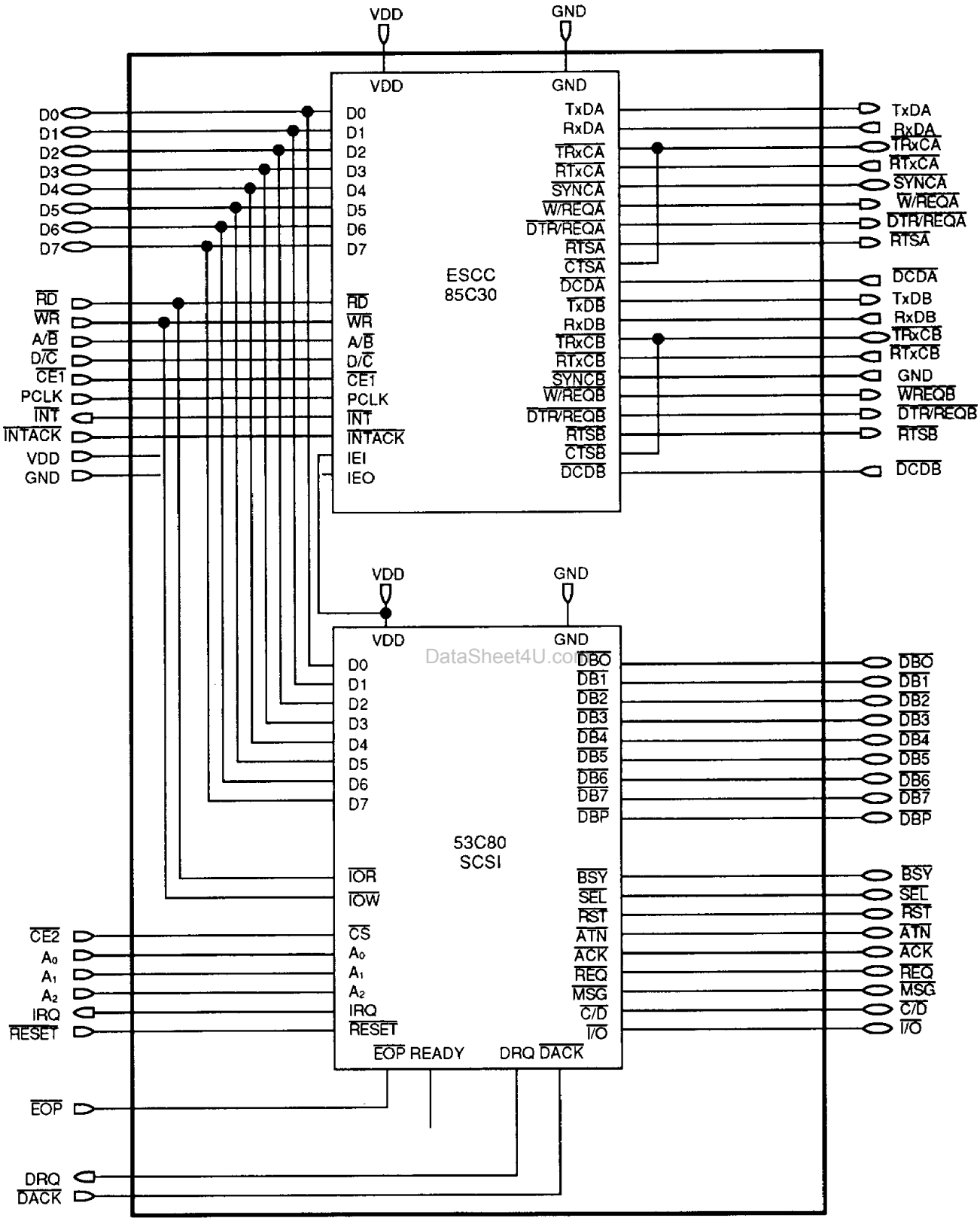
PLCC



12582C-097A

Note:
Pin 1 is marked for orientation.

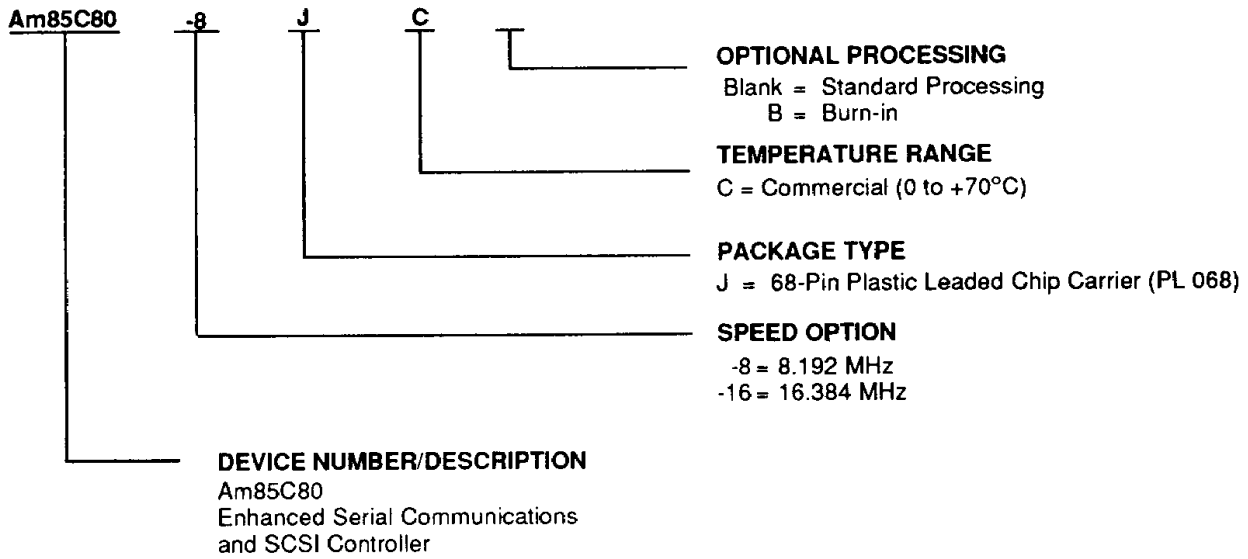
CONNECTION DIAGRAM
Detailed Internal



12582C-098B

ORDERING INFORMATION
Commodity Products

AMD commodity products are available in several packages and operating ranges. The ordering number (Valid Combination) is formed by a combination of:



Valid Combinations	
Am85C80-8 Am85C80-16	JC

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations and to check on newly released combinations.

PIN DESCRIPTION

ESCC

RTxCA, RTxCB

Receive/Transmit Clocks (Inputs; Active Low)

These pins can be programmed in several different modes of operation. In each channel, RTxC may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock of the digital phase-locked loop. RTxCA can also be programmed for use with the SYNCA pin as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

TRxCA, TRxCB

Transmit/Receive Clocks (Inputs/Outputs; Active Low)

These pins can be programmed in several different modes of operation. TRxC may supply the receive clock or the transmit clock in the input mode or supply the output of the digital phase-locked loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

Channel Controls for Modem, DMA, or Other

DCDA, DCDB

Data Carrier Detect (Inputs; Active Low)

These pins function as receiver enables if they are programmed as Auto Enables; otherwise, they may be used as general-purpose input pins. Both are Schmitt-trigger buffered to accommodate slow rise-time signals. The ESCC detects pulses on these pins and may interrupt the CPU on both logic level transitions.

DTR/REQA, DTR/REQB

Data Terminal Ready/Request (Outputs; Active Low)

These outputs follow the inverted state programmed into the DTR bit in WR5. They can also be used as general-purpose outputs or as Request Lines for a DMA controller.

RTSA, RTSB

Request to Send (Outputs; Active Low)

When the Request to Send RTS bit in Write Register 5 is set, the RTS signal goes Low. When the RTS bit is reset in the asynchronous mode and Auto Enable is on, the signal goes High after the transmitter is empty. In SYNC mode or in asynchronous mode with Auto Enable off, the RTS pins strictly follow the inverted state of the RTS bit. Both pins can be used as general-purpose outputs.

SYNCA

Synchronization (Input/Output; Active Low)

This pin can act either as input, output, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), this pin is input similar to DCD. In this mode, transitions on this line affect the state of the Sync/Hunt status bits in Read Register 0 but have no other function.

In External Synchronization mode with the crystal oscillator not selected, this line also acts as an input. In this mode, SYNCA must be driven Low two receive clock cycles after the last bit in the SYNC character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of SYNCA.

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, this pin acts as an output and is active only during the part of the receive clock cycle in which SYNC characters are recognized. The SYNC condition is not latched, so this output is active each time a SYNC pattern is recognized (regardless of character boundaries). In SDLC mode, this pin acts as an output and is valid on receipt of a flag.

W/REQA, W/REQB

Wait/Request (Outputs; Open drain when programmed for a Wait function, driven High or Low when programmed for a Request function)

These dual-purpose outputs may be programmed as Request lines for a DMA controller or as Wait lines to synchronize the CPU to the ESCC data rate. The reset state is Wait.

Control

A/B

Channel A/Channel B Select (Input)

This signal selects the channel in which the read or write operation occurs.

CE1

Chip Enable (Input; Active Low)

This signal selects the ESCC for a read or write operation.

D/C

Data/Control Select (Input)

This signal defines the type of information transferred to or from the ESCC. A High means data is transferred; a Low indicates a command is transferred.

PIN DESCRIPTION (continued)

Interrupt

$\overline{INT}$
Interrupt Request (Output; Active Low, Open Drain)

This signal is activated when the ESCC requests an interrupt.

$\overline{INTACK}$
Interrupt Acknowledge (Input; Active Low)

This signal indicates an active interrupt acknowledge cycle. When $\overline{RD}$ becomes active, the ESCC places an interrupt vector on the data bus. $\overline{INTACK}$ is latched by the rising edge of PCLK.

Serial Data

RxDa, RxDB
Receive Data (Inputs; Active High)

These input signals receive serial data at standard TTL levels.

TxDA, TxDB
Transmit Data (Outputs; Active High)

These output signals transmit serial data at standard TTL levels.

Miscellaneous

GND
Ground

VDD
+5 Volts

PCLK
Clock (Input)

This is the master ESCC clock used to synchronize internal signals. PCLK is not required to have any phase relationship with the master system clock. PCLK is a TTL-level signal. Maximum transmit rate is 1/4 PCLK.

PIN DESCRIPTION (Continued)

SCSI

Microprocessor Interface Signals

A_0–A_2
Address Lines (Input)

These signals are used with $\overline{CE2}$, $\overline{RD}$ or $\overline{WR}$ to address all internal registers.

$\overline{CE2}$
Chip Enable (Input, Active Low)

$\overline{CE2}$ enables a read or write of the SCSI controller internal register selected by A_0–A_2.

$\overline{DACK}$
DMA Acknowledge (Input, Active Low)

$\overline{DACK}$ resets DRQ and selects the data register for input or output data transfers.

DRQ
DMA Request (Output)

DRQ indicates that the data register is ready to be read or written. DRQ occurs only if DMA mode is TRUE in the Command Register. DRQ is cleared by $\overline{DACK}$.

$\overline{EOP}$
End of Process (Input, Active Low)

$\overline{EOP}$ is used to terminate a DMA transfer. If asserted during a DMA cycle, the current byte will be transferred, but no additional bytes will be requested.

IRQ
Interrupt Request (Output)

IRQ alerts a microprocessor of an error condition or an event completion.

$\overline{RESET}$
Reset (Input, Active Low)

$\overline{RESET}$ clears all the SCSI controller registers. It does not force the SCSI $\overline{RST}$ signal to the active state.

SCSI Interface Signals

The following signals are all bidirectional, active-Low, open-collector signals. With 48-mA sink capability, all pins interface directly with the SCSI Bus.

$\overline{ACK}$
Acknowledge (Input/Output; Open Collector, Active Low)

Driven by an Initiator, $\overline{ACK}$ indicates an acknowledgment for a $\overline{REQ}$/$\overline{ACK}$ data-transfer handshake. In the Target role, $\overline{ACK}$ is received as a response to the $\overline{REQ}$ signal.

$\overline{ATN}$
Attention (Input/Output; Open Collector, Active Low)

Driven by an Initiator, $\overline{ATN}$ indicates an Attention condition. This signal is received in the Target role.

$\overline{BSY}$
Busy (Input/Output; Open Collector, Active Low)

This signal indicates that the SCSI Bus is being used and can be driven by both the Initiator and the Target device.

$\overline{C/D}$
Control/Data (Input/Output; Open Collector, Active Low)

A signal driven by the Target, $\overline{C/D}$ indicates Control or Data information is on the Data Bus. This signal is received by the Initiator.

$\overline{I/O}$
Input/Output (Input/Output; Open Collector, Active Low)

$\overline{I/O}$ is a signal driven by a Target which controls the direction of data movement on the SCSI Bus. TRUE indicates input to the intitiator. This signal is also used to distinguish between Selection and Reselection phases.

$\overline{MSG}$
Message (Input/Output; Open Collector, Active Low)

$\overline{MSG}$ is a signal driven by the Target during the Message phase. This signal is received by the Initiator.

$\overline{REQ}$
Request (Input/Output; Open Collector, Active Low)

Driven by a Target, $\overline{REQ}$ indicates a request for a $\overline{REQ}$/$\overline{ACK}$ data-transfer handshake. This signal is received by the Initiator.

$\overline{RST}$
SCSI Bus RESET (Input/Output; Open Collector, Active Low)

The $\overline{RST}$ Signal indicates an SCSI Bus RESET condition. An internal 30 µA pull up transistor is built-in to prevent spurious interrupts.

$\overline{DB_0}$-$\overline{DB_7}$, $\overline{DBP}$
Data Bits, Parity Bit (Input/Output; Open Collector, Active Low)

These eight data bits ($\overline{DB_0}$-$\overline{DB_7}$), plus a parity bit ($\overline{DBP}$) form the Data Bus. $\overline{DB7}$ is the most significant bit (MSB) and has the highest priority during the Arbitration phase. Data parity is odd. Parity is always generated and optionally checked. Parity is not valid during Arbitration.

$\overline{SEL}$
Select (Input/Output; Open Collector, Active Low)

$\overline{SEL}$ is used by an Initiator to select a Target, or by a Target to reselect an Initiator.

PIN DESCRIPTION (continued)

Common CPU Interface Signals

$\overline{RD}$
Read (Input; Active Low)

This signal indicates a read operation and when Am85C80 is selected, enables the Data Bus Drivers. During the interrupt acknowledge cycle for ESCC, this signal gates the interrupt vector onto the bus if the ESCC is the highest priority device requesting an interrupt. It also selects the SCSI input data register when used with $\overline{DACK}$.

$\overline{WR}$
Write (Input; Active Low)

When Am85C80 is selected, along with either $\overline{CE1}$ or $\overline{CE2}$, this signal indicates a write operation. The coincidence of $\overline{RD}$ and $\overline{WR}$ is interpreted as a reset for ESCC. It also selects the SCSI output data register when used with $\overline{DACK}$.

D_0–D_7
Data (Input/Output; Active High)

Common data lines for ESCC and SCSI. These lines transfer data into and out of the Am85C80.

SCSI FUNCTIONAL DESCRIPTION

General

The Small Computer Systems Interface (SCSI) device appears as a set of eight registers to the controlling CPU. By reading and writing the appropriate registers, the CPU may initiate any SCSI Bus activity or may sample and assert any signal on the SCSI Bus. This allows the user to implement all or portions of the SCSI protocol in software. These registers are read (written) by activating $\overline{CE2}$ with an address on A_0–A_2 and then issuing an $\overline{RD}$ ($\overline{WR}$) pulse. This section describes the operation of the internal registers.

Table 1. Register Summary

Address A_2	A_1	A_0	R/W	Register Name
0	0	0	R	Current SCSI Data
0	0	0	W	Output Data
0	0	1	R/W	Initiator Command
0	1	0	R/W	Mode
0	1	1	R/W	Target Command
1	0	0	R	Current SCSI Bus Status
1	0	0	W	Select Enable
1	0	1	R	Bus and Status
1	0	1	W	Start DMA Send
1	1	0	R	Input Data
1	1	0	W	Start DMA Target Receive
1	1	1	R	Reset Parity/Interrupts
1	1	1	W	Start DMA Initiator Receive

Data Registers

The data registers are used to transfer SCSI commands, data, status, and message bytes between the microprocessor Data Bus and the SCSI Bus. The Am85C80 does not interpret any information that passes through the data registers. The data registers consist of the transparent Current SCSI Data Register, the Output Data Register, and the Input Data Register. User note: a one in a register means the pin is Low. For example, when the $\overline{ATN}$ Pin is Low, $\overline{ATN}$ in the bus and status register will be High.

Current SCSI Data Register—Address 0 (Read Only)

The Current SCSI Data Register is a read-only register that allows the microprocessor to read the active SCSI Data Bus. This is accomplished by activating $\overline{CE2}$ with an address on A_2–A_0 of 000 and issuing an $\overline{RD}$ pulse. If parity checking is enabled, the SCSI Bus parity is checked at the beginning of the read cycle. This register is used during a programmed I/O data read or during Arbitration to check for higher priority arbitrating devices. Parity is not guaranteed valid during Arbitration.

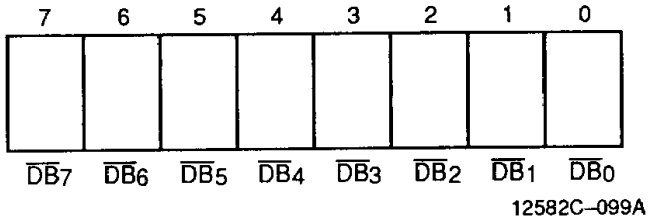


Figure 1. Current SCSI Data Register

Output Data Register—Address 0 (Write Only)

The Output Data Register is a write-only register that is used to send data to the SCSI Bus. This is accomplished by either using a normal CPU write, or under DMA control, by using $\overline{WR}$ and $\overline{DACK}$. This register is also used to assert the proper ID bits or the SCSI Bus during the Arbitration and Selection phases.

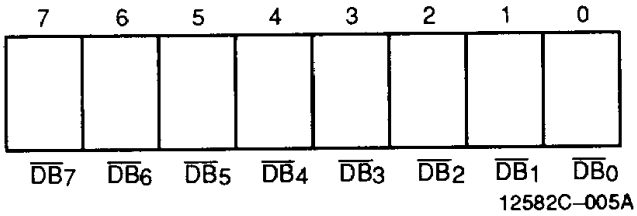


Figure 2. Output Data Register

Input Data Register—Address 6 (Read Only)

The Input Data Register is a read-only register that is used to read latched data from the SCSI Bus. Data is latched either during a DMA Target receive operation when $\overline{ACK}$ goes active or during a DMA Initiator receive when $\overline{REQ}$ goes active. The DMA Mode bit (port 2, bit 1) must be set before data can be latched in the Input Data Register. This register may be read under DMA control using $\overline{RD}$ and $\overline{DACK}$. Parity is optionally checked when the Input Data Register is loaded.

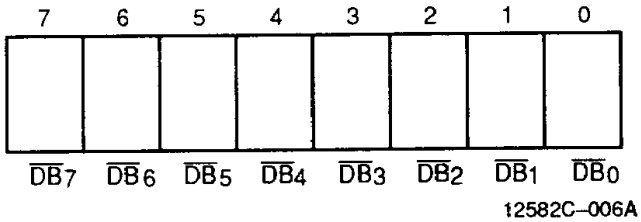


Figure 3. Input Data Register

Initiator Command Register—Address 1 (Read/Write)

The Initiator Command Register is a read/write register that is used to assert certain SCSI Bus signals, to monitor those signals, and to monitor the progress of bus arbitration. Many of these bits are significant only when being used as Initiators; however, most can be used during Target role operation.

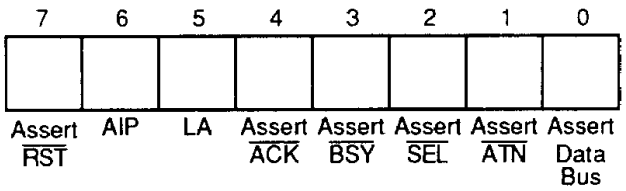


Figure 4-1. Initiator Command Register—Register Read

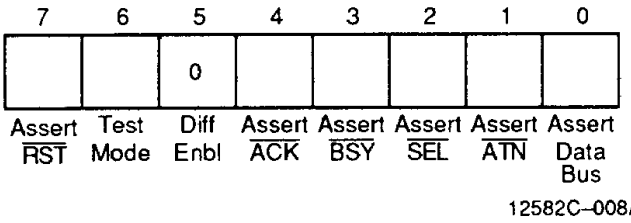


Figure 4-2. Initiator Command Register—Register Write

The following describes the operation of all bits in the Initiator Command Register.

Bit 7—Assert $\overline{RST}$

Whenever a one is written to bit 7 of the Initiator Command Register, the $\overline{RST}$ signal is asserted on the SCSI Bus. The $\overline{RST}$ signal will remain asserted until this bit is reset or until an external $\overline{RESET}$ occurs. After this bit is set (1), IRQ goes active and all internal logic and control registers are reset (except for the interrupt latch and the Assert $\overline{RST}$ bit). Writing a zero to bit 7 of the Initiator Command Register de-asserts the $\overline{RST}$ signal. Reading this register simply reflects the status of this bit.

Bit 6—AIP (Arbitration in Progress) (Read Bit)

This bit is used to determine if Arbitration is in progress. For this bit to be active, the Arbitrate bit (port 2, bit 0) must have been set previously. It indicates that a Bus-Free condition has been detected and that the chip has asserted $\overline{BSY}$ and the contents of the Output Data Register (port 0) onto the SCSI Bus. AIP will remain active until the Arbitrate bit is reset.

Bit 6—Test Mode (Write Bit)

This bit may be written during a test environment to disable all output drivers, effectively removing the Am85C80 from the circuit. Resetting this bit returns the part to normal operation.

Bit 5—LA (Lost Arbitration) (Read Bit)

This bit, when active, indicates that the Am85C80 detected a Bus-Free condition, arbitrated for use of the bus by asserting $\overline{BSY}$ and its ID on the Data Bus, and lost Arbitration due to $\overline{SEL}$ being asserted by another bus device. For this bit to be active, the Arbitrate bit (port 2, bit 0) must be active.

Bit 5—Diff Enbl (Differential Enable) (Write Bit)

This bit should be written with a zero for proper operation.

Bit 4—Assert $\overline{ACK}$

This bit is used by the bus initiator to assert $\overline{ACK}$ on the SCSI Bus. In order to assert $\overline{ACK}$, the Targetmode bit (port 2, bit 6) must be False. Writing a zero to this bit resets $\overline{ACK}$ on the SCSI Bus. Reading this register simply reflects the status of this bit.

Bit 3—Assert $\overline{BSY}$

Writing a one (1) into this bit position asserts $\overline{BSY}$ onto the SCSI Bus. Conversely, a zero resets the $\overline{BSY}$ signal. Asserting $\overline{BSY}$ indicates a successful selection or reselection and resetting this bit creates a Bus-Disconnect condition. Reading this register simply reflects the status of this bit.

Bit 2—Assert $\overline{SEL}$

Writing a one (1) into this bit position asserts $\overline{SEL}$ onto the SCSI Bus. $\overline{SEL}$ is normally asserted after Arbitration has been successfully completed. $\overline{SEL}$ may be de-asserted by resetting this bit to a zero. A read of this register simply reflects the status of this bit.

Bit 1—Assert $\overline{ATN}$

$\overline{ATN}$ may be asserted on the SCSI Bus by setting this bit to a one (1) if the Targetmode bit (port 2, bit 6) is False. $\overline{ATN}$ is normally asserted by the initiator to request a Message Out bus phase. Note that since Assert $\overline{SEL}$ and Assert $\overline{ATN}$ are in the same register, a select with $\overline{ATN}$ may be implemented with one CPU write. $\overline{ATN}$ may be de-asserted by resetting this bit to zero. A read of this register simply reflects the status of this bit.

Bit 0—Assert Data Bus

The Assert Data Bus bit, when set, allows the contents of the Output Data Register to be enabled as chip outputs on the signals $\overline{DB_0}$–$\overline{DB_7}$. Parity is also generated and asserted on $\overline{DBP}$.

When connected as an initiator, the outputs are only enabled if the Targetmode bit (port 2, bit 6) is False, the received signal $\overline{I/O}$ is False, and the phase signals ($\overline{C/D}$, $\overline{I/O}$, and $\overline{MSG}$) match the contents of the Assert $\overline{C/D}$, Assert $\overline{I/O}$, and Assert $\overline{MSG}$ in the Target Command Register.

This bit should also be set during DMA send operations.

Mode Register—Address 2 (Read/Write)

The Mode Register is used to control the operation of the chip. This register determines whether the Am85C80 operates as an Initiator or a Target, whether DMA transfers are being used, whether parity is checked, and whether interrupts are generated on various external conditions. This register may be read to check the value of these

internal control bits. Figure 5 describes the operation of these control bits.

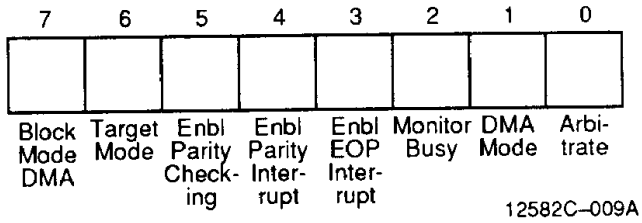


Figure 5. Mode Register

Bit 7—Block Mode DMA

The Block Mode DMA bit controls the characteristics of the DMA DRQ-DACK handshake. When this bit is reset (0) and the DMA Mode bit is active (1), the DMA handshake uses the normal interlocked handshake, and the rising edge of DACK indicates the end of each byte being transferred. In block mode operations, Block Mode DMA bit set (1), and DMA Mode bit set (1), the end of RD or WR signifies the end of each byte transferred and DACK is allowed to remain active throughout the DMA operation. Ready can then be used to request the next transfer.

Bit 6—Target Mode

The Targetmode bit allows the Am85C80 to operate as either an SCSI Bus Initiator, bit reset (0), or as an SCSI Bus Target device, bit set (1). In order for the signals ATN and ACK to be asserted on the SCSI Bus, the Targetmode bit must be reset (0). In order for the signals C/D, I/O, MSG, and REQ to be asserted on the SCSI Bus, the Targetmode bit must be set (1).

Bit 5—Enable Parity Checking

The Enable Parity Checking bit determines whether parity errors will be ignored or saved in the parity error latch. If this bit is reset (0), parity will be ignored. Conversely, if this bit is set (1), parity errors will be saved.

Bit 4—ENABLE PARITY INTERRUPT

The Enable Parity Interrupt bit, when set (1), will cause an interrupt (IRQ) to occur if a parity error is detected. A parity interrupt will only be generated if the Enable Parity Checking bit (bit 5) is also enabled (1).

Bit 3—Enable EOP Interrupt

The Enable EOP Interrupt bit, when set (1), causes an interrupt to occur when the EOP (End of Process) signal is received from the DMA controller logic.

Bit 2—Monitor Busy

The Monitor Busy bit, when True (1), causes an interrupt to be generated for an unexpected loss of BSY. When the interrupt is generated due to loss of BSY, the lower six bits of the Initiator Command Register are reset (0) and all signals are removed from the SCSI Bus.

Bit 1—DMA Mode

The DMA Mode bit is normally used to enable a DMA transfer and must be set (1) prior to writing ports 5 through 7. Ports 5 through 7 are used to start DMA transfers. The Targetmode bit (port 2, bit 6) must be consistent with writes to port 6 and 7 [i.e., set (1) for a write to port 6 and reset (0) for a write to port 7]. The control bit Assert Data Bus (port 1, bit 0) must be True (1) for all DMA send operations. In the DMA mode, REQ and ACK are automatically controlled.

The DMA Mode bit is not reset upon the receipt of an EOP signal. Any DMA transfer may be stopped by writing a zero into this bit location; however, care must be taken not to cause CE2 and DACK to be active simultaneously.

Bit 0—Arbitrate

The Arbitrate bit is set (1) to start the Arbitration process. Prior to setting this bit, the Output Data Register should contain the proper SCSI device ID value. Only one data bit should be active for SCSI Bus Arbitration. The Am85C80 will wait for a Bus-Free condition before entering the Arbitration phase. The results of the Arbitration phase may be determined by reading the status bits LA and AIP (port 1, bits 5 and 6, respectively).

Target Command Register—Address 3 (Read/Write)

When connected as a target device, the Target Command Register allows the CPU to control the SCSI Bus Information Transfer phase and/or to assert REQ simply by writing this register. The Targetmode bit (port 2, bit 6) must be True (1) for bus assertion to occur. The SCSI Bus phases are described in Table 2.

Table 2. SCSI Information Transfer Phases

Bus Phase	Assert I/O	Assert C/D	Assert MSG
Data Out	0	0	0
Unspecified	0	0	1
Command	0	1	0
Message Out	0	1	1
Data In	1	0	0
Unspecified	1	0	1
Status	1	1	0
Message In	1	1	1

When connected as an Initiator with DMA Mode True, if the phase lines (I/O, C/D, and MSG) do not match the phase bits in the Target Command Register, a phase-mismatch interrupt is generated when REQ goes active. In order to send data as an Initiator, the Assert I/O, Assert C/D, and Assert MSG bits must match the corresponding bits in the Current SCSI Bus Status Register (port 4). The Assert REQ bit (bit 3) has no meaning when operating as an Initiator.

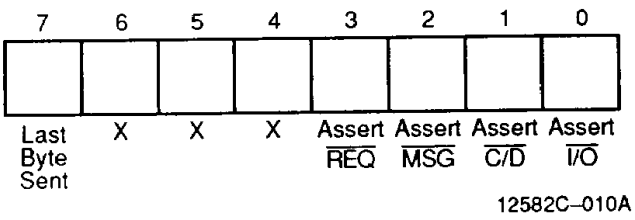


Figure 6. Target Command Register

The Am85C80 uses bit 7 of the SCSI Target Command Register to determine when the last byte of a DMA

transfer is sent to the SCSI bus. This flag is necessary since the end of DMA bit in the Bus and Status Register only indicates when the last byte was received from the DMA.

Current SCSI Bus Status Register—Address 4 (Read Only)

The Current SCSI Bus Status Register is a read-only register that is used to monitor seven SCSI Bus control signals plus the Data Bus parity bit. For example, an Initiator device can use this register to determine the current bus phase and to poll $\overline{REQ}$ for pending data transfers. This register may also be used to determine why a particular interrupt occurred. Figure 7 describes the Current SCSI Bus Status Register.

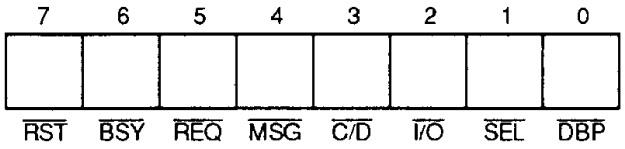


Figure 7. Current SCSI Bus Status Register

The Am85C80 uses bit 7 of this register to determine when the last byte of a DMA transfer is sent to the SCSI bus. This flag is necessary since the end of DMA bit in the Bus and Status Register only reflects when the last byte was received from the DMA.

Select Enable Register—Address 4 (Write Only)

The Select Enable Register is a write-only register which is used as a mask to monitor a signal ID during a selection attempt. The simultaneous occurrence of the correct ID bit, $\overline{BSY}$ False, and $\overline{SEL}$ True will cause an interrupt. This interrupt can be disabled by resetting all bits in this register. If the Enable Parity Checking bit (port 2, bit 5) is active (1), parity will be checked during selection.

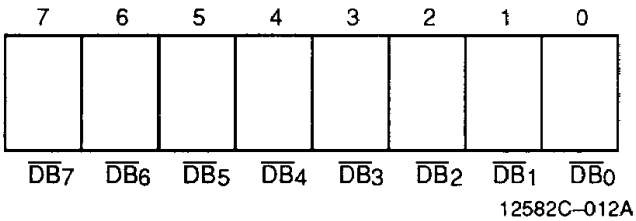


Figure 8. Select Enable Register

Bus and Status Register—Address 5 (Read Only)

The Bus and Status Register is a read-only register that can be used to monitor the remaining SCSI control signals not found in the Current SCSI Bus Status Register ($\overline{ATN}$ and $\overline{ACK}$), as well as six other status bits. Individual descriptions of each bit of the Bus and Status Register follow.

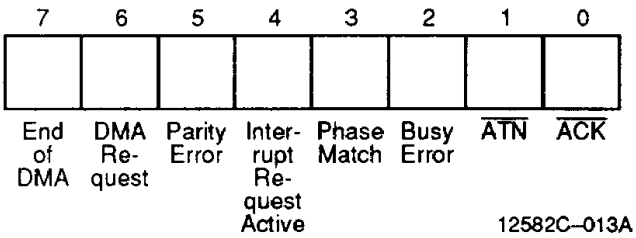


Figure 9. Bus and Status Register

Bit 7—End of DMA Transfer

The End Of DMA Transfer bit is set if $\overline{EOP}$, $\overline{DACK}$, and either $\overline{RD}$ or $\overline{WR}$ are simultaneously active for at least 100 ns. Since the $\overline{EOP}$ signal can occur during the last byte sent to the Output Data Register (port 0), the $\overline{REQ}$ and $\overline{ACK}$ signals should be monitored to ensure that the last byte has been transferred. This bit is reset when the DMA Mode bit is reset (0) in the Mode Register (port 2).

Bit 6—DMA Request

The DMA Request bit allows the CPU to sample the output pin DRQ. DRQ can be cleared by asserting $\overline{DACK}$ or by resetting the DMA Mode bit (bit 1) in the Mode Register (port 2). The DRQ signal does not reset when a phase-mismatch interrupt occurs.

Bit 5—Parity Error

This bit is set if a parity error occurs during a data receive or a device selection. The Parity Error bit can only be set (1) if the Enable Parity Check bit (port 2, bit 5) is active (1). This bit may be cleared by reading the Reset Parity/Interrupt Register (port 7).

Bit 4—Interrupt Request Active

This bit is set if an enabled interrupt condition occurs. It reflects the current state of the IRQ output and can be cleared by reading the Reset Parity/Interrupt Register (port 7).

Bit 3—Phase Match

The SCSI signals, $\overline{MSG}$, $\overline{C/D}$, and $\overline{I/O}$, represent the current Information Transfer phase. The Phase Match bit indicates whether the current SCSI Bus phase matches the lower 3 bits of the Target Command Register. Phase Match is continuously updated and is only significant when operating as a Bus Initiator. A phase match is required for data transfers to occur on the SCSI Bus.

Bit 2—Busy Error

The Busy Error bit is active if an unexpected loss of the $\overline{BSY}$ signal has occurred. This latch is set whenever the Monitor Busy bit (port 2, bit 2) is True and $\overline{BSY}$ is False. An unexpected loss of $\overline{BSY}$ will disable any SCSI outputs and will reset the DMA MODE bit (port 2, bit 1).

Bit 1—$\overline{ATN}$

This bit reflects the condition of the SCSI Bus control signal $\overline{ATN}$. This signal is normally monitored by the Target device.

Bit 0—$\overline{ACK}$

This bit reflects the condition of the SCSI Bus control signal $\overline{ACK}$. This signal is normally monitored by the Target device.

DMA Registers

Three write-only registers are used to initiate all DMA activity. They are Start DMA Send (port 5), Start DMA Target Receive (port 6), and Start DMA Initiator Receive (port 7). Simply writing these registers starts the DMA transfers. Data presented to the Am85C80 on signals D_0–D_7 during the register write is meaningless and has no effect on the operation. Prior to writing these registers, the Block Mode DMA bit (bit 7), the DMA Mode bit (bit 1) and the Targetmode bit (bit 6) in the Mode Register (port 2) must be appropriately set. The individual registers are briefly described as follows.

Start DMA Send—Address 5 (Write Only)

This register is written to initiate a DMA send, from the DMA to the SCSI Bus, for either Initiator or Target role operations. The DMA Mode bit (port 2, bit 1) must be set prior to writing this register.

Start DMA Target Receive—Address 6 (Write Only)

This register is written to initiate a DMA receive—from the SCSI Bus to the DMA—for Target operation only. The DMA Mode bit (bit 1) and the Targetmode bit (bit 6) in the Mode Register (port 2) must both be set (1) prior to writing this register.

Start DMA Initiator Receive—Address 7 (Write Only)

This register is written to initiate a DMA receive—from the SCSI Bus to the DMA, for Initiator operation only. The DMA Mode bit (bit 6) must be False (0) in the Mode Register (port 2) prior to writing this register.

Reset Parity/Interrupt—Address 7 (Read Only)

Reading this register resets the Parity Error bit (bit 5), the Interrupt Request bit (bit 4), and the Busy Error bit (bit 2) in the Bus and Status Register (port 5).

On-Chip SCSI Hardware Support

The Am85C80 is easy to use because of its simple architecture. The chip allows direct control and monitoring of the SCSI Bus by providing a latch for each signal. However, portions of the protocol define timings that are much too quick for traditional microprocessors to control. Therefore, hardware support has been provided for DMA transfers, bus arbitration, phase-change monitoring, bus disconnection, bus reset, parity generation, parity checking, and device selection/reselection.

Arbitration is accomplished using a Bus-Free filter to continuously monitor $\overline{BSY}$. If $\overline{BSY}$ remains inactive for at least 400 ns, then the SCSI Bus is considered free and Arbitration may begin. Arbitration will begin if the bus is free, $\overline{SEL}$ is inactive, and the Arbitration bit (port 2, bit 0) is active. Once arbitration has begun ($\overline{BSY}$ asserted), an arbitration delay of 2.2 µs must elapse before the Data Bus can be examined to determine if Arbitration has been won. This delay must be implemented in the controlling software driver.

The Am85C80 has no clock. Delays such as bus-free delay, bus-set delay, and bus-settle delay are implemented using gate delays. These delays may differ between devices because of inherent process variations, but are well within the proposed ANSI X3T9.2 specification.

Interrupts

The Am85C80 provides an interrupt output (IRQ) to indicate a task completion or an abnormal bus occurrence. The use of interrupts is optional and may be disabled by resetting the appropriate bits in the Mode Register (port 2) or the Select Enable Register (port 4).

When an interrupt occurs, the Bus and Status Register and the Current SCSI Bus Status Register must be read to determine which condition created the interrupt. IRQ can be reset simply by reading the Reset Parity/Interrupt Register (port 7) or by an external chip reset ($\overline{RESET}$ active for 100 ns).

Assuming the Am85C80 has been properly initialized, an interrupt will be generated if the chip is selected or reselected, if an $\overline{EOP}$ signal occurs during a DMA transfer, if a SCSI Bus reset occurs, if a parity error occurs during a data transfer, if a bus phase mismatch occurs, or if a SCSI Bus disconnection occurs.

Selection/Reselection

The Am85C80 can generate a select interrupt if $\overline{SEL}$ is True (1), its device ID is True (1), and $\overline{BSY}$ is False for at least a bus-settle delay (400 ns). If $\overline{I/O}$ is active, this should be considered a reselect interrupt. The correct ID bit is determined by a match in the Select Enable Register (port 4). Only a single bit match is required to generate an interrupt. This interrupt may be disabled by writing zeros into all bits of the Select Enable Register.

If parity is supported, parity should also be good during the selection phase. Therefore, if the Enable Parity bit (port 2, bit 5) is active, then the Parity Error bit should be checked to ensure that a proper selection has occurred. The Enable Parity Interrupt bit need not be set for this interrupt to be generated.

The proposed SCSI specification also requires that no more than two device IDs be active during the selection process. To ensure this, the Current SCSI Data Register (port 0) should be read.

The proper values for the Bus and Status Register (port 5) and the Current SCSI Bus Status Register (port 4) for Selection/Reselection interrupt are displayed in Figures 10 and 11, respectively.

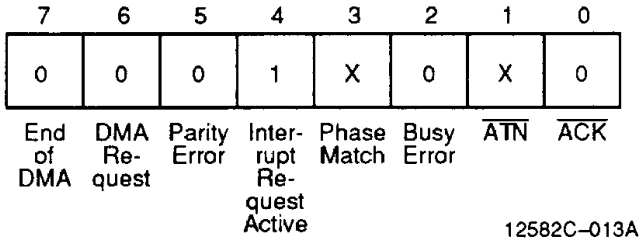


Figure 10. Condition of Bus and Status Register for Select Interrupt to Occur

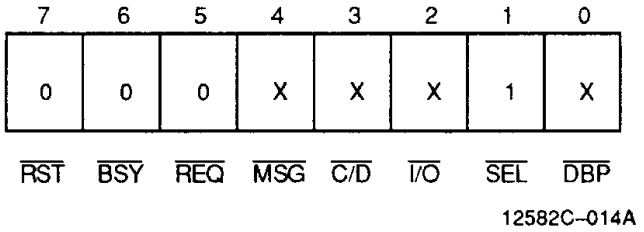


Figure 11. Condition of Current SCSI Bus Status Register for Select Interrupt to Occur

End of Process (EOP) Interrupt

An End of Process signal ($\overline{EOP}$) that occurs during a DMA transfer (DMA Mode True) will set the End Of DMA Status bit (port 5, bit 7) and will optionally generate an interrupt if the Enable $\overline{EOP}$ Interrupt bit (port 2, bit 3) is True. The $\overline{EOP}$ pulse will not be recognized (End Of DMA bit set) unless $\overline{EOP}$, $\overline{DACK}$, and either $\overline{RD}$ or $\overline{WR}$ are concurrently active for at least 100 ns. DMA transfers can still occur if $\overline{EOP}$ was not asserted at the correct time. This interrupt can be disabled by resetting the Enable $\overline{EOP}$ Interrupt bit (port 2, bit 3).

The proper values for the Bus and Status Register (port 5) and the Current SCSI Bus Status Register (port 4) for this interrupt are shown in Figures 12 and 13, respectively.

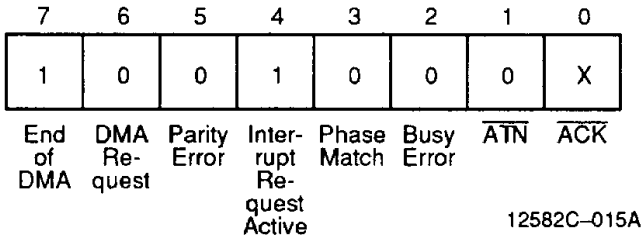


Figure 12. Condition of Bus and Status Register for End of Process Interrupt to Occur

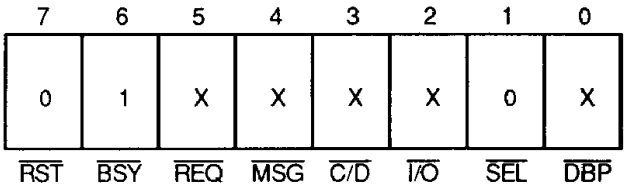


Figure 13. Condition of Current SCSI Bus Status Register for End of Process Interrupt to Occur

The End Of DMA bit (port 5, bit 7) is used to determine when a block transfer is complete. Receive operations are complete when there are no data left in the chip and no additional handshakes occurring. The only exception to this is receiving data as an Initiator when the Target opts to send additional data for the same phase. In this case, $\overline{REQ}$ goes active and the new data is present in the Input Data Register (port 6). Since a phase-mismatch interrupt will not occur, $\overline{REQ}$ and $\overline{ACK}$ need to be sampled to determine that the Target is attempting to send more data.

For send operations, the End Of DMA bit (port 5, bit 7) is set when the DMA finishes its transfer, but the SCSI transfer may still be in progress. If connected as a Target, $\overline{REQ}$ and $\overline{ACK}$ should be sampled until both are False. If connected as an Initiator, a phase change interrupt can be used to signal the completion of the previous phase. It is possible for the Target to request additional data for the same phase. In this case, a phase change will not occur and both $\overline{REQ}$ and $\overline{ACK}$ must be sampled to determine when the last byte was transferred.

SCSI Bus Reset

The Am85C80 generates an interrupt when the $\overline{RST}$ signal transitions to True. The device releases all bus signals within a bus-clear delay (800 ns) of this transition. This interrupt also occurs after setting the Assert $\overline{RST}$ bit (port 1, bit 7). This interrupt cannot be disabled. (Note: $\overline{RST}$ is not latched in bit 7 of the Current SCSI Bus Status Register (port 4) and may not be active when this port is read. For this case, the Bus Reset interrupt may be determined by default.)

The proper values for the Bus and Status Register (port 5) and the Current SCSI Bus Status Register (port 4) for this interrupt are displayed in Figures 14 and 15, respectively.

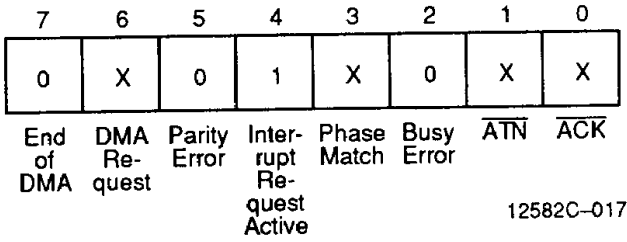


Figure 14. Condition of Bus and Status Register for SCSI Bus Reset to Occur

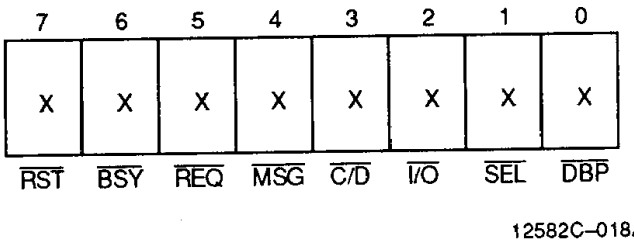


Figure 15. Condition of Current SCSI Bus Status Register for SCSI Bus Reset to Occur

Parity Error

An interrupt is generated for a received parity error if the Enable Parity Check (bit 5) and the Enable Parity Interrupt (bit 4) bits are set (1) in the Mode Register (port 2). Parity is checked during a read of the Current SCSI Data Register (port 0) and during a DMA receive operation. A parity error can be detected without generating an interrupt by disabling the Enable Parity Interrupt bit (port 2, bit 4) and checking the Parity Error flag (port 5, bit 5).

The proper values for the Bus and Status Register (port 5) and the Current SCSI Bus Status Register (port 4) for Parity Error interrupt are displayed in Figures 16 and 17, respectively.

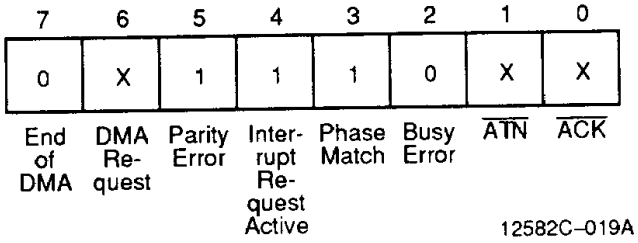


Figure 16. Condition of Bus and Status Register for Parity Error Interrupt to Occur

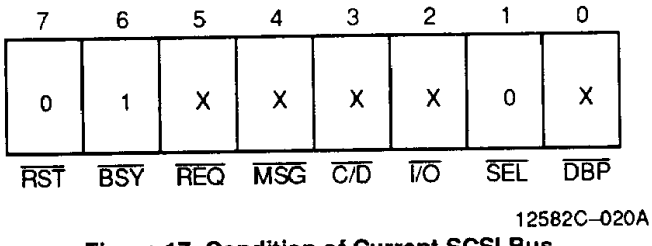


Figure 17. Condition of Current SCSI Bus Status Register for Parity Error Interrupt to Occur

Bus Phase Mismatch

The SCSI phase lines are composed of the signals $\overline{I/O}$, $\overline{C/D}$, and $\overline{MSG}$. These signals are compared with the corresponding bits in the Target Command Register: Assert $\overline{I/O}$ (bit 0), Assert $\overline{C/D}$ (bit 1), and Assert $\overline{MSG}$ (bit 2). The comparison occurs continually and is reflected in the Phase Match bit (bit 3) of the Bus and Status Register (port 5). If the DMA Mode bit (port 2, bit 1) is active and a phase mismatch occurs when $\overline{REQ}$ changes from False to True, an interrupt (IRQ) is generated.

A phase mismatch prevents the recognition of $\overline{REQ}$ and removes the chip from the bus during an initiator send operation [$\overline{DB_0}$–$\overline{DB_7}$ and $\overline{DBP}$ will not be driven even though the Assert Data Bus bit (port 1, bit 0) is active]. This interrupt is only significant when connected as an initiator and may be disabled by resetting the DMA Mode bit. (Note: It is possible for this interrupt to occur when connected as a Target if another device is driving the phase lines to a different state.)

The proper values for the Bus and Status Register (port 5) and the Current SCSI Bus Status Register (port 4) for Bus Phase Mismatch interrupt to occur are displayed in Figures 18 and 19, respectively.

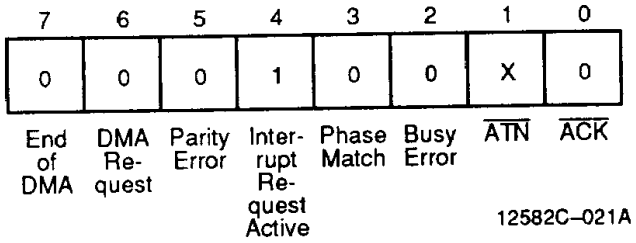


Figure 18. Condition of Bus and Status Register for Bus Phase Mismatch Interrupt to Occur

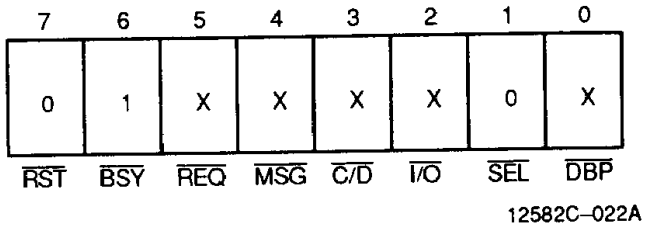


Figure 19. Condition of Current SCSI Bus Status Register for Bus Phase Mismatch Interrupt to Occur

Loss of $\overline{BSY}$

If the Monitor Busy bit (bit 2) in the Mode Register (port 2) is active, an interrupt will be generated if the $\overline{BSY}$ signal goes False for at least a bus-settle delay (400 ns). This interrupt may be disabled by resetting the Monitor Busy bit. Register values are displayed in Figures 20 and 21.

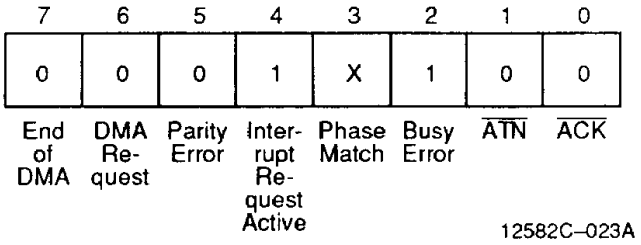


Figure 20. Bus and Status Register

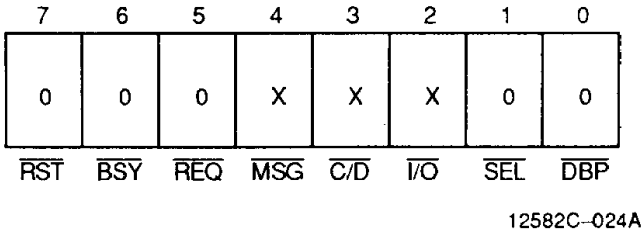


Figure 21. Current SCSI Bus Status Register

Reset Conditions

Three possible reset situations exist with the Am85C80.

Hardware Chip Reset

When the signal $\overline{RST}$ is active for at least 200 ns, the SCSI device is re-initialized and all internal logic and control registers of SCSI are cleared. This is a chip reset only and does not create an SCSI Bus-Reset condition.

SCSI Bus Reset ($\overline{RST}$) Received

When an SCSI $\overline{RST}$ signal is received, an IRQ interrupt is generated and a SCSI chip reset is performed. All internal logic and registers are cleared, except for the IRQ interrupt latch and the Assert $\overline{RST}$ bit (bit 7) in the Initiator Command Register (port 1). (Note: The $\overline{RST}$ signal may be sampled by reading the Current SCSI Bus Status Register (port 4); however, this signal is not latched and may not be present when this port is read.)

SCSI Bus Reset ($\overline{RST}$) Issued

If the CPU sets the Assert $\overline{RST}$ bit (bit 7) in the Initiator Command Register (port 1), the $\overline{RST}$ signal goes active on the SCSI Bus and an internal reset is performed. Again, all internal logic and registers are cleared except for the IRQ interrupt latch and the Assert $\overline{RST}$ bit (bit 7) in the Initiator Command Register (port 1). The $\overline{RST}$ signal will continue to be active until the Assert $\overline{RST}$ bit is reset or until a hardware reset occurs.

Data Transfers

Data may be transferred between SCSI Bus devices in one of four modes: (1) Programmed I/O, (2) Normal DMA, (3) Block Mode DMA, or (4) Pseudo DMA. The following sections describe these modes in detail. (Note: For all data transfer operations, $\overline{DACK}$ and $\overline{CE2}$ should never be active simultaneously.)

Programmed I/O Transfers

Programmed I/O is the most primitive form of data transfer. The $\overline{REQ}$ and $\overline{ACK}$ handshake signals are individually monitored and asserted by reading and writing the appropriate register bits. This type of transfer is normally used when transferring small blocks of data, such as command blocks or message and status bytes.

An Initiator send operation would begin by setting the $\overline{C/D}$, $\overline{I/O}$, and $\overline{MSG}$ bits in the Target Command Register to the correct state so that a phase match exists. In addition to the phase match condition, it is necessary for the Assert Data Bus bit (port 1, bit 0) to be True and the received I/O signal to be False for the Am85C80 to send data.

For each transfer, the data is loaded into the Output Data Register (port 0). The CPU then waits for the $\overline{REQ}$ bit (port 4, bit 5) to become active. Once $\overline{REQ}$ goes active, the Phase Match bit (port 5, bit 3) is checked and the Assert $\overline{ACK}$ bit (port 1, bit 4) is set. The $\overline{REQ}$ bit is sampled until it becomes False and the CPU resets the Assert $\overline{ACK}$ bit to complete the transfer.

Normal DMA Mode

DMA transfers are normally used for large block transfers. The SCSI chip outputs a DMA request (DRQ) whenever it is ready for a byte transfer. External DMA logic uses this DRQ signal to generate $\overline{DACK}$ and an $\overline{RD}$ or an $\overline{WR}$ pulse to the Am85C80. DRQ goes inactive when $\overline{DACK}$ is asserted, and $\overline{DACK}$ goes inactive some time after the minimum read or write pulse width. This process is repeated for every byte. For this mode, $\overline{DACK}$ should not be allowed to cycle unless a transfer is taking place.

Block Mode DMA

Some popular DMA controllers such as the Am9517A provide a Block Mode DMA transfer. This type of transfer allows the DMA controller to transfer blocks of data without relinquishing the use of the Data Bus to the CPU after each byte is transferred; thus, faster transfer rates are achieved by eliminating the repetitive access and release of the CPU Bus.

If the Block Mode DMA bit (port 2, bit 7) is active, the Am85C80 will begin the transfer by asserting DRQ. The DMA controller then asserts $\overline{DACK}$ for the remainder of the block transfer. DRQ goes inactive for the duration of the transfer.

Non–Block Mode DMA transfers end when $\overline{DACK}$ goes False, whereas Block mode transfers end when $\overline{RD}$ or $\overline{WR}$ becomes inactive. Since this is the case, DMA transfers may be started sooner in a Block Mode transfer.

The methods described under "Halting a DMA Operation" apply for all DMA operations.

Pseudo DMA Mode

To avoid the tedium of monitoring and asserting the request/acknowledge handshake signals for programmed I/O transfers, the system may be designed to implement a pseudo DMA mode. This mode is implemented by programming the Am85C80 to operate in the DMA mode, but using the CPU to emulate the DMA handshake. DRQ may be detected by polling the DMA Request bit (bit 6) in the Bus and Status Register (port 5), by sampling the signal through an external port, or by using it to generate a CPU interrupt. Once DRQ is detected, the CPU can perform a read or write data transfer. This CPU read/write is externally decoded to generate the appropriate $\overline{DACK}$ and $\overline{RD}$ or $\overline{WR}$ signals.

Often, external decoding logic is necessary to generate the Am85C80 $\overline{CE2}$ signal. This same logic may be used to generate $\overline{DACK}$ at no extra system cost and provide an increased performance in programmed I/O transfers.

Halting a DMA Operation

The $\overline{EOP}$ signal is not the only way to halt a DMA transfer. A bus phase mismatch or a reset of the DMA Mode bit (port 2, bit 1) can also terminate a DMA cycle for the current bus phase.

Using the $\overline{EOP}$ Signal

If $\overline{EOP}$ is used, it should be asserted for at least 100 ns while $\overline{DACK}$ and $\overline{RD}$ or $\overline{WR}$ are simultaneously active. Note, however, that if $\overline{RD}$ or $\overline{WR}$ is not active an interrupt will be generated, but the DMA activity will continue. The $\overline{EOP}$ signal does not reset the DMA Mode bit. Since the $\overline{EOP}$ signal can occur during the last byte sent to the Output Data Register (port 0), the $\overline{REQ}$ and $\overline{ACK}$ signals should be monitored to ensure that the last byte has transferred.

Bus Phase Mismatch Interrupt

A bus phase mismatch interrupt may be used to halt the transfer if operating as an Initiator. Using this method frees the host from maintaining a data length counter and frees the DMA logic from providing the $\overline{EOP}$ signal. If performing an Initiator send operation, the Am85C80 requires $\overline{DACK}$ to cycle before $\overline{ACK}$ goes inactive. Since phase changes cannot occur if $\overline{ACK}$ is active, either $\overline{DACK}$ must be cycled after the last byte is sent or the DMA Mode bit must be reset in order to receive the phase mismatch interrupt.

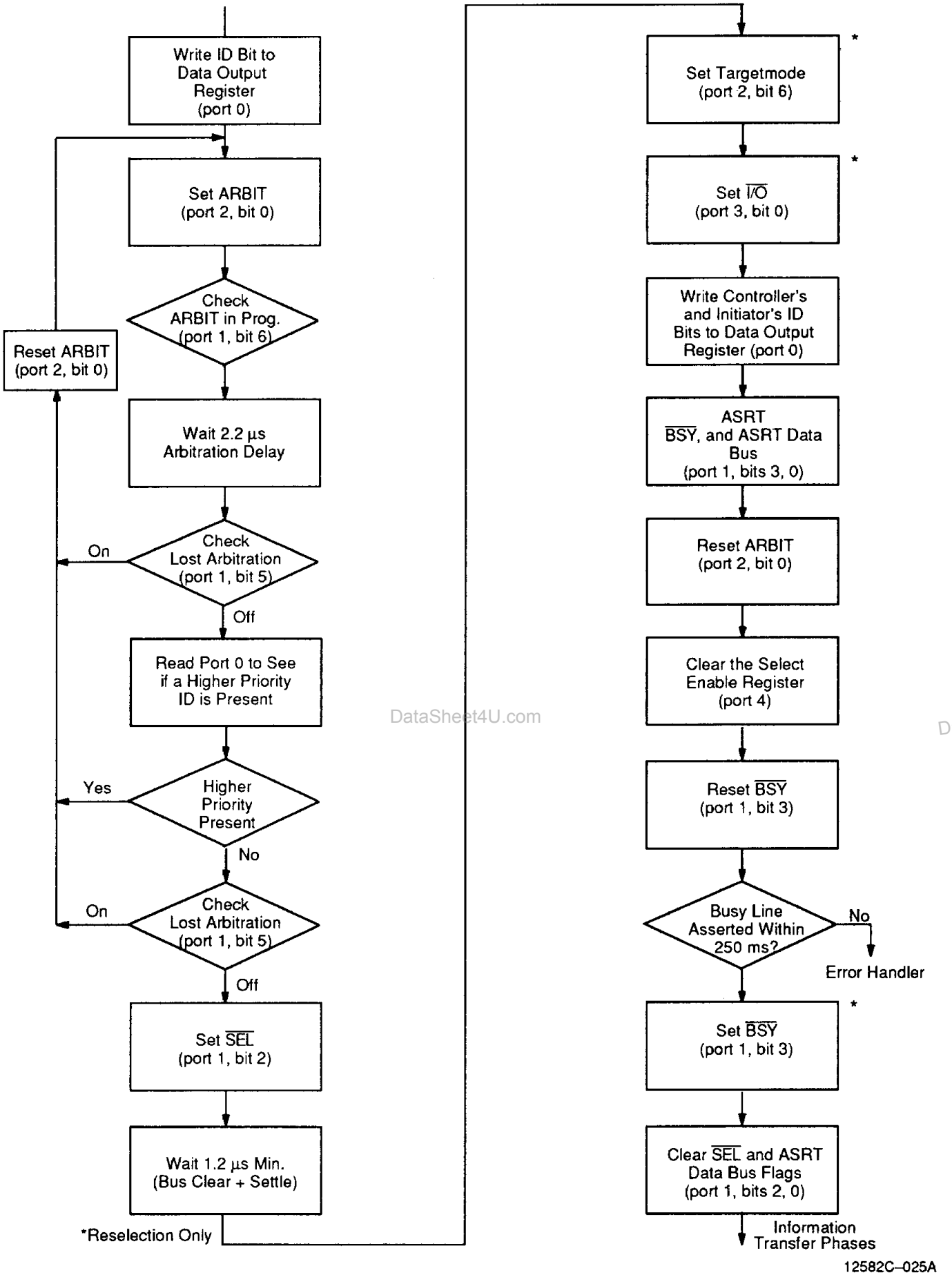
Resetting the DMA Mode Bit

A DMA operation may be halted at any time simply by resetting the DMA Mode bit. It is recommended that the DMA Mode bit be reset after receiving an $\overline{EOP}$ or bus phase-mismatch interrupt. The DMA Mode bit must then be set before writing any of the start DMA registers for subsequent bus phases.

If resetting the DMA Mode bit is used instead of $\overline{EOP}$ for Target role operation, then care must be taken to reset this bit at the proper time. If receiving data as a Target device, the DMA Mode bit must be reset once the last DRQ is received and before $\overline{DACK}$ is asserted to prevent an additional $\overline{REQ}$ from occurring. Resetting this bit causes DRQ to go inactive. However, the last byte received remains in the Input Data Register and may be obtained either by performing a normal CPU read or by cycling $\overline{DACK}$ and $\overline{RD}$. In most cases $\overline{EOP}$ is easier to use when operating as a Target device.

Flowcharts

Flowcharts are provided (see Figures 22 through 25) as a guideline to facilitate your firmware development. Firmware will vary depending on the application and the level of the SCSI protocol being supported.



12582C–025A

Figure 22. Arbitration and (Re) Selection

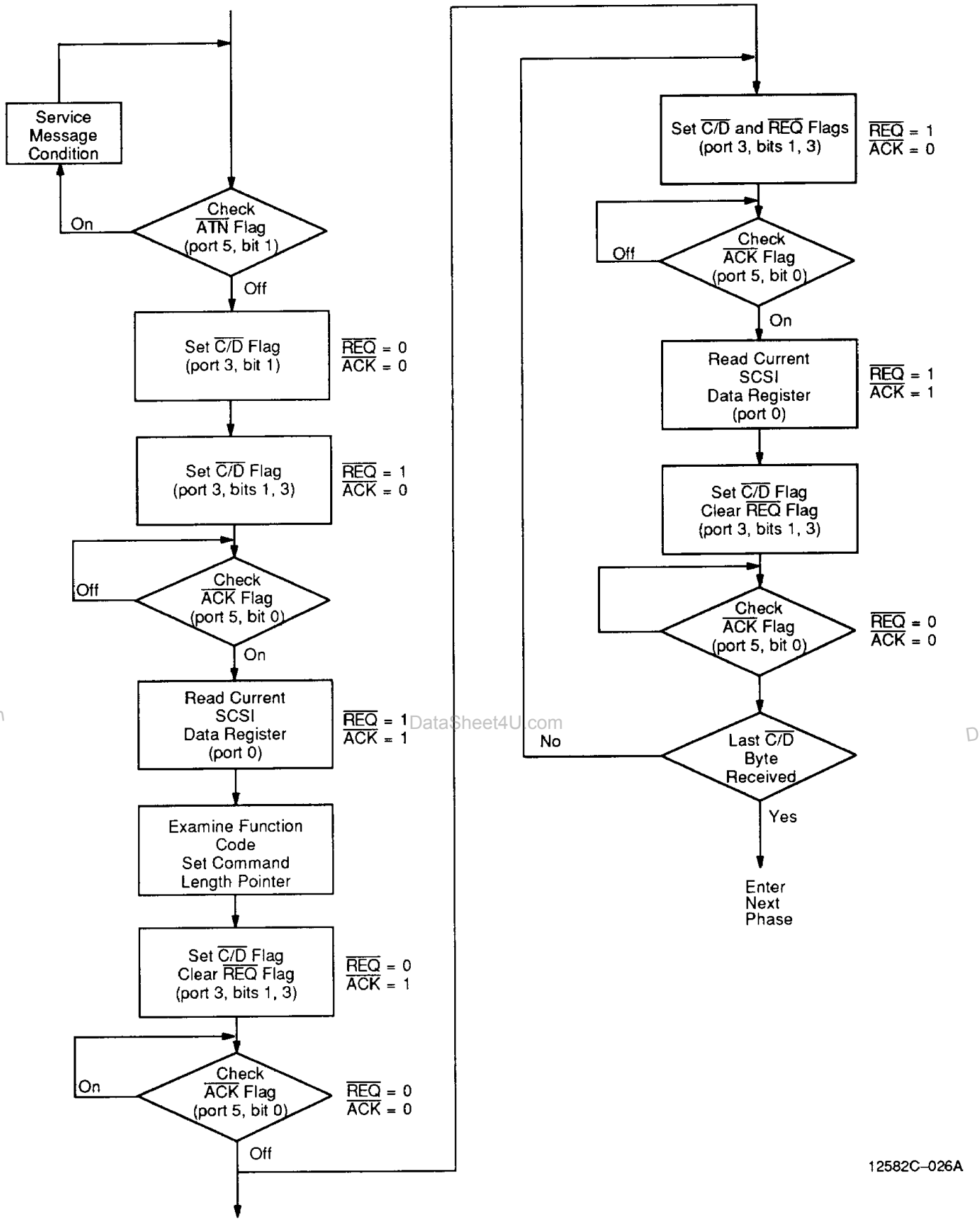
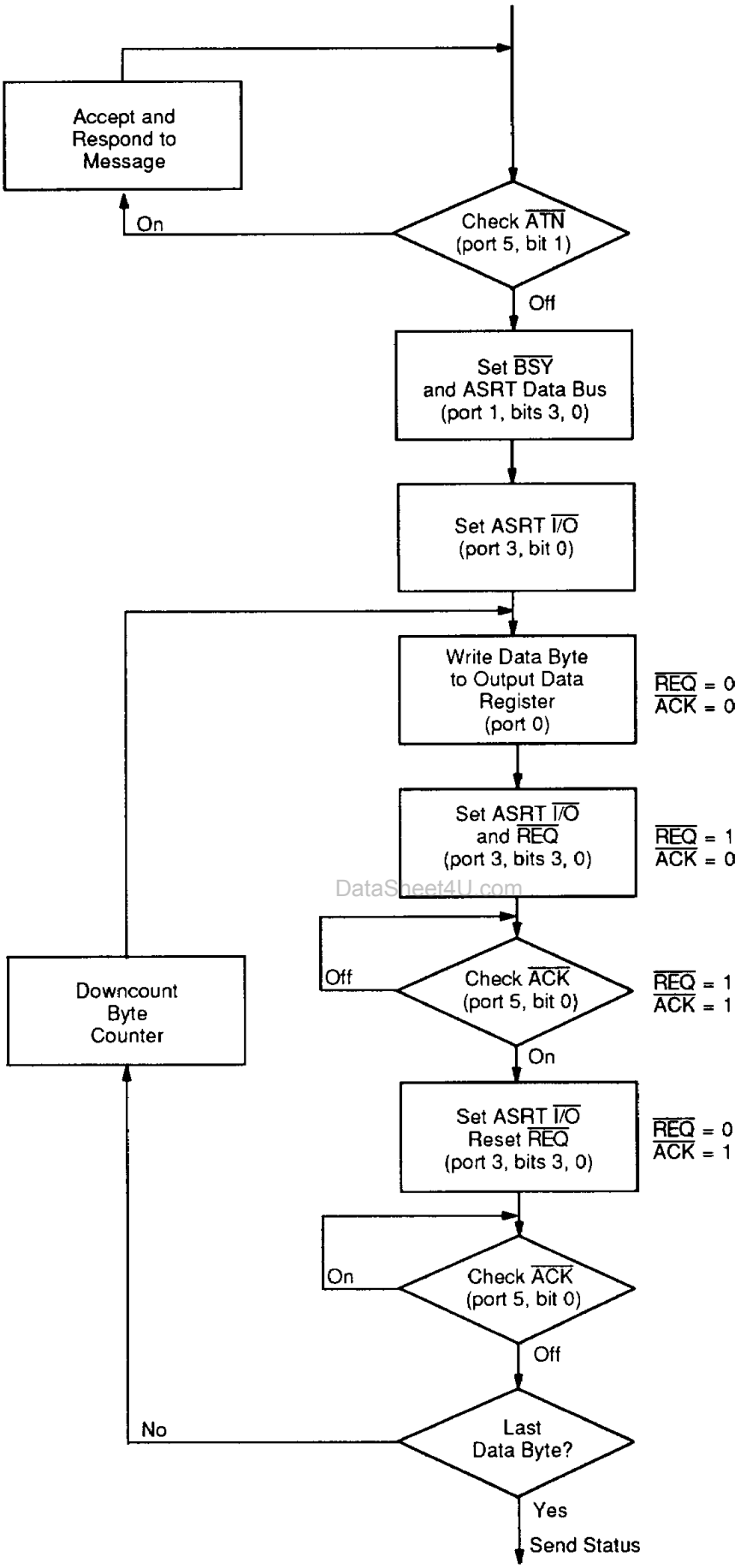
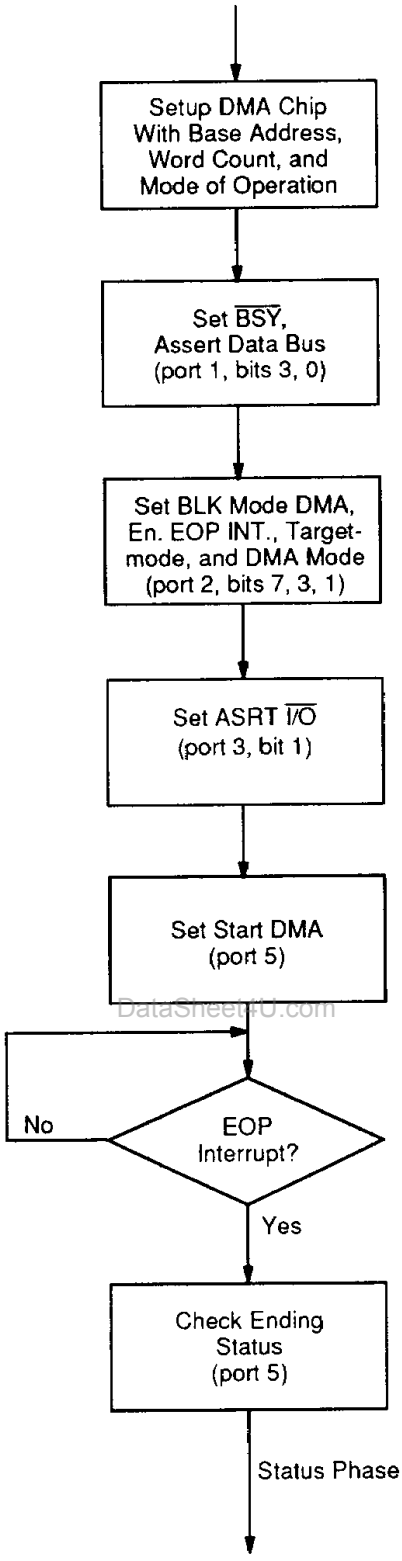


Figure 23. Command Transfer Phase (Target)



12582C–027A

Figure 24. Data Transfer to Host via Programmed I/O

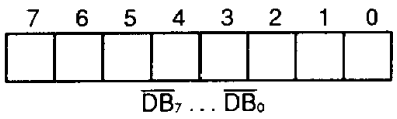


12582C–028A

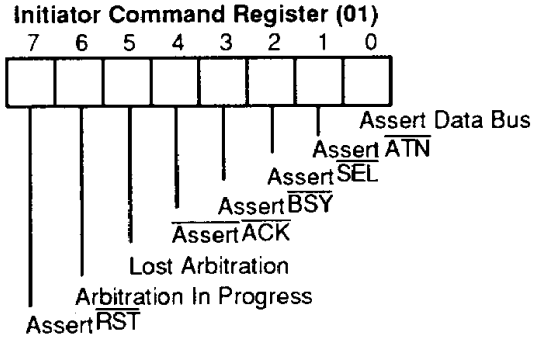
Figure 25. Data Transfer via DMA

Read

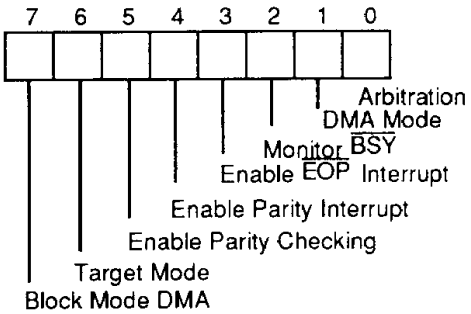
Current SCSI Data (00)



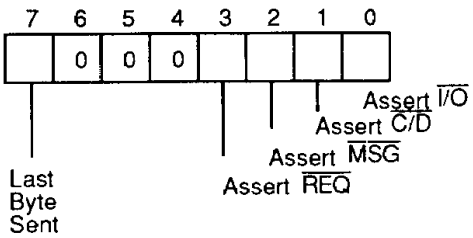
Initiator Command Register (01)



Mode Register (02)

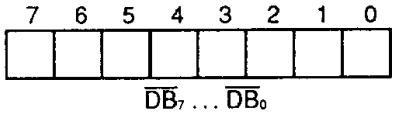


Target Command Register (03)

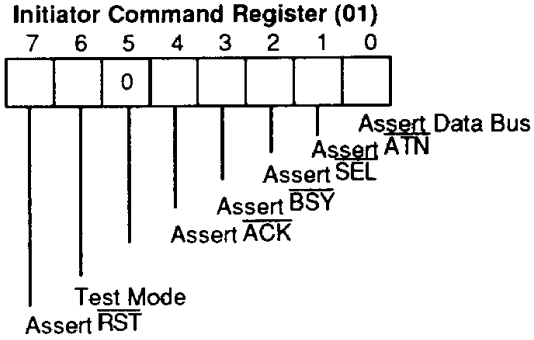


Write

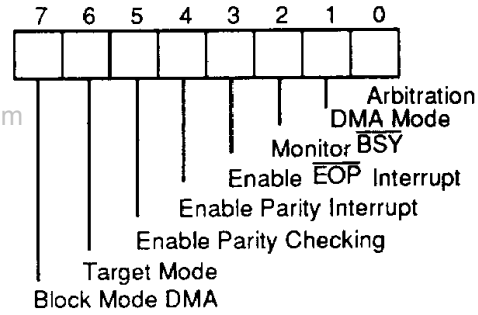
Output Data Register (00)



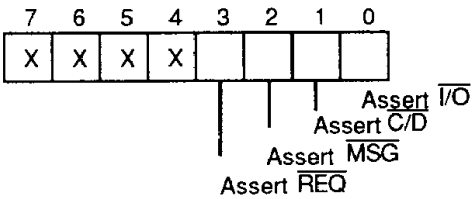
Initiator Command Register (01)



Mode Register (02)



Target Command Register (03)

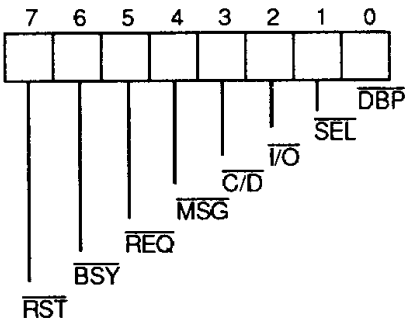


12582C–029A

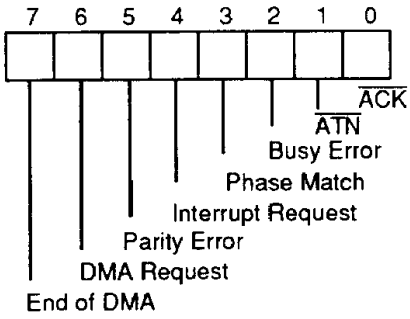
Figure 26. Register Reference Chart

Read

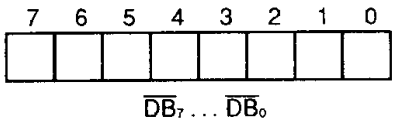
Current SCSI Bus Status (04)



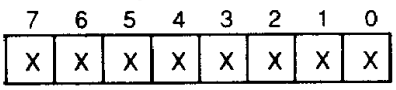
Bus and Status Register (05)



Input Data Register (06)

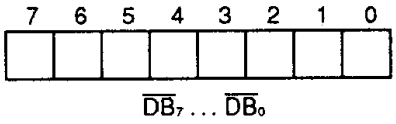


Reset Parity/Interrupt (07)

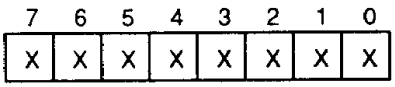


Write

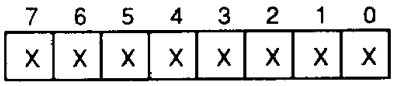
Select Enable Register (04)



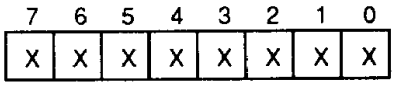
Start DMA Send (05)



Start DMA Target Receive (06)



Start DMA Initiator Receive (07)



DF006090

Note: X = Don't Care

12582C–030A

Figure 26. Register Reference Chart (continued)

ESCC ARCHITECTURE

The ESCC internal structure includes two full-duplex channels, two 10 × 19 bit SDLC/HDLC frame status FIFOs, two baud rate generators, internal control and interrupt logic, and a bus interface to a non-multiplexed bus. Associated with each channel are a number of read and write registers for mode control and status information, as well as logic necessary to interface with modems or other external devices (see Logic Symbol).

The logic for both channels provides formats, synchronization, and validation for data transferred to and from the channel interface. The modem control inputs are monitored by the control logic under program control. All of the modem control signals are general-purpose in nature and can optionally be used for functions other than modem control.

The register set for each channel includes ten control (write) registers, two SYNC character (write) registers, and four status (read) registers. In addition, each baud rate generator has two (read/write) registers for holding the time constant that determines the baud rate. Finally, associated with the interrupt logic is a write register for the interrupt vector accessible through either channel, a write-only Master Interrupt Control register and three read registers: one containing the vector with status information (Channel B only), one containing the vector without status (A only), and one containing the interrupt Pending bits (A only).

The registers for each channel are designated as follows:

WR0–WR15—Write Registers 0 through 15. An additional write register, WR7 Prime (WR7'), is available for enabling or disabling additional SDLC/HDLC enhancements if bit D0 of WR15 is set.

RR0–RR3, RR10, RR12, RR13, RR15—Read Registers 0 through 3, 10, 12, 13, 15.

If bit D2 of WR15 is set, then two additional Read Registers, RR6 and RR7, are available. These registers are used with the 10 × 19 bit Frame Status FIFO.

Table 3 lists the functions assigned to each read and write register. The ESCC contains only one WR2 and WR9, but they can be accessed by either channel. All other registers are paired (one for each channel).

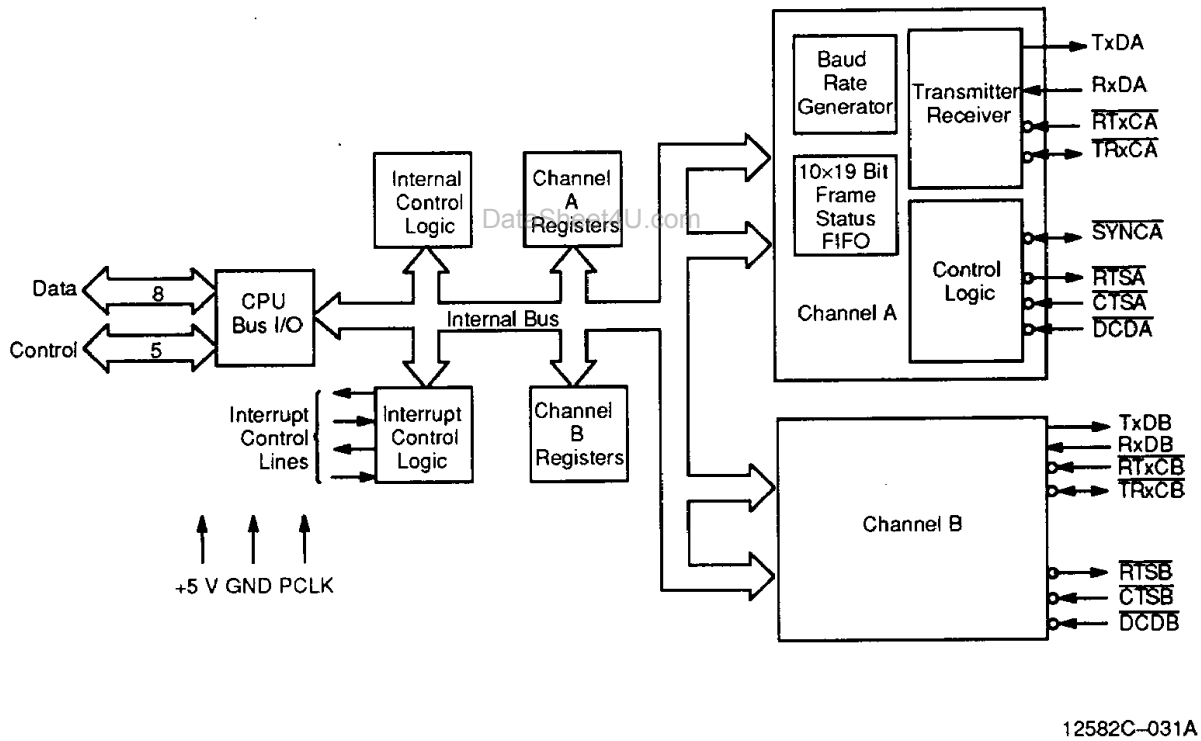


Figure 27. Block Diagram of ESCC Architecture

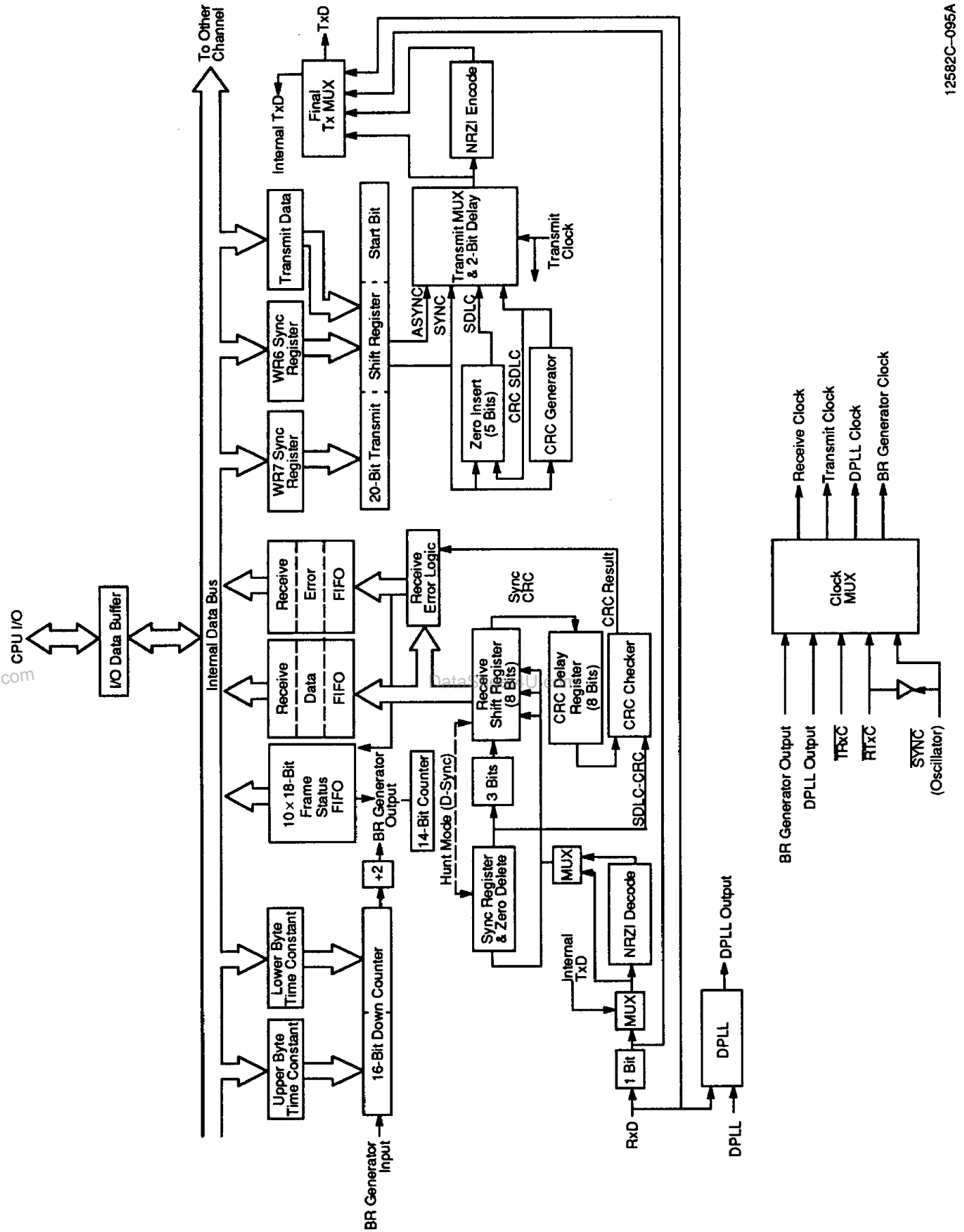
Data Path

The transmit and receive data path illustrated in Figure 28 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement, in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high-speed data. Incoming data are routed through one of several paths (data or CRC) depending on the selected mode (the character length in asynchronous modes also determines the data path).

The transmitter has an 8-bit transmit data buffer register loaded from the internal data bus and a 20-bit transmit shift register that can be loaded either from the sync-character registers or from the transmit data register. Depending on the operational mode, outgoing data are routed through one of four main paths before they are transmitted from the Transmit Data output (TxD).

Table 3. Read and Write Register Functions

Read Register Functions		Write Register Functions	
RR0	Transmit/Receive buffer status and External status	**WR0**	Command Register, Register Pointers CRC initialize, initialization commands for the various modes, shift right/shift left command
RR1	Special Receive Condition status (also 10 × 19 bit FIFO Frame Reception Status if WR15 bit D2 is set)	**WR1**	Interrupt conditions and data transfer mode definition
RR2	Modified interrupt vector (Channel B only) Unmodified interrupt vector (Channel A only)	**WR2**	Interrupt vector (accessed through either channel)
RR3	Interrupt Pending bits (Channel A only)	**WR3**	Receive parameters and control
RR6	LSB Byte Count (14-bit counter) (if WR15 bit D2 set)	**WR4**	Transmit/Receive miscellaneous parameters and modes
RR7	MSB Byte Count (14-bit counter) and 10 × 19 bit FIFO Status (if WR15 bit D2 is set)	**WR5**	Transmit parameters and controls
RR8	Receive buffer	**WR6**	Sync character or SDLC address field
RR10	Miscellaneous XMTR, RCVR status	**WR7**	Sync character or SDLC flag
RR12	Lower byte of baud rate generator time constant	**WR7′**	SDLC/HDLC enhancements (if bit D0 of WR15 set)
RR13	Upper byte of baud rate generator time constant	**WR8**	Transmit buffer
RR15	External/Status interrupt information	**WR9**	Master interrupt control and reset (accessed through either channel)
		WR10	Miscellaneous transmitter/receiver control bits, data encoding
		WR11	Clock mode control, Rx and Tx clock source
		WR12	Lower byte of baud rate generator time constant
		WR13	Upper byte of baud rate generator time constant
		WR14	Miscellaneous control bits, DPLL control
		WR15	External/Status interrupt control



12582C–095A

Figure 28. Data Path

DETAILED DESCRIPTION

The functional capabilities of the ESCC can be described from two different points of view; as a data communications device, it transmits and receives data in a wide variety of data communications protocols; as a microprocessor peripheral, it interacts with the CPU and provides vectored interrupts and handshaking signals.

Data Communications Capabilities

The ESCC provides two independent full-duplex channels programmable for use in any common asynchronous or SYNC data-communication protocol. Figure 29 and the following description briefly detail these protocols.

Asynchronous Modes

Transmission and reception can be accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input. If the Low does not persist (as in the case of a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur. Vectored interrupts allow fast servicing of error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of framing error as a new start bit; a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

The ESCC does not require symmetric transmit and receive clock signals—a feature allowing use of a wide variety of clock sources. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32, or 1/64 of the clock rate supplied to the receive and transmit clock inputs. In asynchronous modes, the $\overline{SYNC}$ pin may be programmed as an input used for functions, such as monitoring a ring indicator.

Synchronous Modes

The ESCC supports both byte-oriented and bit-oriented synchronous communication. SYNC byte-oriented protocols can be handled in several modes, allowing character synchronization with a 6-bit or 8-bit SYNC character (Monosync), any 12-bit or 16-bit SYNC pattern (Bisync), or with an external SYNC signal. Leading SYNC characters can be removed without interrupting the CPU.

Five- or 7-bit SYNC characters are detected with 8- or 16-bit patterns in the ESCC by overlapping the larger pattern across multiple incoming SYNC characters as shown in Figure 30.

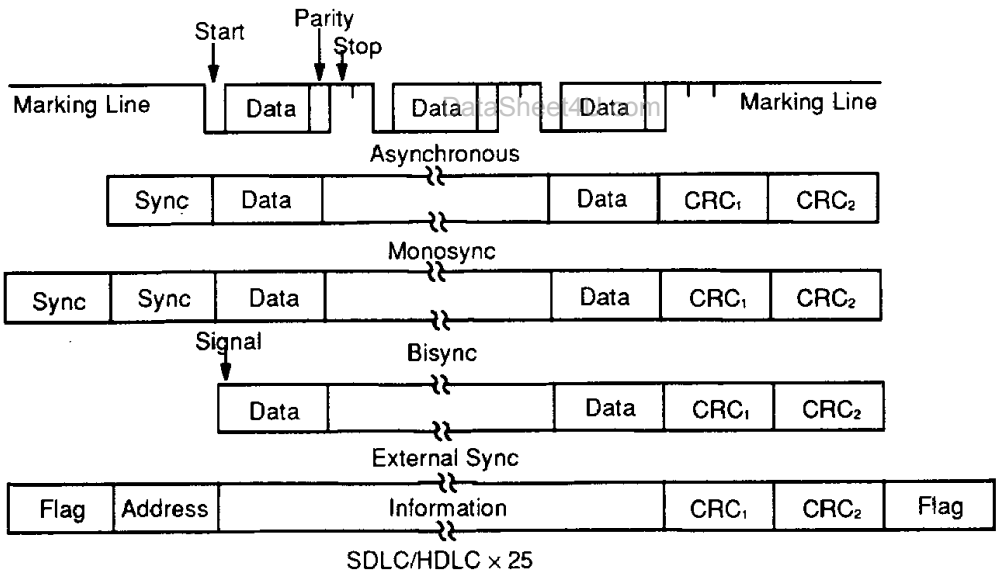


Figure 29. ESCC Protocols

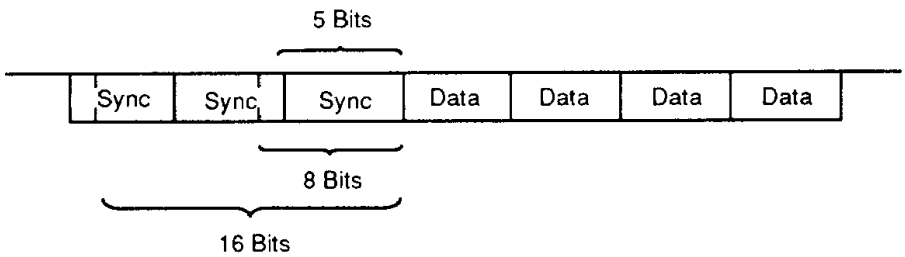


Figure 30. Detecting 5- or 7-Bit Synchronous Characters

CRC checking for Synchronous byte-oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols, such as IBM BISYNC.

Both CRC-16 ($X^{16} + X^{15} + X^2 + 1$) and CCITT ($X^{16} + X^{12} + X^5 + 1$) error checking polynomials are supported. Either polynomial may be selected in BISYNC and MONOSYNC modes. Users may preset the CRC generator and checker to all "1"s or all "0"s. The ESCC also provides a feature that automatically transmits CRC data when no other data are available for transmission. This allows for high-speed transmissions under DMA control with no need for CPU intervention at the end of a message. When there are no data or CRC to send in SYNC modes, the transmitter inserts 6-, 8-, or 16-bit SYNC characters, regardless of the programmed character length.

The ESCC supports SYNC bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero bit insertion, and CRC generation. A special command can be used to abort a frame in transmission. At the end of a message, the ESCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line consisting of continuous flag characters or a steady marking condition.

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. The ESCC may also be programmed to send an abort itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent allowing reception of a message with no prior information about the character structure in the information field of a frame.

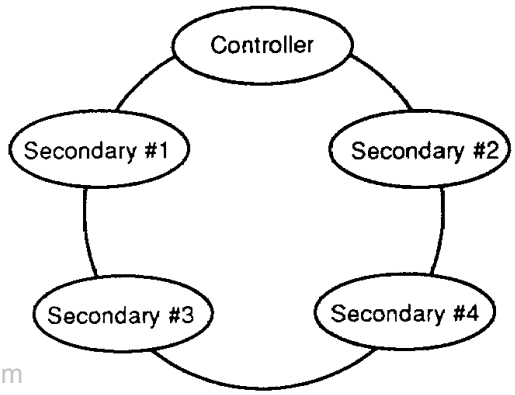
The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the $\overline{SYNC}$ pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all "0"s inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. In SDLC mode, the ESCC must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all "1"s or all "0"s. The CRC is inverted before transmission and the receiver checks against the bit pattern "0001110100001111."

NRZ, NRZI or FM coding may be used in any 1X mode. The parity options available in asynchronous modes are available in synchronous modes.

The ESCC can be conveniently used under DMA control to provide high-speed reception or transmission. In reception, for example, the ESCC can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The ESCC then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received. The CPU may also enable the DMA first and have the ESCC interrupt only on end-of-frame. This procedure allows all data to be transferred via the DMA.

SDLC Loop Mode

The ESCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow and any number of secondary stations. In SDLC Loop mode, the ESCC performs the functions of a secondary station while an ESCC operating in regular SDLC mode can act as a controller (Figure 31).



12582C–035A

Figure 31. An SDLC Loop

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop and, in fact, must pass these messages to the rest of the loop by retransmitting them with a 1-bit time delay. The secondary station can place its own message on the loop only at specific times. The controller signals that secondary stations may transmit messages by sending a special character, called an EOP (End of Poll), around the loop. The EOP character is the bit pattern "11111110." Because of zero insertion during messages, this bit pattern is unique and easily recognized.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary one of the EOP to a zero before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit can then append their messages to the message of the first secondary station by the same process. Any secondary stations without messages to send

merely echo the incoming messages and are prohibited from placing messages on the loop (except upon recognizing an EOP).

SDLC Loop mode is a programmable option in the ESCC. NRZ, NRZI, and FM coding may all be used in SDLC Loop mode.

Baud Rate Generator

Each channel in the ESCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On start-up, the flip-flop on the output is set in a High state; the value in the time constant register is loaded into the counter; and the counter starts counting down. The output of the baud rate generator toggles upon reaching zero; the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the digital phase-locked loop (see next section).

If the receive clock or transmit clock is not programmed to come from the $\overline{TRxC}$ pin, the output of the baud rate generator may be echoed out via the $\overline{TRxC}$ pin.

The following formula relates the time constant to the baud rate where PCLK or $\overline{RTxC}$ is the baud rate generator input frequency in Hz. The clock mode is X1, X16, X32, or X64 as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select X1 and Asynchronous should select X16, X32, or X64.

$$\text{Time Constant} = \left[\frac{\text{PCLK or RTxC Frequency}}{2\,(\text{Baud Rate})(\text{Clock Mode})}\right] - 2$$

The following formula relates the time constant to the baud rate. (The baud rate is in bits/second and the BR clock period is in seconds given by Clock Mode/Clock Frequency.)

$$\text{baud rate} = \frac{1}{2\,(\text{Time Constant} + 2) \times (\text{BR Clock Period})}$$

Time Constant Values for Standard Baud Rates at BR Clock = 3.9936 MHz			
Rate (Baud)	Time Constant (decimal/Hex notation)		Error
19200	102	(0066)	0
9600	206	(00CE)	0
7200	275	(0113)	0.12%
4800	414	(019E)	0
3600	553	(0229)	0.06%
2400	830	(033E)	0
2000	996	(03E4)	0.04%
1800	1107	(0453)	0.03%
1200	1662	(067E)	0
600	3326	(0CFE)	0
300	6654	(19FE)	0
150	13310	(33FE)	0
134.5	14844	(39FC)	0.0007%
110	18151	(46E7)	0.0015%
75	26622	(67FE)	0
50	39934	(98FE)	0

Digital Phase-Locked Loop

The ESCC contains a digital phase-locked loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock may then be used as the ESCC receive clock, the transmit clock, or both.

For NRZI encoding, the DPLL counts the 32X clock to create nominal bit times. As the 32X clock is counted, the DPLL is searching the incoming data stream for edges (either 1/0 or 0/1). As long as no transitions are detected, the DPLL output will be free running and its input clock source will be divided by 32, producing an output clock without any phase jitter. Upon detecting a transition the DPLL will adjust its clock output (during the next counting cycle) by adding or subtracting a count of 1, thus producing a terminal count closer to the center of the bit cell. The adding or subtracting of a count of 1 will produce a phase jitter of ±5.63° on the output of the DPLL. Because the ESCC's DPLL uses both edges of the incoming signal to compare with its clock source, the mark-space ratio (50%) of the incoming signal should not deviate by more than ±1.5% if proper locking is to occur.

For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16 and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15/16 counting transition.

The 32X clock for the DPLL can be programmed to come from either the $\overline{RTxC}$ input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the ESCC via the $\overline{TRxC}$ pin (if this pin is not being used as an input).

Crystal Oscillator

When using a crystal oscillator to supply the receive or transmit clocks to a channel of the ESCC, the user should :

1. Select a crystal oscillator which satisfies the following specifications:
   - 30 ppm @ 25°C
   - 50 ppm over temperature of –20° to 70°C
   - 5 ppm/yr aging
   - 5 mW drive level
2. Place crystal across $\overline{RTxC}$ and $\overline{SYNC}$ pins
3. Place 30 pF capacitors to ground from both $\overline{RTxC}$ and $\overline{SYNC}$ pins
4. Set bit D_7 of WR11 to "1."

Data Encoding

The ESCC may be programmed to encode and decode the serial data in four different ways (Figure 6). In NRZ encoding, a "1" is represented by a High level, and a "0" is represented by a Low level. In NRZI encoding, a "1" is represented by no change in level, and a "0" is represented by a change in level. In FM_1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A "1" is represented by an additional transition at the center of the bit cell, and a "0" is represented by no additional transition at the center of the bit cell. In FM_0 (bi-phase space), a transition occurs at the beginning of every bit cell. A "0" is represented by an additional transition at the center of the bit cell, and a "1" is represented by no additional transition at the center of the bit cell. In addition to these four methods, the ESCC can be used to decode Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0/1, the bit is a "0." If the transition is 1/0, the bit is a "1."

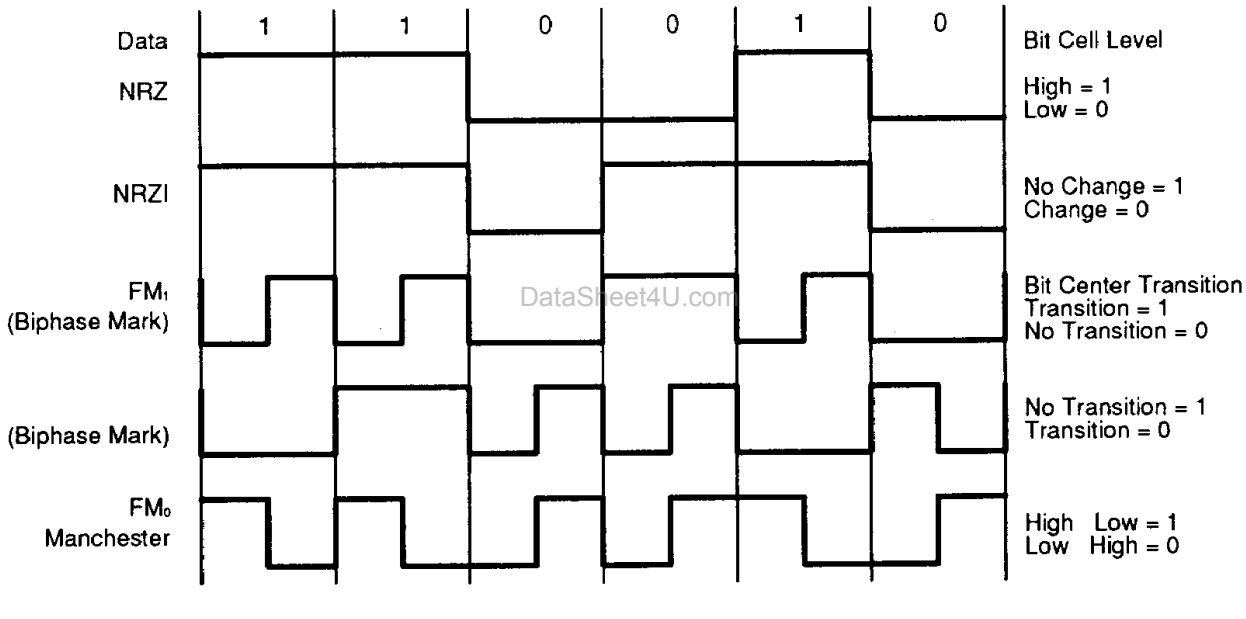


Figure 32. Data Encoding Methods

Auto Echo and Local Loopback

The ESCC is capable of automatically echoing everything it receives. This feature is useful mainly in asynchronous modes but works in SYNC and SDLC modes as well. In Auto Echo mode, TxD is RxD. Auto Echo mode can be used with NRZI or FM encoding with no additional delay, because the data stream is not decoded before retransmission. In this mode, the transmitter is actually bypassed, and the programmer is responsible for disabling transmitter interrupts and $\overline{WAIT/REQUEST}$ on transmit.

The ESCC is also capable of Local Loopback. In this mode, TxD is RxD just as in Auto Echo mode. However, in Local Loopback mode, the internal transmit data is tied to the internal receive data, and RxD is ignored (except to be echoed out via TxD). The $\overline{DCD}$ input is also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in asynchronous, SYNC and SDLC modes with NRZ, NRZI or FM coding of the data stream.

I/O Interface Capabilities

The ESCC offers the choice of Polling, Interrupt (vectored or nonvectored), and Block Transfer modes to transfer data, status, and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

Polling

All interrupts are disabled. Three status registers in the ESCC are automatically updated whenever any function is performed. For example, end-of-frame in SDLC mode sets a bit in one of these status registers. The idea behind polling is for the CPU to periodically read a status register until the register contents indicate the need for data to be transferred. Only one register needs to be read; depending on its contents, the CPU either writes data, reads data, or continues. Two bits in the register indicate the need for data transfer. An alternative is a poll of the Interrupt Pending register to determine the source of an interrupt. The status for both channels resides in one register.

Interrupts

When an ESCC responds to an Interrupt Acknowledge signal ($\overline{INTACK}$) from the CPU, an interrupt vector may be placed on the data bus. This vector is written in WR2 and may be read in RR2A or RR2B (Figures 34 and 35).

To speed interrupt response time, the ESCC can modify three bits in this vector to indicate status. If the vector is read in Channel A, status is never included; if it is read in Channel B, status is always included.

Each of the six sources of interrupts in the ESCC (Transmit, Receive and External/Status interrupts in both channels) has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt Under Service (IUS), and Interrupt Enable (IE). Operation of the IE bit is straightforward. If the IE bit is set for a given interrupt source, then that source can request interrupts. The exception is when the MIE (Master Interrupt Enable) bit in WR9 is reset and no interrupts may be requested. The IE bits are write-only.

In the ESCC, the IP bit signals a need for interrupt servicing. When an IP bit is set to "1" and the IEI input is High, the $\overline{INT}$ output is pulled Low, requesting an interrupt. In the ESCC, if the IE bit is set for an interrupt, then the IP for that source can never be set. The IP bits are readable in RR3A.

There are three types of interrupts: Transmit, Receive and External/Status. Each interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receive, Transmit and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted when the transmit buffer becomes empty. (This implies that the transmitter must have had a data character written into it so that it can become empty.) When enabled, the Receive can interrupt the CPU in one of three ways:

- Interrupt on First Receive Character or Special Receive condition
- Interrupt on all Receive Characters or Special Receive condition
- Interrupt on Special Receive condition only

Interrupt on First Character or Special Condition and Interrupt on Special Condition Only are typically used with the Block Transfer mode. A Special Receive Condition is one of the following: receiver overrun, framing error in asynchronous mode, end-of-frame in SDLC mode, and optionally, a parity error. The Special Receive Condition interrupt is different from an ordinary Receive Character Available interrupt only in the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt can occur from Special Receive Conditions any time after the first Receive Character Interrupt.

The main function of the External/Status interrupt is to monitor the signal transitions of the $\overline{DCD}$, and $\overline{SYNCA}$ pins; however, an External/Status interrupt is also caused by a Transmit Underrun condition, a zero count in the baud rate generator, the detection of a Break (asynchronous mode), Abort (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the Abort or EOP has a special feature allowing the ESCC to interrupt when the Abort or EOP sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Abort condition in external logic in SDLC mode. In SDLC Loop mode, this feature allows secondary stations to recognize the wishes of the primary station to regain control of the loop during a poll sequence.

CPU/DMA Block Transfer

The ESCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the $\overline{WAIT/REQUEST}$ output in conjunction with the Wait/Request bits in WR1. The $\overline{WAIT/REQUEST}$ output can be defined under software control as a $\overline{WAIT}$ line in the CPU Block Transfer mode or as a $\overline{REQUEST}$ line in the DMA Block Transfer mode.

To a DMA controller, the ESCC $\overline{REQUEST}$ output indicates that the ESCC is ready to transfer data to or from memory. To the CPU, the $\overline{WAIT}$ line indicates that the ESCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The $\overline{DTR/REQUEST}$ can be used as the transmit request line, thus allowing full-duplex operation under DMA control.

PROGRAMMING INFORMATION

Each channel has fifteen Write registers that are individually programmed from the system bus to configure the functional personality of each channel. Each channel also has eight Read registers from which the system can read Status, Baud rate, or Interrupt information.

On the ESCC, only four data registers (Read and Write for Channels A and B) are directly selected by a High on the D/$\overline{C}$ input and the appropriate levels on the $\overline{RD}$, $\overline{WR}$ and A/$\overline{B}$ pins. All other registers are addressed indirectly by the content of Write Register 0 in conjunction with a Low on the D/$\overline{C}$ input and the appropriate levels on the $\overline{RD}$, $\overline{WR}$ and A/$\overline{B}$ pins. If bit D3 in WR0 is 1 and bits 5 and 6 are 0, then bits 0, 1, 2 address the higher registers 8 through 15. If bits 4, 5, 6 contain a different code, bits 0, 1, 2 address the lower registers 0 through 7 as shown in Table 4.

Writing to or reading from any register except RR0, WR0 and the Data Registers thus involves two operations:

First, write the appropriate code into WR0, then follow this by a write or read operation on the register thus specified. Bits 0 through 4 in WR0 are automatically cleared after this operation, so that WR0 then points to WR0 or RR0 again.

Channel A/Channel B selection is made by the A/$\overline{B}$ input (HIGH = A, Low = B)

The system program first issues a series of commands to initialize the basic mode of operation. This is followed by other commands to qualify conditions within the selected mode. For example, the asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first. Then the interrupt mode would be set and, finally, receiver or transmitter enable.

Table 4. Register Addressing

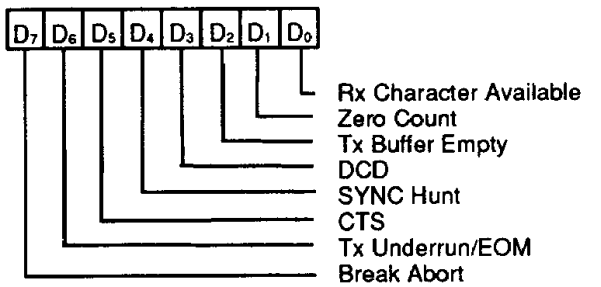
D/$\overline{C}$	"Point High" Code In WR0:	D2, D1, D0 In WR0:			Write Register	Read Register
HIGH	Either Way	x	x	x	Data	Data
LOW	Not True	0	0	0	0	0
LOW	Not True	0	0	1	1	1
LOW	Not True	0	1	0	2	2
LOW	Not True	0	1	1	3	3
LOW	Not True	1	0	0	4	(0)
LOW	Not True	1	0	1	5	(1)
LOW	Not True	1	1	0	6	(2)
LOW	Not True	1	1	1	7	(3)
LOW	True	0	0	0	Data	Data
LOW	True	0	0	1	9	–
LOW	True	0	1	0	10	10
LOW	True	0	1	1	11	(15)
LOW	True	1	0	0	12	12
LOW	True	1	0	1	13	13
LOW	True	1	1	0	14	(10)
LOW	True	1	1	1	15	15

Read Registers

The ESCC contains eight read registers [actually nine, counting the receive buffer (RR8) in each channel]. Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers (RR12 and RR13) may be read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A). In addition, if bit D2 of WR15 is set, RR6 and RR7 are available for providing frame status from the 10 × 19 bit Frame Status FIFO. Figure 33 shows the formats for each read register.

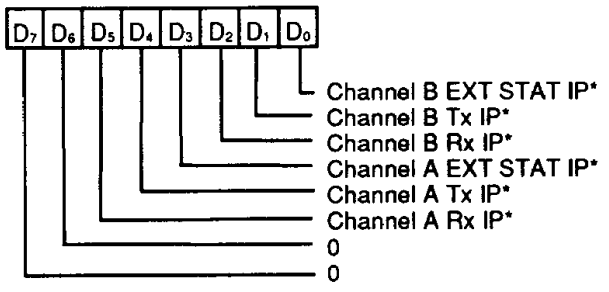
The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring, for example, when the interrupt vector indicates a Special Receive Condition interrupt, all the appropriate error bits can be read from a single register (RR1).

Read Register 0



12582C–037A

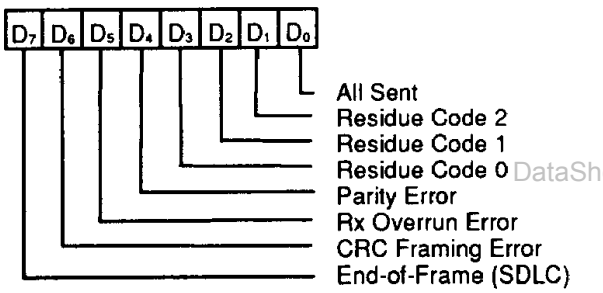
Read Register 3



*Always 0 in B Channel

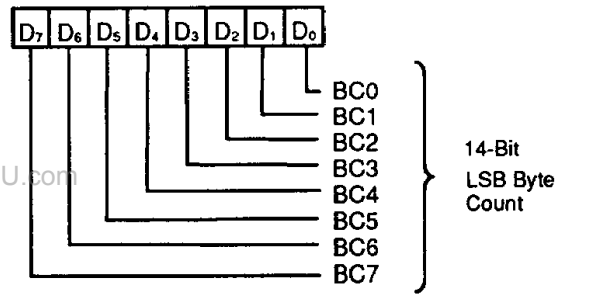
12582C–040A

Read Register 1



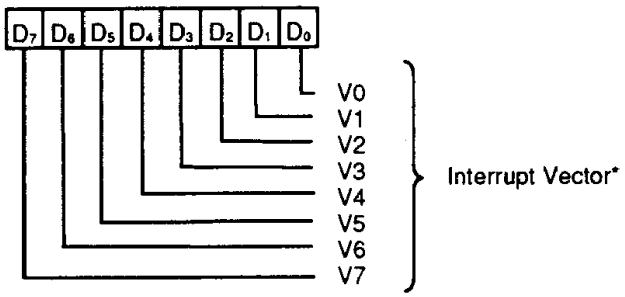
12582C–038A

Read Register 6



12582C–041A

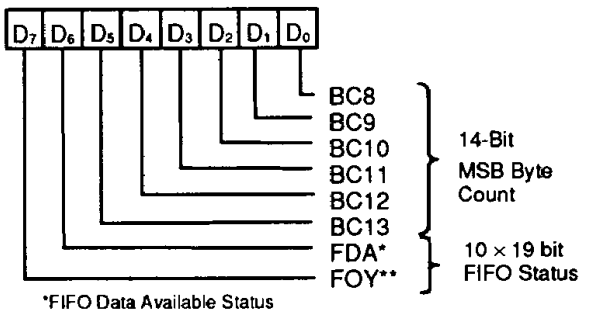
Read Register 2



*Modified in B Channel

12582C–039A

Read Register 7



*FIFO Data Available Status
**FIFO Overflow Status

12582C–042A

Figure 33. Read Register Bit Functions

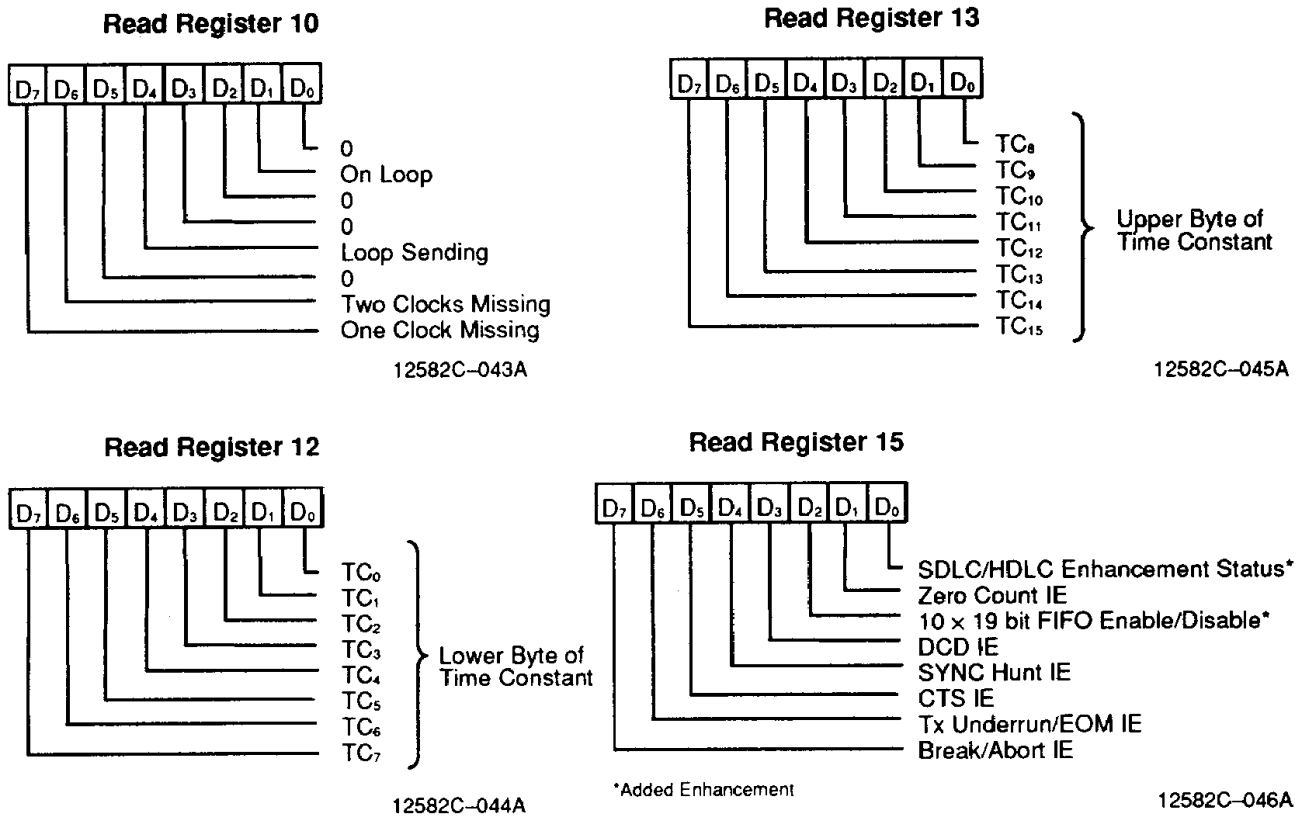


Figure 33. Read Register Bit Functions (continued)

Write Registers

The ESCC contains 15 write registers (16 counting WR8, the transmit buffer) in each channel. These write registers are programmed separately to configure the functional "personality" of the channels. Two registers (WR2 and WR9) are shared by the two channels that can be accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits. In addition, if bit D0 of WR15 is set, write register seven prime (WR7′) is available for programming additional SDLC/HDLC enhancements. When bit D0 of WR15 is set, executing a write to WR7 actually writes to WR7′ to further enhance the functional "personality" of each channel. Figure 34 shows the format of each write register.

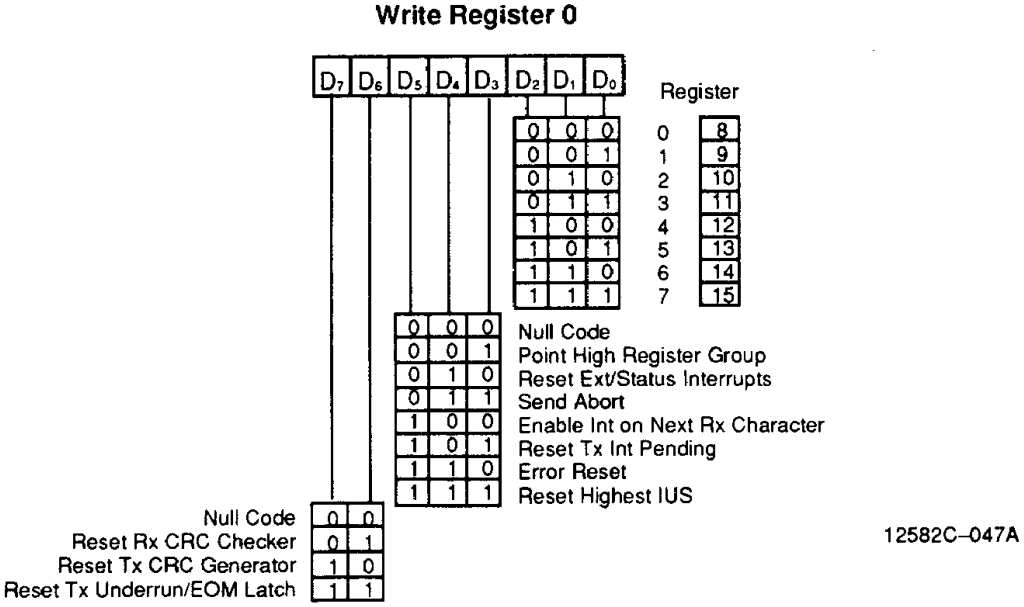
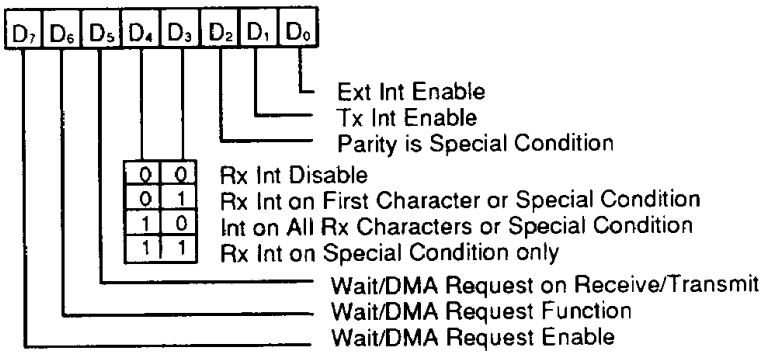


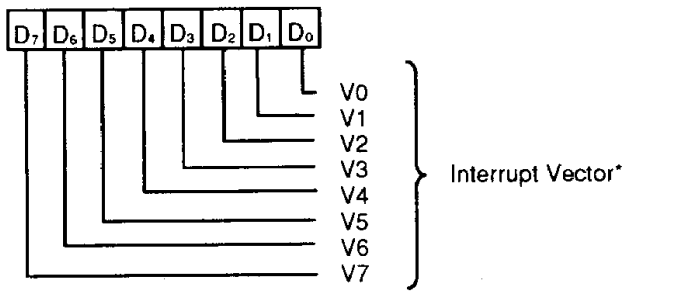
Figure 34. Write Register Bit Functions

Write Register 1



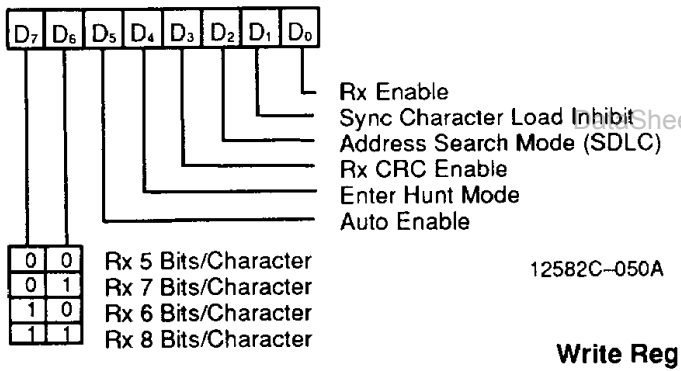
12582C–048A

Write Register 2



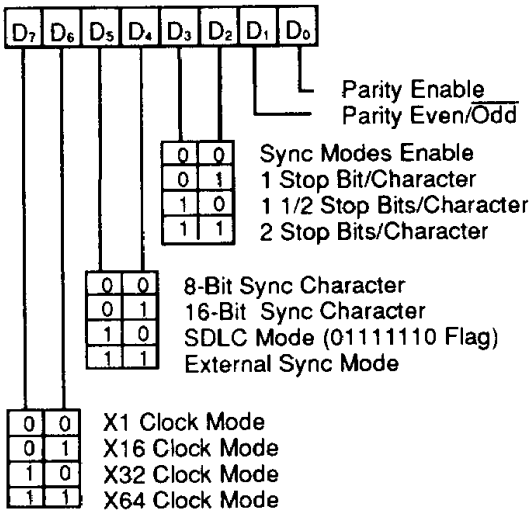
12582C–049A

Write Register 3



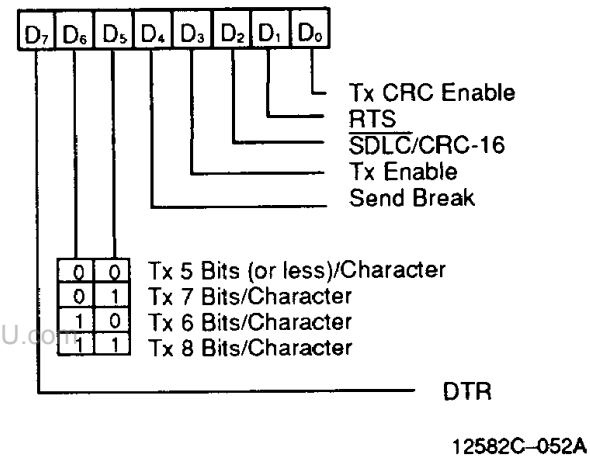
12582C–050A

Write Register 4



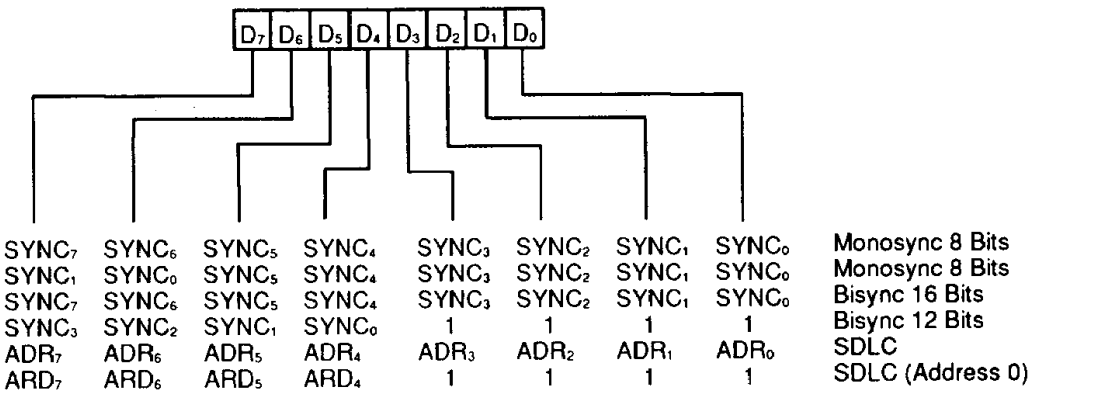
12582C–051A

Write Register 5



12582C–052A

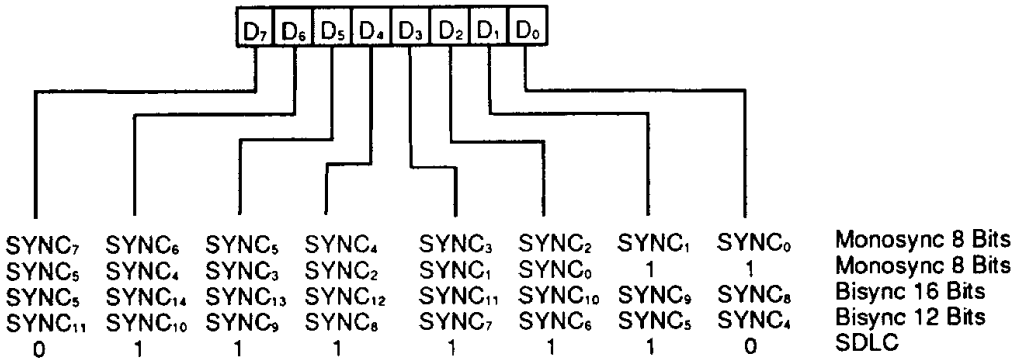
Write Register 6



12582C–053A

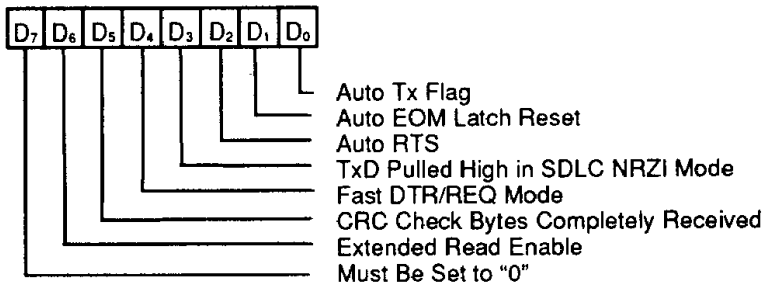
Figure 34. Write Register Bit Functions (continued)

Write Register 7



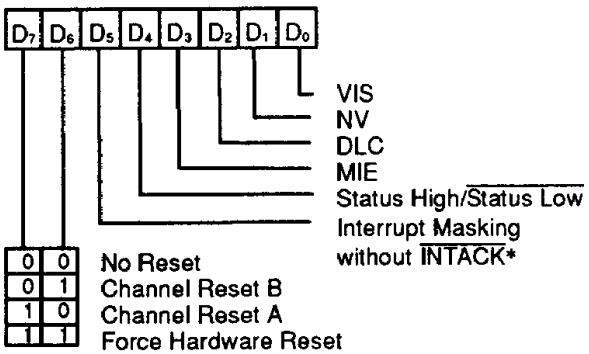
12582C–054A

Write Register 7′



12582C–055A

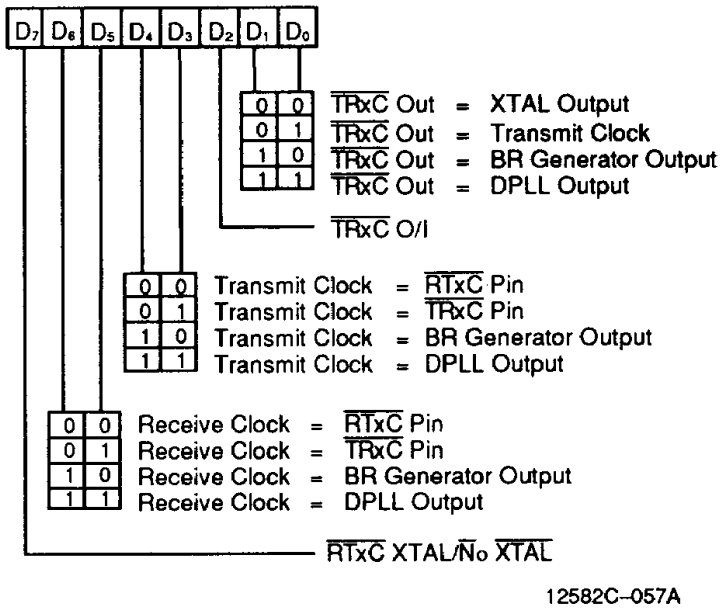
Write Register 9



*Added Enhancement

12582C–056A

Write Register 11



12582C–057A

Figure 34. Write Register Bit Functions (continued)

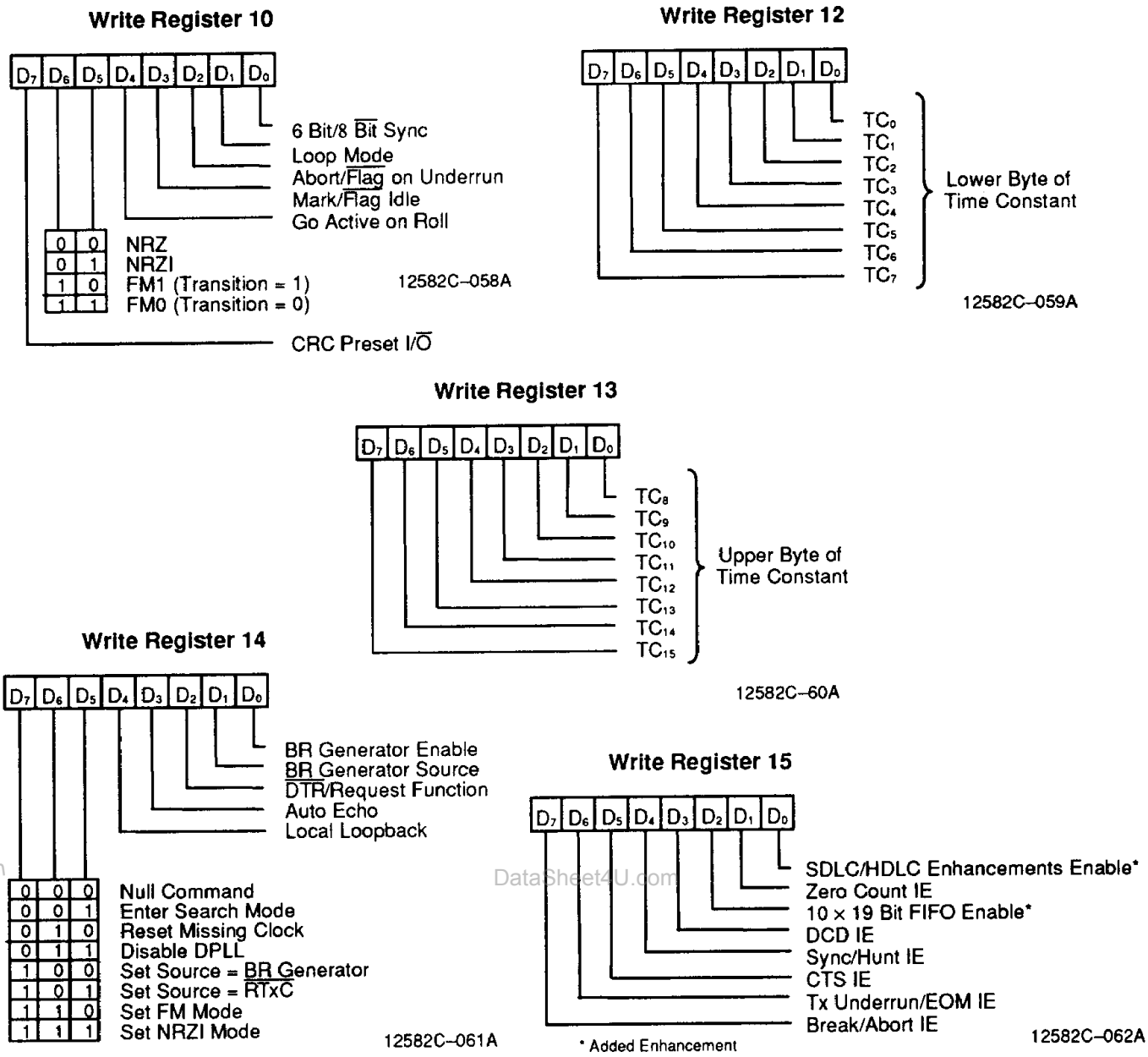


Figure 34. Write Register Bit Functions (continued)

ESCC Timing

The ESCC generates internal control signals from $\overline{WR}$ and $\overline{RD}$ that are related to PCLK. Since PCLK has no phase relationship with $\overline{WR}$ and $\overline{RD}$, the circuitry generating these internal control signals must provide time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK. The recovery time applies only between bus transactions involving the ESCC. The recovery time required for proper operation is specified from the falling edge of $\overline{WR}$ or $\overline{RD}$ in the first transaction involving the ESCC, to the falling edge of $\overline{WR}$ or $\overline{RD}$ in the second transaction involving the ESCC. This time must be at least 3 1/2 PCLK regardless of which register or channel is being accessed.

Interrupt Acknowledge Cycle Timing

Figure 37 illustrates Interrupt Acknowledge cycle timing. The ESCC may be programmed to respond to $\overline{RD}$ Low by placing its interrupt vector on D0–D7 and it then sets the appropriate Interrupt-Under-Service latch internally.

Read Cycle Timing

Figure 35 illustrates Read cycle timing. Addresses on A/$\overline{B}$ and D/$\overline{C}$ and the status on $\overline{INTACK}$ must remain stable throughout the cycle. If $\overline{CE1}$ falls after $\overline{RD}$ falls or if it rises before $\overline{RD}$ rises, the effective $\overline{RD}$ is shortened. $\overline{CE2}$ and $\overline{DACK}$ must be inactive.

Write Cycle Timing

Figure 36 illustrates Write Cycle timing. Addresses on A/$\overline{B}$ and D/$\overline{C}$ and the status on $\overline{INTACK}$ must remain stable throughout the cycle. If $\overline{CE1}$ falls after $\overline{WR}$ falls or if it rises before $\overline{WR}$ rises, the effective $\overline{WR}$ is shortened. Data must be valid before the rising edge of $\overline{WR}$. $\overline{CE2}$ and $\overline{DACK}$ must be inactive.

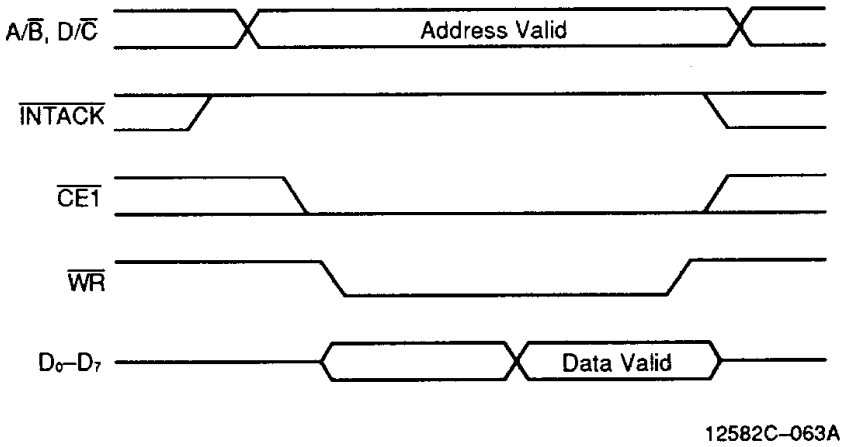


Figure 35. Read Cycle Timing

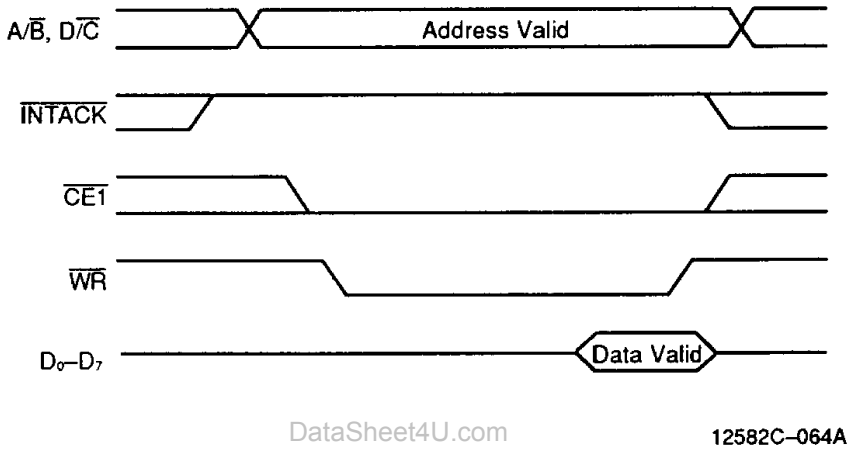


Figure 36. Write Cycle Timing

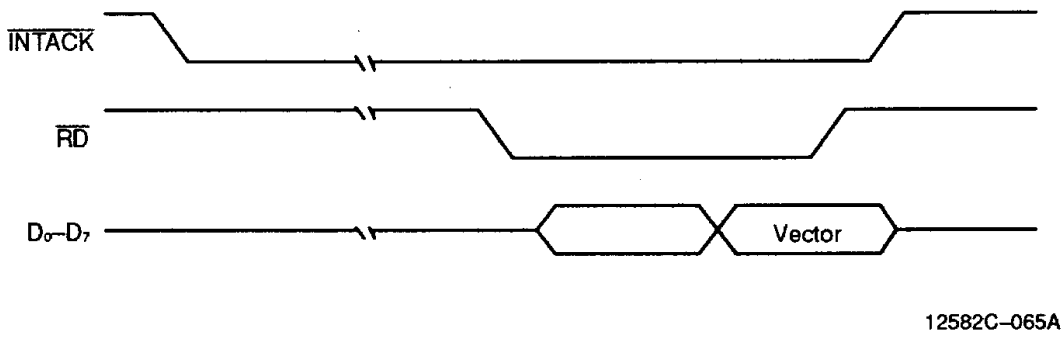


Figure 37. Interrupt Acknowledge Cycle Timing

FIFO

FIFO Enhancements

When used with a DMA controller, the ESCC Frame Status FIFO enhancement maximizes the ESCC's ability to receive high-speed back-to-back SDLC messages while minimizing frame overruns due to CPU latencies in responding to interrupts.

Additional logic was added to the industry-standard NMOS Am8530H consisting of a 10-deep by 19-bit status FIFO, a 14-bit receive byte counter, and control logic as shown in Figure 38. The 10×19 bit status FIFO is separate from the existing three-byte receive data and Error FIFOs.

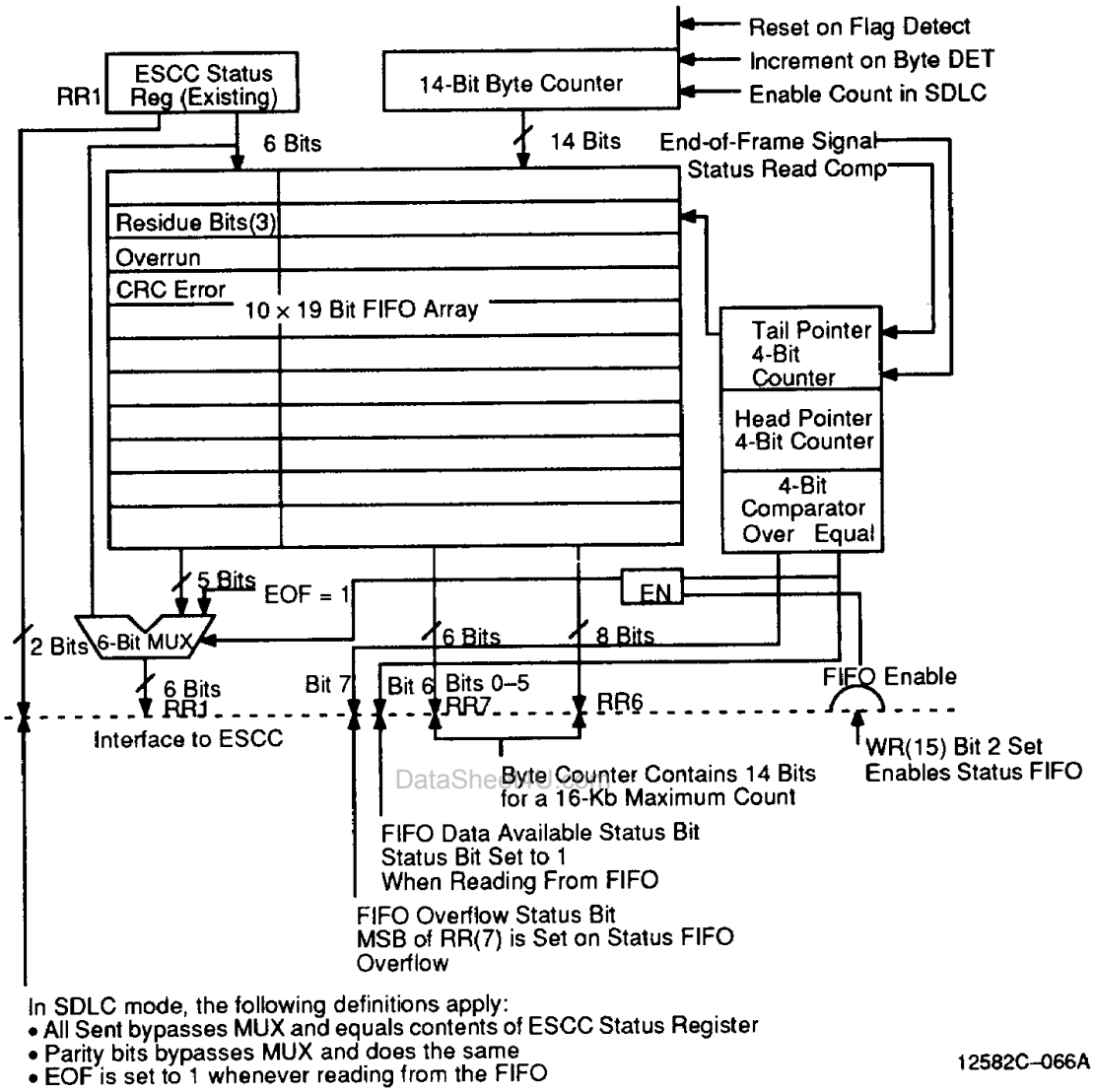


Figure 38. ESCC Status Register Modifications

When the enhancement is enabled, the status in Read Register 1 (RR1) and byte count for the SDLC frame will be stored in the 10 × 19 bit status FIFO. This allows the DMA controller to transfer the next frame into memory while the CPU verifies that the message was properly received.

Summarizing the operation, data is received, assembled, and loaded into the three-byte receive FIFO before being transferred to memory by the DMA controller. When a flag is received at the end of an SDLC frame, the frame byte count from the 14-bit counter and five status bits are loaded into the status FIFO for verification by the CPU. The CRC checker is automatically reset in preparation for the next frame which can begin immediately.

Since the byte count and status are saved for each frame, the message integrity can be verified at a later time. Status information for up to 10 frames can be stored before a status FIFO overrun could occur.

If receive interrupts are enabled while the 10×19 FIFO is enabled, an SDLC end-of-frame special condition will not lock the three-byte Receive data FIFO. An SDLC end-of-frame still locks the three-byte Receive data FIFO in "Interrupt on first Receive Character or Special Condition" and "Interrupt on Special Condition Only" modes when the 10 × 19 FIFO is disabled. This feature allows the 10×19 SDLC FIFO to accept multiple SDLC frames without CPU intervention at the end of each frame.

FIFO Detail

For a better understanding of details of the FIFO operation, refer to the block diagram contained in Figure 38.

Enable/Disable

This FIFO is implemented so that it is enabled when WR15 bit 2 is set and the ESCC is in the SDLC/HDLC mode, otherwise the status register contents bypass the FIFO and go directly to the bus interface (the FIFO pointer logic is reset either when disabled or via a channel or power-on reset). When the FIFO mode is disabled, the ESCC is completely downward-compatible with the NMOS Am8530H. The FIFO mode is disabled on power-up (WR15 bit 2 is set to "0" on reset). The effects of backward compatibility on the register set are that RR4 is an image of RR0, RR5 is an image of RR1, RR6 is an image of RR2, and RR7 is an image of RR3. For the details of the added registers, refer to Figure 15. The status of the FIFO Enable signal can be obtained by reading RR15 bit 2. If the FIFO is enabled, the bit will be set to "1"; otherwise, it will be reset.

Read Operation

When WR15 bit 2 is set and the FIFO is not empty, the next read to status register RR1 or the additional registers RR7 and RR6 will actually be from the FIFO. Reading status register RR1 causes one location of the FIFO to be emptied, so status should be read after reading the byte count, otherwise the count will be incorrect. Before the FIFO underflows, it is disabled. In this case, the multiplexer is switched to allow status to be read directly from the status register, and reads from RR7 and RR6 will contain bits that are undefined. Bit 6 of RR7 (FIFO Data Available) can be used to determine if status data is coming from the FIFO or directly from the status register, since it is set to "1" whenever the FIFO is not empty.

Because not all status bits are stored in the FIFO, the All Sent, Parity, and EOF bits will bypass the FIFO. The status bits sent through the FIFO will be Residue Bits (3), Overrun, and CRC Error.

The sequence for proper operation of the byte count and FIFO logic is to read the registers in the following order, RR7, RR6, and RR1 (reading RR6 is optional). Additional logic prevents the FIFO from being emptied by multiple reads from RR1. The read from RR7 latches the FIFO empty/full status bit (bit 6) and steers the status multiplexer to read from the ESCC megacell instead of the status FIFO (since the status FIFO is empty). The read from RR1 allows an entry to be read from the FIFO (if the FIFO was empty, logic is added to prevent a FIFO underflow condition).

Write Operation

When the end of an SDLC frame (EOF) has been received and the FIFO is enabled, the contents of the status and byte-count registers are loaded into the FIFO. The EOF signal is used to increment the FIFO. If the FIFO overflows, the MSB of RR7 (FIFO Overflow) is set to indicate the overflow. This bit and the FIFO control logic is reset by disabling and re-enabling the FIFO control bit (WR15 bit 2). For details of FIFO control timing during an SDLC frame, refer to Figure 39.

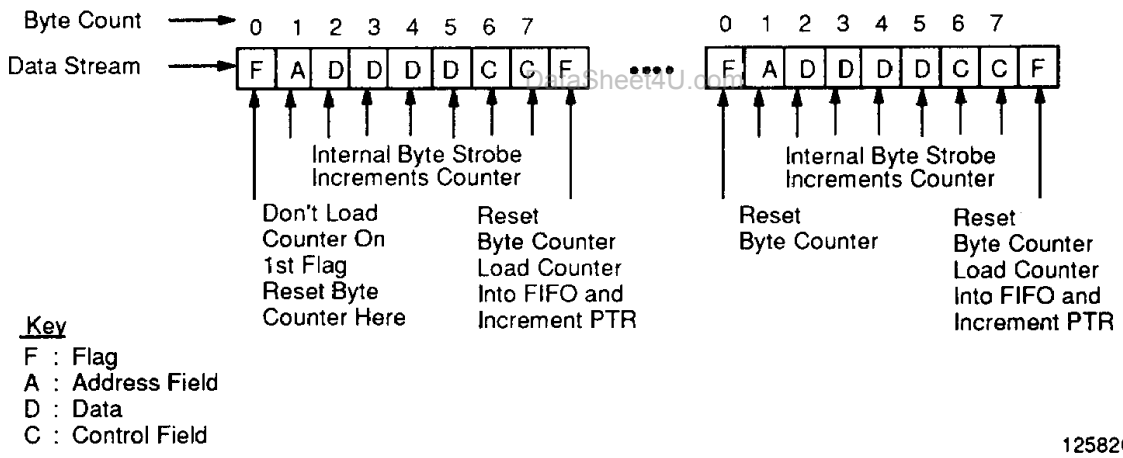


Figure 39. SDLC Byte Counting Detail

Byte Counter Detail

The 14-bit byte counter allows for packets up to 16K bytes to be received. For a better understanding of its operation, refer to Figures 38 and 39.

Enable

The byte counter is enabled when the ESCC is in the SDLC/HDLC mode and WR15 bit 2 is set to "1."

Reset

The byte counter is reset whenever an SDLC flag character is received. The reset is timed so that the contents of the byte counter are successfully written into the FIFO.

Increment

The byte counter is incremented by writes to the data FIFO. The counter represents the number of bytes received by the ESCC, rather than the number of bytes transferred from the ESCC. (These counts may differ by up to the number of bytes in the receive data FIFO contained in the ESCC.)

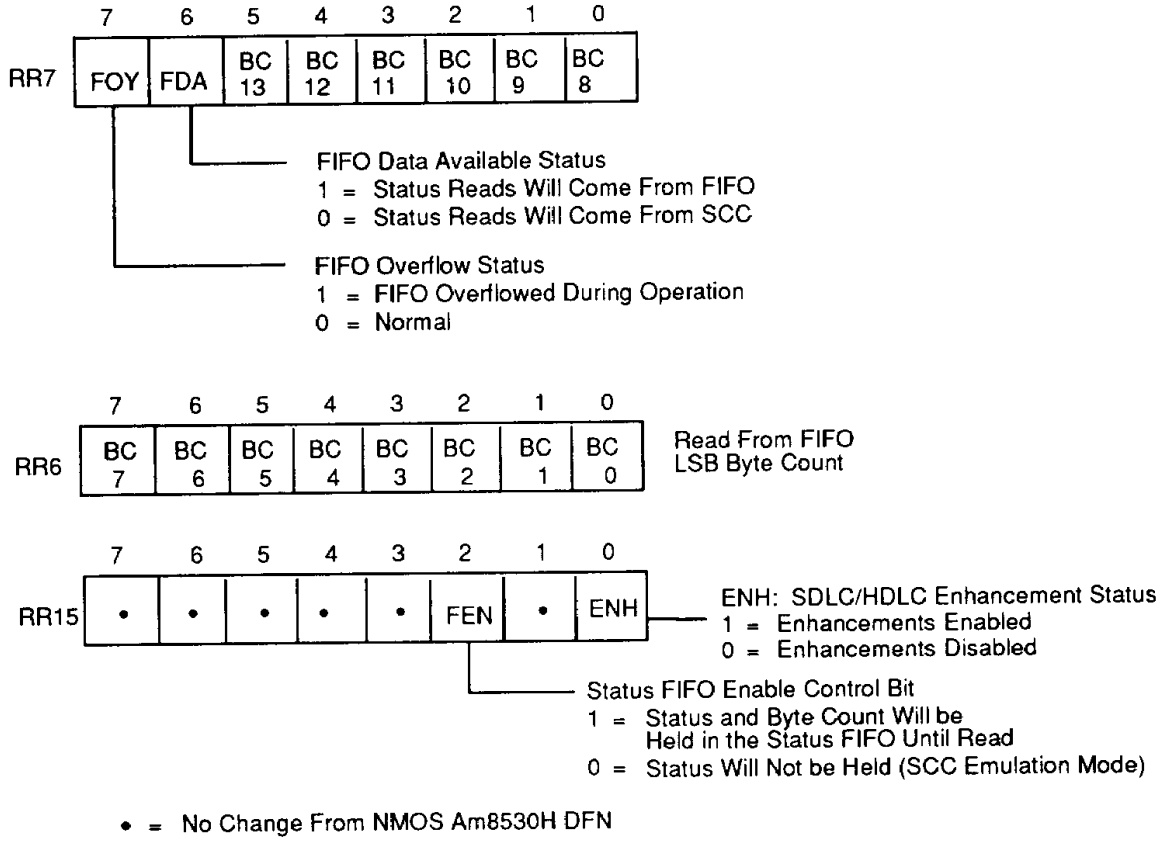


Figure 40. ESCC Additional Registers

ESCC SDLC/HDLC Enhancement Register Access

SDLC/HDLC enhancements on the ESCC are enabled or disabled via bits D2 or D0 in WR15. Bit D2 determines whether or not the 10 × 19 bit SDLC/HDLC frame status FIFO is enabled while bit D0 determines whether or not other enhancements are enabled via WR7′. Table 5 shows what functions on the ESCC are enabled when these bits are set.

Table 5. Enhancement Options

WR15 Bit D2 10 × 19 Bit FIFO Enabled	WR15 Bit D0 SDLC/HDLC Enhancement Enabled	WR7′ Bit D6 Extended Read Enabled	Functions Enabled
1	0	x	10 × 19 bit FIFO enhancement enabled only
0	1	0	SDLC/HDLC enhancements enabled only
0	1	1	SDLC/HDLC enhancements enabled with extended read enabled
1	1	0	10 × 19 bit FIFO and SDLC/HDLC enhancements enabled
1	1	1	10 × 19 bit FIFO and SDLC/HDLC enhancements with extended read enabled

When bit D2 of WR15 is set to "1," two additional registers (RR6 and RR7) per channel specific to the 10 × 19 bit Frame Status FIFO are made available. The ESCC register map when this function is enabled is shown in Table 6.

Table 6. 10 × 19 Bit FIFO Enabled

A/$\overline{B}$	PNT_2	PNT_1	PNT_0	Write	Read
0	0	0	0	WR0B	RR0B
0	0	0	1	WR1B	RR1B
0	0	1	0	WR2	RR2B
0	0	1	1	WR3B	RR3B
0	1	0	0	WR4B	(RR0B)
0	1	0	1	WR5B	(RR1B)
0	1	1	0	WR6B	RR6B
0	1	1	1	WR7B	RR7B
1	0	0	0	WR0A	RR0A
1	0	0	1	WR1A	RR1A
1	0	1	0	WR2	RR2A
1	0	1	1	WR3A	RR3A
1	1	0	0	WR4A	(RR0A)
1	1	0	1	WR5A	(RR1A)
1	1	1	0	WR6A	RR6A
1	1	1	1	WR7A	RR7A
With the Point High command:					
0	0	0	0	WR8B	RR8B
0	0	0	1	WR9	RR13B
0	0	1	0	WR10B	RR10B
0	0	1	1	WR11B	(RR15B)
0	1	0	0	WR12B	RR12B
0	1	0	1	WR13B	RR13B
0	1	1	0	WR14B	(RR10B)
0	1	1	1	WR15B	RR15B
1	0	0	0	WR8A	RR8A
1	0	0	1	WR9	(RR13A)
1	0	1	0	WR10A	RR10A
1	0	1	1	WR11A	(RR15A)
1	1	0	0	WR12A	RR12A
1	1	0	1	WR13A	RR13A
1	1	1	0	WR14A	(RR10A)
1	1	1	1	WR15A	RR15A

Bit D0 of WR15 determines whether or not other enhancements pertinent only to SDLC/HDLC Mode operation are available for programming via WR7′ as shown below. Write Register 7 prime (WR7′) can be written to when bit D0 of WR15 is set to "1." When this bit is set, writing to WR7 (flag register) actually writes to WR7′. If bit D6 of this register is set to "1," previously unreadable registers WR3, WR4, WR5, and WR10 are readable by the processor. In addition, WR7′ is also readable by having this bit set. WR3 is read when a bogus RR9 register is accessed during a read cycle. WR10 is read by accessing RR11, and WR7′ is accessed by executing a read to RR14. The ESCC register map with bit D0 of WR15 and bit D6 of WR7′ set is shown in Table 7.

D7	D6	D5	D4	D3	D2	D1	D0
Must Be Set to 0	Ext. Read Enable	Rx comp. CRC	$\overline{DTR}/\overline{REQ}$ Fast Mode	Force TxD High	SDLC/HDLC Auto $\overline{RTS}$ Turnoff	SDLC/HDLC Auto EOM Reset	SDLC/HDLC Auto Tx Flag

WR7′—SDLC/HDLC Programmable Enhancements*

***Note:**
Options 3, 4, 5, and 6 may be used regardless of whether SDLC/HDLC mode is selected.

Table 7. SDLC/HDLC Enhancements Enabled

A/$\overline{B}$	PNT_2	PNT_1	PNT_0	Write	Read
0	0	0	0	WR0B	RR0B
0	0	0	1	WR1B	RR1B
0	0	1	0	WR2	RR2B
0	0	1	1	WR3B	RR3B
0	1	0	0	WR4B	RR4B (WR4B)
0	1	0	1	WR5B	RR5B (WR5B)
0	1	1	0	WR6B	(RR2B)
0	1	1	1	WR7B	(RR3B)
1	0	0	0	WR0A	RR0A
1	0	0	1	WR1A	RR1A
1	0	1	0	WR2	RR2A
1	0	1	1	WR3A	RR3A
1	1	0	0	WR4A	RR4A (WR4A)
1	1	0	1	WR5A	RR5A (WR5A)
1	1	1	0	WR6A	(RR2A)
1	1	1	1	WR7A	(RR3A)
With the Point High command:					
0	0	0	0	WR8B	RR8B
0	0	0	1	WR9	RR9 (WR3B)
0	0	1	0	WR10B	RR10B
0	0	1	1	WR11B	RR11B (WR10B)
0	1	0	0	WR12B	RR12B
0	1	0	1	WR13B	RR13B
0	1	1	0	WR14B	RR14B (WR7'B)
0	1	1	1	WR15B	RR15B
1	0	0	0	WR8A	RR8A
1	0	0	1	WR9	RR9A (WR3A)
1	0	1	0	WR10A	RR10A
1	0	1	1	WR11A	RR11A (WR10A)
1	1	0	0	WR12A	RR12A
1	1	0	1	WR13A	RR13A
1	1	1	0	WR14A	RR14A (WR7A)
1	1	1	1	WR15A	RR15A

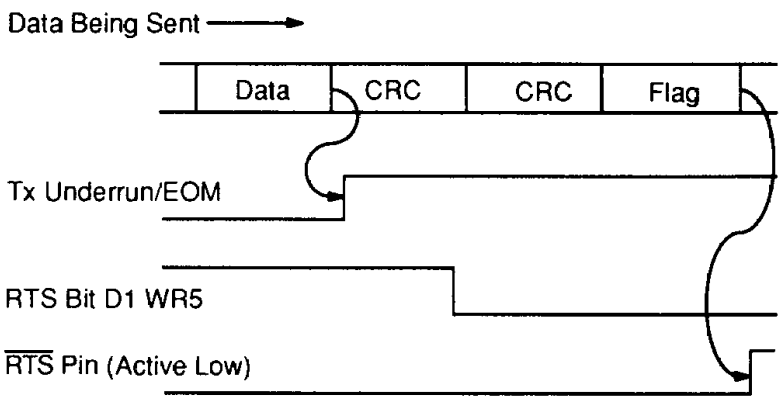
If both bits D0 and D2 of WR15 are set to "1" and D6 of WR7′ is set to "1," then the ESCC register map is as shown in Table 8.

Table 8. SDLC/HDLC Enhancements and 10 × 19 Bit FIFO Enabled

A/$\overline{B}$	PNT_2	PNT_1	PNT_0	Write	Read
0	0	0	0	WR0B	RR0B
0	0	0	1	WR1B	RR1B
0	0	1	0	WR2	RR2B
0	0	1	1	WR3B	RR3B
0	1	0	0	WR4B	RR4B (WR4B)
0	1	0	1	WR5B	RR5B (WR5B)
0	1	1	0	WR6B	RR6B
0	1	1	1	WR7B	RR7B
1	0	0	0	WR0A	RR0A
1	0	0	1	WR1A	RR1A
1	0	1	0	WR2	RR2A
1	0	1	1	WR3A	RR3A
1	1	0	0	WR4A	RR4A (WR4A)
1	1	0	1	WR5A	RR5A (WR5A)
1	1	1	0	WR6A	RR6A
1	1	1	1	WR7A	RR7A
With the Point High command:					
0	0	0	0	WR8B	RR8B
0	0	0	1	WR9	RR9 (WR3B)
0	0	1	0	WR10B	RR10B
0	0	1	1	WR11B	RR11B (WR10B)
0	1	0	0	WR12B	RR12B
0	1	0	1	WR13B	RR13B
0	1	1	0	WR14B	RR14B (WR7′B)
0	1	1	1	WR15B	RR15B
1	0	0	0	WR8A	RR8A
1	0	0	1	WR9	RR9A (WR3A)
1	0	1	0	WR10A	RR10A
1	0	1	1	WR11A	RR11A (WR10A)
1	1	0	0	WR12A	RR12A
1	1	0	1	WR13A	RR13A
1	1	1	0	WR14A	RR14A (WR7′A)
1	1	1	1	WR15A	RR15A

Auto $\overline{RTS}$ Reset

On the CMOS ESCC, if bit D0 of WR15 and bit D2 of $\overline{WR7'}$ are set to "1" and the channel is in SDLC Mode, the $\overline{RTS}$ pin may be reset early in the Tx Underrun routine and the $\overline{RTS}$ pin will remain active until the last zero bit of the closing flag leaves the TxD pin as shown in Figure 16. Note that in order for this to function properly, bits D3 and D2 of WR10 must be set to "1" and "0" respectively.



12582C-069A

Figure 41. Auto $\overline{RTS}$ Reset Mode

CRC Character Reception

NMOS Am8530H

On the NMOS Am8530H , when the end-of-frame flag is detected, the contents of the Receive Shift Register are transferred to the Receive Data FIFO regardless of the number of bits accumulated. Because of the 3-bit delay between the Receive SYNC Register and Receive Shift Register, the last two bits of the CRC check character received are never transferred to the Receive Data FIFO. Thus, the received CRC characters are unavailable for use.

CMOS ESCC

On the ESCC, the option of being able to receive the complete CRC characters generated by the Transmitter is provided when both bit D0 of WR15 and bit D5 of WR7′ are set to "1." When these two bits are set and an end-of-frame flag is detected, the last two bits of the CRC will be clocked into the Receive Shift Register before its contents are transferred to the Receive Data FIFO.The data-CRC boundary and CRC character bit formatsfor each Residue Code provided is shown in Figures 42A through 42D for each character length selected.

Residue Code 012/001

D	D	D	D	D	C_0	C_1	C_2
C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/101

D	D	D	D	D	D	C_0	C_1
D	C_0	C_1	C_2	C_3	C_4	C_5	C_6
C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/100

D	D	D	D	D	D	D	C_0
D	D	C_0	C_1	C_2	C_3	C_4	C_5
C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/010

D	D	D	D	D	D	D	D
D	D	D	C_0	C_1	C_2	C_3	C_4
C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/110

D	D	D	D	D	D	D	D
D	D	D	D	C_0	C_1	C_2	C_3
C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

12582C–070A

Figure 42A. 5 Bits/Character

Residue Code 012/010

D	D	D	D	D	D	C_0	C_1
C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/110

D	D	D	D	D	D	D	C_0
D	C_0	C_1	C_2	C_3	C_4	C_5	C_6
C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/001

D	D	D	D	D	D	D	D
D	D	C_0	C_1	C_2	C_3	C_4	C_5
C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/101

D	D	D	D	D	D	D	D
D	D	D	C_0	C_1	C_2	C_3	C_4
C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/011

D	D	D	D	D	D	D	D
D	D	D	D	C_0	C_1	C_2	C_3
C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/100

D	D	D	D	D	D	D	D
D	D	D	D	D	C_0	C_1	C_2
C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

12582C–071A

Figure 42B. 6 Bits/Character

Residue Code 012/111

D	D	D	D	D	D	D	C_0
C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/100

D	D	D	D	D	D	D	D
D	C_0	C_1	C_2	C_3	C_4	C_5	C_6
C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/010

D	D	D	D	D	D	D	D
D	D	C_0	C_1	C_2	C_3	C_4	C_5
C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/110

D	D	D	D	D	D	D	D
D	D	D	C_0	C_1	C_2	C_3	C_4
C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/001

D	D	D	D	D	D	D	D
D	D	D	D	C_0	C_1	C_2	C_3
C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/101

D	D	D	D	D	D	D	D
D	D	D	D	D	C_0	C_1	C_2
C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/011

D	D	D	D	D	D	D	D
D	D	D	D	D	D	C_0	C_1
C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

12582C–072A

Figure 42C. 7 Bits/Character

Residue Code 012/011 (No Residue)

D	D	D	D	D	D	D	D
C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/111 (1 Residue Bit)

D	D	D	D	D	D	D	D
D	C_0	C_1	C_2	C_3	C_4	C_5	C_6
C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/000 (2 Residue Bits)

D	D	D	D	D	D	D	D
D	D	C_0	C_1	C_2	C_3	C_4	C_5
C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/100 (3 Residue Bits)

D	D	D	D	D	D	D	D
D	D	D	C_0	C_1	C_2	C_3	C_4
C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/010 (4 Residue Bits)

D	D	D	D	D	D	D	D
D	D	D	D	C_0	C_1	C_2	C_3
C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/110 (5 Residue Bits)

D	D	D	D	D	D	D	D
D	D	D	D	D	C_0	C_1	C_2
C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/001 (6 Residue Bits)

D	D	D	D	D	D	D	D
D	D	D	D	D	D	C_0	C_1
C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

Residue Code 012/101 (7 Residue Bits)

D	D	D	D	D	D	D	D
D	D	D	D	D	D	D	C_0
C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}

12582C–073A

Figure 42D. 8 Bits/Character

Auto Flag Mode

On the NMOS Am8530H, if the transmitter is actively mark idling and a frame of data is ready to be transmitted, the MARK/FLAG idle bit must be set to "0" before data is written to WR8, otherwise the opening flag will not be sent properly. However, care must be exercised in doing this because the mark idle pattern (eight "1" bits) is transmitted eight bits at a time, and all eight bits must have transferred out the Transmit Shift Register before a flag may be loaded and sent. If data is written into the Transmit Buffer (WR8) before the flag is loaded into the Transmit Shift Register, the data character written to WR8 will supersede flag transmission and the opening flag will not be transmitted.

On the CMOS ESCC, if bit D0 of WR15 is set to "1," and the ESCC is programmed for SDLC operation, an option is provided via bit D0 of WR7′ that eliminates this requirement. If bit D0 of WR7′ is set to "1" and a character is written to the Transmit Buffer while the Transmitter is mark idling, the Mark/Flag Idle bit in WR10 need not be reset to "0" in order to have the opening flag sent because the Transmitter will automatically send it before commencing to send data.

In addition, as long as bit D0 of WR15 and bit D1 of WR7′ are set to "1," the CRC transmit generator will be automatically preset to the initial state programmed by bit D7 of WR10 (so the Reset Tx CRC Generator command is also not necessary), and the Tx Underrun/EOM latch will be reset automatically on every new frame sent. This ensures that an opening flag and proper CRC generation and transmission will always be sent without processor intervention under varying bus latency conditions.

Auto Transmit CRC Generator Preset

The NMOS Am8530H does not automatically preset the CRC generator prior to frame transmission. This must be done in software, usually during the initialization routine. This is accomplished by issuing the Reset Tx CRC Generator Command via WR0. For proper results, this command must be issued while the transmitter is enabled and idling and before any data are written to the Transmit Buffer.

In addition, if CRC is to be used, the transmit CRC generator must be enabled by setting bit D0 of WR5 to "1." CRC is normally calculated on all characters between opening and closing flags, so this bit should be set to "1" at initialization and never changed.

On the CMOS ESCC, setting bit D0 of WR15 to "1" will cause the transmit CRC generator to be preset automatically every time an opening flag is sent, so the Reset Tx CRC Generator Command is not necessary.

Auto Tx Underrun/EOM Latch Reset

On the ESCC, the transmission of the CRC check characters is controlled by the Transmit CRC Enable bit in WR5 (D0) and the Tx Underrun/EOM bit in RR0 (D6). However, if the Transmit Enable bit is set to "0" when a transmit underrun (i.e., both the Transmit Buffer and Transmit Shift Register go empty) occurs, the CRC check characters will not be sent regardless of the state of the Tx Underrun/EOM bit.

If the Transmit Enable bit is set to "1" when an underrun occurs, then the state of the Tx Underrun/EOM bit and the Abort/Flag on Underrun bit in WR10 (D2) determine the action taken by the Transmitter. The Abort/Flag on Underrun bit may be set or reset by the processor, whereas, the Tx Underrun/EOM bit is set by the Transmitter and can only be reset by the processor via the Reset Tx Underrun/EOM Command in WR0.

If the Tx Underrun/EOM bit is set to "1" when an underrun occurs, the Transmitter will close the frame by sending a flag; however, if this bit is set to "0," the frame data will be appended with either the accumulated CRC characters followed by a flag or an abort pattern followed by a flag, depending on the state of the Abort/Flag on Underrun bit in the WR10 (D2). In either case, after the closing flag is sent, the Transmitter will idle the transmission line as specified by the Mark/Flag Idle bit D3 in WR10.

Hence, if the CRC check characters are to be properly appended to a frame, the Abort/Flag on Underrun bit must be set to "0," and the Reset Tx Underrun/EOM Command must be issued after the first but before the last character is written to the Transmit Buffer. This will ensure that either an abort or the CRC will be transmitted if an underrun occurs. Normally, the Abort/Flag on Underrun bit in WR10 should be set to "1" around the same time that the Tx Underrun/EOM bit is reset so that an abort will be sent if the transmitter accidentally underruns, and then set to "0" near the end of the frame to allow the correct transmission of CRC.

On the ESCC, if bit D0 of WR15 is set to "1," the option of having the Tx Underrun/EOM bit reset automatically at the start of every frame is provided via bit D1 of WR7′. This helps alleviate the software burden of having to respond within one character time when high-speed data are being sent.

SDLC/HDLC NRZI Transmitter Disabling

On the NMOS Am8530H, if NRZI encoding is being used and the Transmitter is disabled, the state of the TxD pin will depend on the last bit sent. That is, the TxD pin may either idle in a Low or High state as shown in Figure 43.

On the CMOS ESCC, an option is provided that allows setting the TxD pin High when operating in SDLC Mode with NRZI encoding enabled. If bit D0 of WR15 is set to "1," then bit D3 of WR7′ can be used to set the TxD pin High. Note that the operation of this bit is independent of the Tx Enable bit in WR5. The Tx Enable bit in WR5 is used to disable and enable the transmitter, whereas bit D3 of WR7′ acts as a pseudo transmitter disable and enable by just forcing the TxD pin High when set even though the transmitter may actually be mark or flag idling. Care must be used when setting this bit because any character being transmitted at the time this bit is set will be "chopped off," and data written to the Transmit Buffer while this bit is set will be lost.

When the transmit underrun occurs and the CRC and closing flag have been sent, bit D3 can be set to pull TxD High. When ready to start sending data again this bit must be reset to "0" before the first character is written to the Transmit Buffer. Note that resetting this bit causes the TxD pin to take whatever state the NRZI encoder is in at the time so synchronization at the Receiver may take longer because the first transition seen on the TxD pin may not coincide with a bit boundary. Note that in order for this to function properly, bits D3 and D2 of WR10 must be set to "1" and "0" respectively.

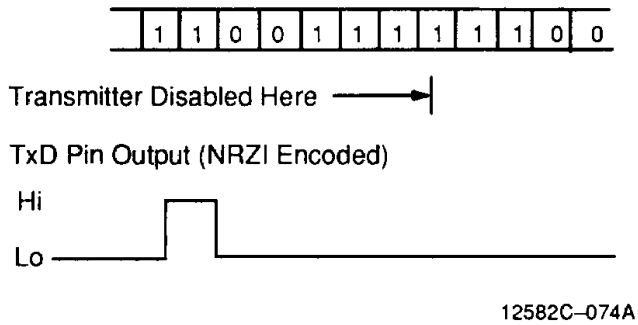


Figure 43. Transmitter Disabling with NRZI Encoding

Interrupt Masking Without $\overline{\text{INTACK}}$

The NMOS Am8530H's ability to mask lower priority interrupts is done via the IUS bit. This bit is internal to the ESCC and is not observable by the processor. Being able to automatically mask lower priority interrupts allows a modular approach to coding interrupt routines. However, using the masking capabilities of the NMOS ESCC requires that the $\overline{\text{INTACK}}$ cycle be generated. In stand-alone applications, having to generate $\overline{\text{INTACK}}$ through external hardware in order to use this capability is an unnecessary expense.

On the CMOS ESCC, if bit D5 in WR9 is set to "1," the $\overline{\text{INTACK}}$ cycle does not need to be generated in order to have the IUS bit set. This allows the user to respond to ESCC interrupt requests with a software acknowledgment through RR2. When bit D5 in WR9 is set and an interrupt occurs, a read to RR2 emulates a hardware Interrupt Acknowledge cycle as it functions in Vectored Mode. In this case the CPU must first read RR2 to determine the internal interrupt source and then jump to the appropriate interrupt routine. Reading RR2 sets the IUS bit for the highest priority IP. After the interrupting condition is cleared, the routine can then read RR3 to determine if any other IPs are set and clear them. At the end of the interrupt routine, a Reset IUS Command must be issued to unlock the internal daisy chain.

Since the CPU can acknowledge the ESCC of highest priority with a read of its RR2 interrupt vector, there is no need for an external daisy chain. When acknowledging an ESCC interrupt request, the CPU must issue one read to RR2 per interrupt request. The modified interrupt vector can be read from Channel B, or the original vector stored in WR2 can be read from Channel A. Either action will produce the same internal actions on the IUS logic. Note that the No Vector and Vector Includes Status bits in WR9 are ignored when bit D5 in WR9 is set to "1."

1 Mb/s FM Data Transmission and Reception

The 8-MHz version of the CMOS ESCC (ESCC-8) is capable of transmitting and receiving FM-encoded data at the rate of 1 Mb/s. This is accomplished by applying a 16-MHz clock to the $\overline{\text{RTxC}}$ pin and assigning this waveform to drive the Internal Digital Phase Locked Loop (DPLL) clock. This feature allows the user to send both clock and data information over the same line at 1 Mb/s and can eliminate external DPLLs required for high-speed NRZ data clock generation.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	−65 to +150°C
Voltage at any Pin Relative to V_{SS}	−0.5 to +7.0 V

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Ambient Temperature (T_A)	0 to +70°C
Supply Voltage (V_{CC})	+5 V ± 5%

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating ranges

Parameter Symbol	Parameter Description	Test Conditions	Min.	Max.	Unit
V_{IH}	Input High Voltage	Commercial	2.2	V_{CC} +0.3*	V
V_{IL}	Input Low Voltage		−0.3*	0.8	V
V_{OH1}	Output High Voltage	I_{OH} = −3.0 mA	2.4		V
V_{OH2}	Output High Voltage	I_{OH} = −250 µA (Note 2)	V_{CC}−0.8		V
V_{OL1}	Output Low Voltage	I_{OL} = +2.4 mA		0.5	V
V_{OL2}	Output Low Voltage	I_{OL} = +48 mA (Note 1)		0.5	V
I_{IL}	Input Leakage	0.4 V ≤ V_{IN} ≤ 2.4 V		±10.0	µA
	SCSI Bus Pins Except $\overline{RST}$	V_{IH} = 5.25 V, V_{IL} = 0		±50	µA
I_{OL}	Output Leakage	0.4 V ≤ V_{OUT} ≤ 2.4 V		±10.0	µA
I_{CC}	V_{CC} Supply Current	CLK = 8 MHz, inputs at voltage rails, output unloaded		40	mA
C_{IN}	Input Capacitance	Unmeasured pins returned to ground. f = 1 MHz over specified temperature range		10	pF
C_{OUT}	Output Capacitance			15	pF
C_{MO}	Bidirectional Capacitance Except SCSI Bus Pins			20	pF

* V_{IH} Max. and V_{IL} Min. not tested. Guaranteed by design.

Notes:

1. SCSI Bus Pins only.
2. SCC outputs only.

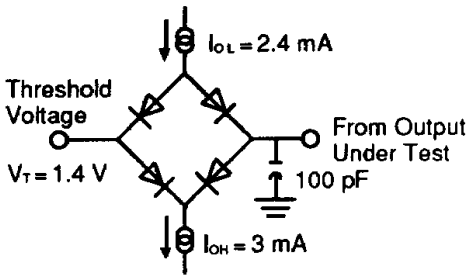
Standard Test Conditions

The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

+4.75 V ≤ V_{CC} ≤ +5.25 V
GND = 0 V
0°C ≤ T_A ≤ 70°C

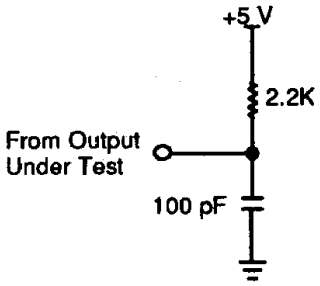
SWITCHING TEST CIRCUITS

Standard Test Dynamic Load Circuit



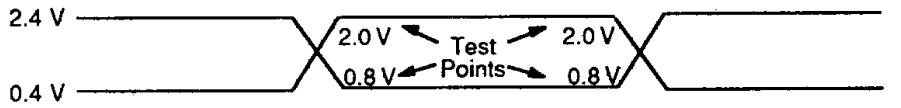
12582C–075A

Open-Drain Test Load



12582C–076A

SWITCHING TEST INPUT/OUTPUT WAVEFORM



12582C–077A

AC testing: Inputs are driven at 2.4 V for a logic "1" and 0.4 V for a logic "0."
Timing measurements are made at 2.0 V for a logic "1" and 0.8 V for logic "0."

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	Must be Steady	Will be Steady
	May Change from H to L	Will be Changing from H to L
	May Change from L to H	Will be Changing from L to H
	Don't Care, Any Change Permitted	Changing, State Unknown
	Does Not Apply	Center Line is High-Impedance "Off" State

ESCC SWITCHING CHARACTERISTICS over COMMERCIAL operating range
General Timing (see Figure 44)

No.	Parameter Symbol	Parameter Description	8.192 MHz		16.384 MHz		Unit
			Min.	Max.	Min.	Max.	
1	TdPC(REQ)	PCLK ↓ to W/REQ Valid Delay		250		80	ns
2	TdPC(W)	PCLK ↓ to Wait Inactive Delay		350		180	ns
3	TsRXC(PC)	RxC ↑ to PCLK ↑ Setup Time (Notes 1, 4 & 8)	NA	NA	NA	NA	
4	TsRXD(RXCr)	RxD to RxC ↑ Setup Time (XI Mode) (Note 1)	0		0		ns
5	ThRXD(RXCr)	RxD to RxC ↑ Hold Time (XI Mode) (Note 1)	150		50		ns
6	TsRXD(RXCf)	RxD to RxC ↓ Setup Time (XI Mode) (Notes 1, 5)	0		0		ns
7	ThRXD(RXCf)	RxD to RxC ↓ Hold Time (XI Mode) (Notes 1, 5)	150		50		ns
8	TsSY(RXC)	SYNCA to RxC ↑ Setup Time (Note 1)	−200		−100		ns
9	ThSY(RXC)	SYNCA to RxC ↑ Hold Time (Note 1)	5TcPc		5TcPc		ns
10	TsTXC(PC)	TxC ↓ to PCLK ↑ Setup Time (Notes 2, 4 & 8)	NA		NA		
11	TdTXCf(TXD)	TxC ↓ to TxD Delay (XI Mode) (Note 2)		200		80	ns
12	TdTXCr(TXD)	TxC ↑ to TxD Delay (XI Mode) (Notes 2, 5)		200		80	ns
13	TdTXD(TRX)	TxD to TRxC Delay (Send Clock Echo)		200		80	ns
14a	TwRTXh	RTxC High Width (Note 6)	150		80		ns
14b	TwRTxh(E)	RTxC High Width (Note 9)	50		15.6		ns
15a	TwRTXl	RTxC Low Width (Note 6)	150		80		ns
15b	TwRTXl(E)	RTxC Low Width (Note 9)	50		15.6		ns
16a	TcRTX	RTxC Cycle Time (Notes 6, 7)	488		244		ns
16b	TcRTx(E)	RTxC Cycle Time (Note 9)	125		31.25		ns
17	TcRTXX	Crystal Oscillator Period (Note 3)	125	1000	62	1000	ns
18	TwTRXh	TRxC High Width (Note 6)	150		80		ns
19	TwTRXl	TRxC Low Width (Note 6)	150		80		ns
20	TcTRX	TRxC Cycle Time (Notes 6, 7)	488		244		ns
21	TwEXT	DCD Pulse Width	200		70		ns
22	TwSY	SYNCA Pulse Width	200		70		ns

Notes:

1. RxC is RTxC or TRxC, whichever is supplying the receive clock.
2. TxC is TRxC or RTxC, whichever is supplying the transmit clock.
3. Both RTxC and SYNCA have 30-pF capacitors to ground connected to them.
4. Parameter applies only if the data rate is one-fourth the PCLK rate. In all other cases, no phase relationship between RxC and PCLK or TxC and PCLK is required.
5. Parameter applies only to FM encoding/decoding.
6. Parameter applies only for transmitter and receiver, DPLL and baud rate generator timing requirements are identical to chip PCLK requirements.
7. The maximum receive or transmit data is 1/4 PCLK.
8. External PCLK to RxC or TxC synchronization requirement eliminated for PCLK divide-by-four operation. TRxC and RTxC rise and fall times are identical to PCLK. Reference timing specs Tfpc and Trpc. Tx and Rx input clock slow rates should be kept to a maximum of 30 nsec. All parameters related to input CLK edges should be referenced at the point at which the transition begins or ends, whichever is worst case.
9. ENHANCED FEATURE— RTxC used as input to internal DPLL only.

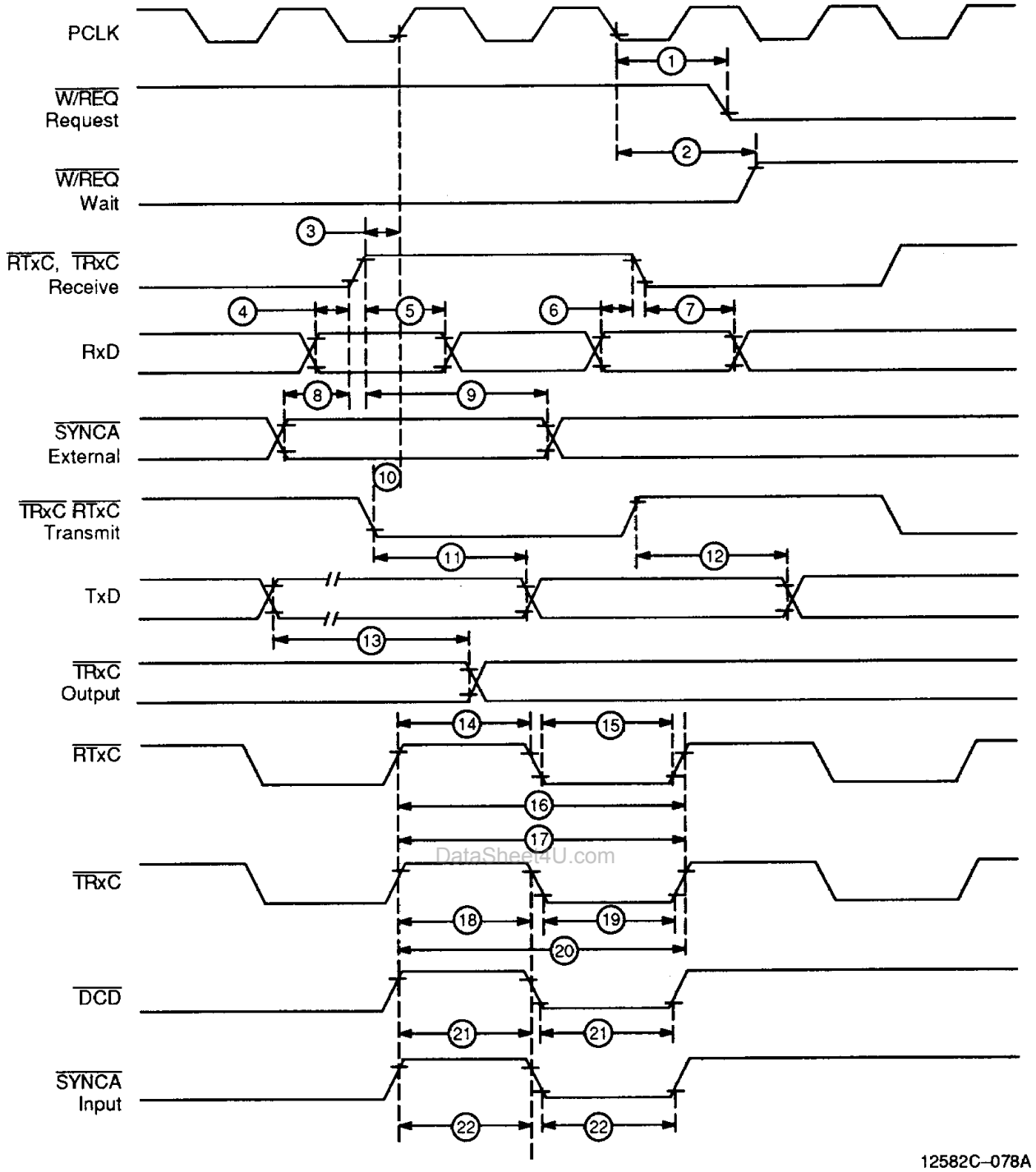


Figure 44. General Timing

ESCC SWITCHING CHARACTERISTICS over COMMERCIAL operating range (continued)
System Timing (see Figure 45)

No.	Parameter Symbol	Parameter Description	8.192 MHz		16.384 MHz		Unit
			Min.	Max.	Min.	Max.	
1	TdRXC(REQ)	$\overline{RXC}$ ↑ $\overline{W/REQ}$ Valid Delay (Note 2)	8	12	8	12	TcPc
2	TdRXC(W)	$\overline{RXC}$ ↑ to Wait Inactive Delay (Notes 1, 2)	8	14	8	14	TcPc
3	TdRXC(SY)	$\overline{RxC}$ ↑ to $\overline{SYNC}$ Valid Delay (Note 2)	4	7	4	7	TcPc
4	TdRXC(INT)	$\overline{RxC}$ ↑ to $\overline{INT}$ Valid Delay (Notes 1, 2)	10	16	10	16	TcPc
5	TdTXC(REQ)	$\overline{TxC}$ ↑ to $\overline{W/REQ}$ Valid Delay (Note 3)	5	8	5	8	TcPc
6	TdTXC(W)	$\overline{TxC}$ ↓ to Wait Inactive Delay (Notes 1, 3)	5	11	5	11	TcPc
7a	TdTXC(DRQ)	$\overline{TxC}$ ↓ to $\overline{DTR/REQ}$ Valid Delay (Note 3)	4	7	4	7	TcPc
7b	TdTXC(EDRQ)	$\overline{TxC}$ ↓ to $\overline{DTR/REQ}$ Valid Delay (Notes 3, 4)	5	8	5	8	TcPc
8	TdTXC(INT)	$\overline{TxC}$ ↓ to $\overline{INT}$ Valid Delay (Notes 1, 3)	6	10	6	10	TcPc
9	TdSY(INT)	$\overline{SYNCA}$ Transition to $\overline{INT}$ Valid Delay (Note 1)	2	6	2	6	TcPc
10	TdEXT(INT)	$\overline{DCD}$ Transition to $\overline{INT}$ Valid Delay (Note 1)	2	6	2	6	TcPc

Notes:

1. Open-drain output, measured with open-drain test load.
2. $\overline{RxC}$ is $\overline{RTxC}$ or $\overline{TRxC}$, whichever is supplying the receive clock.
3. $\overline{TxC}$ is $\overline{TRxC}$ or $\overline{RTxC}$, whichever is supplying the transmit clock.
4. Parameter applies to Enhanced Request mode only.

ESCC SWITCHING CHARACTERISTICS over COMMERCIAL operating range (continued)
Read and Write Timing (see Figure 46)

No.	Parameter Symbol	Parameter Description	8.192 MHz		16.384 MHz		Unit
			Min.	Max.	Min.	Max.	
1	TwPCl	PCLK Low Width	50	2000	26	2000	ns
2	TwPCh	PCLK High Width	50	2000	26	2000	ns
3	TfPC	PCLK Fall Time		15		8	ns
4	TrPC	PCLK Rise Time		15		8	ns
5	TcPC	PCLK Cycle Time	122	4000	61	4000	ns
6	TsA(WR)	Address to $\overline{WR}$ ↓ Setup Time	70		35		ns
7	ThA(WR)	Address to $\overline{WR}$ ↑ Hold Time	0		0		ns
8	TsA(RD)	Address to $\overline{RD}$ ↓ Setup Time	70		35		ns
9	ThA(RD)	Address to $\overline{RD}$ ↑ Hold Time	0		0		ns
10	TsIA(PC)	$\overline{INTACK}$ to PCLK ↑ Setup Time	20		15		ns
11	TsIA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↓ Setup Time (Note 1)	145		70		ns
12	ThIA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
13	TsIA(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ Setup Time (Note 1)	145		70		ns
14	ThIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↑ Hold Time	0		0		ns
15	ThIA(PC)	$\overline{INTACK}$ to PCLK ↑ Hold Time	40		15		ns
16	TsCEl(WR)	$\overline{CE1}$ Low to $\overline{WR}$ ↓ Setup Time	0		0		ns
17	ThCE(WR)	$\overline{CE1}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
18	TsCEh(WR)	$\overline{CE1}$ High to $\overline{WR}$ ↓ Setup Time	60		30		ns
19	TsCEl(RD)	$\overline{CE1}$ Low to $\overline{RD}$ ↓ Setup Time (Note 1)	0		0		ns
20	ThCE(RD)	$\overline{CE1}$ to $\overline{RD}$ ↑ Hold Time (Note1)	0		0		ns
21	TsCEh(RD)	$\overline{CE1}$ High to $\overline{RD}$ ↓ Setup Time (Note 1)	60		30		ns
22	TwRDl	$\overline{RD}$ Low Width (Note 1)	150		75		ns
23	TdRD(DRA)	$\overline{RD}$ ↓ to Read Data Active Delay	0		0		ns
24	TdRDr(DR)	$\overline{RD}$ ↑ to Read Data Not Valid Delay	0		0		ns
25	TdRDf(DR)	$\overline{RD}$ ↓ to Read Data Valid Delay		140		70	ns
26	TdRD(DRz)	$\overline{RD}$ ↑ to Read Data Float Delay (Note 2)		40		20	ns

Notes:

1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float delay is defined as the time at which the data bus is released from its drive state with a maximum DC Load and minimum AC load.

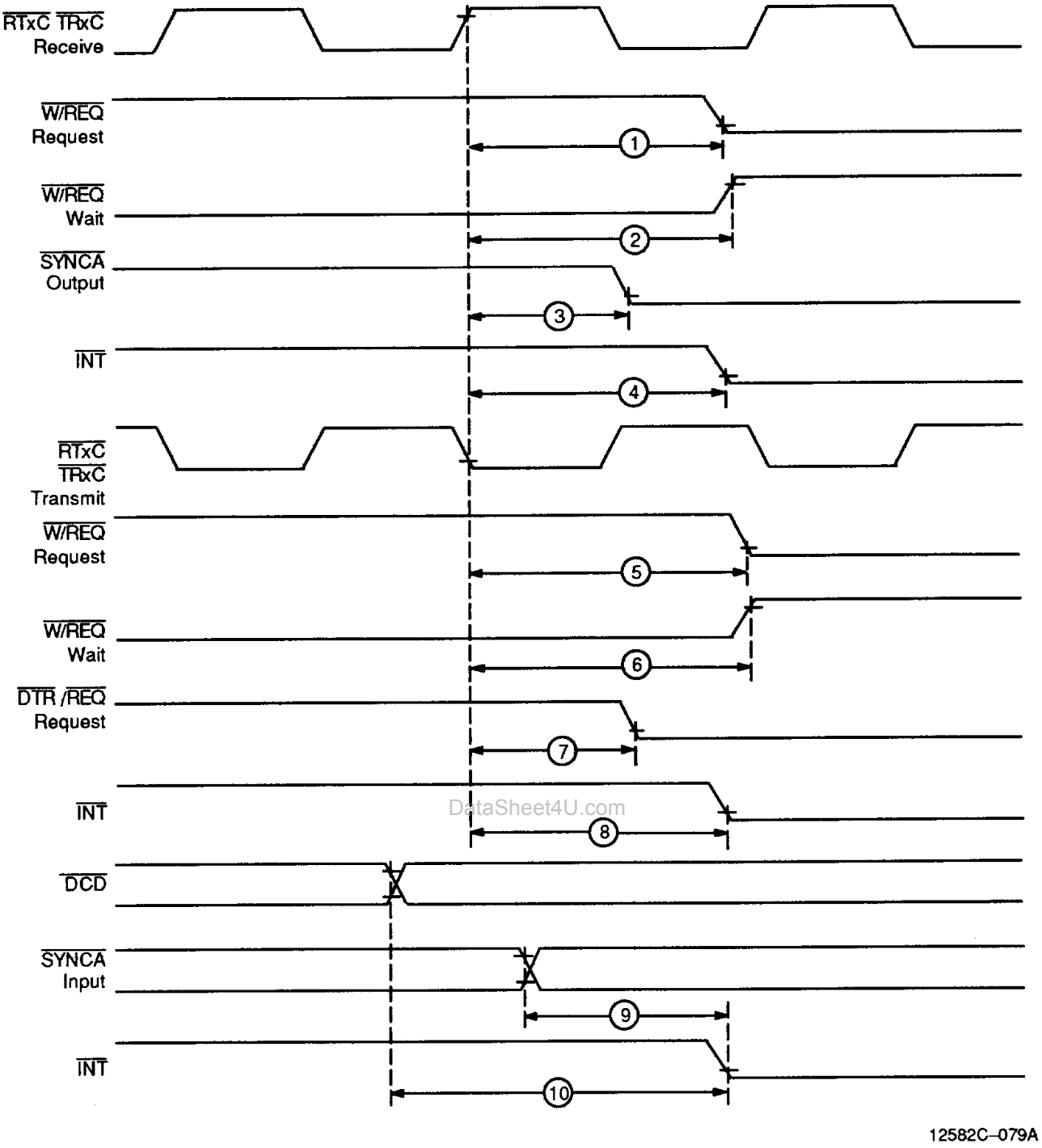
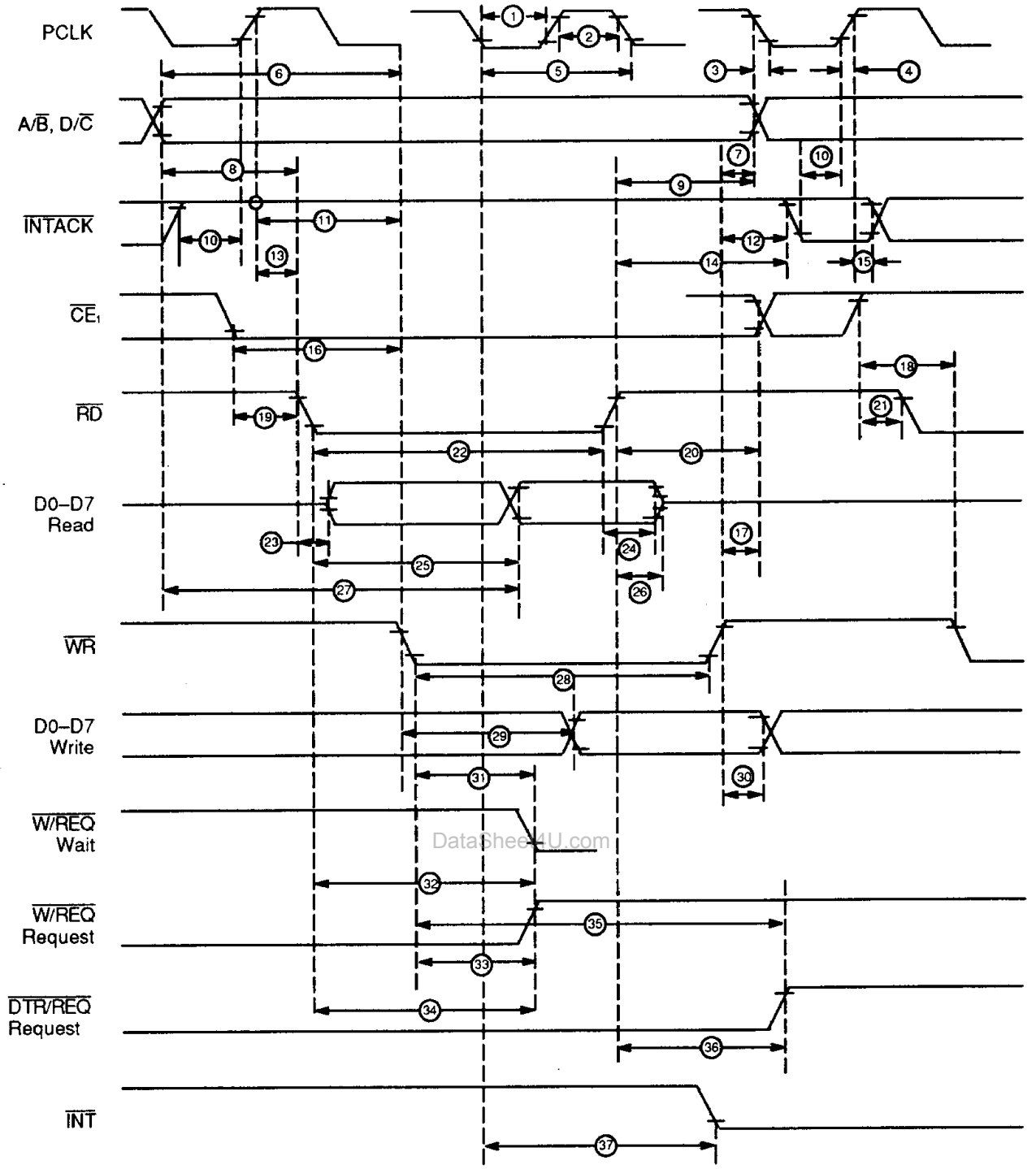


Figure 45. System Timing



12582C–080A

Figure 46. Read and Write Timing

ESCC SWITCHING CHARACTERISTICS over COMMERCIAL operating range (continued)
Interrupt Acknowledge Timing, Reset Timing, Cycle Timing (see Figures 47–49)

No.	Parameter Symbol	Parameter Description	8.192 MHz		16.384 MHz		Unit
			Min.	Max.	Min.	Max.	
27	TdA(DR)	Address Required Valid to Read Data Valid Delay		220		100	ns
28	TwWRI	$\overline{WR}$ Low Width	150		75		ns
29	TdWRf(DW)	$\overline{WR}$ ↓ to Write Data Valid		35		20	ns
30	ThDW(WR)	Write Data to $\overline{WR}$ ↑ Hold Time	0		10		ns
31	TdWR(W)	$\overline{WR}$ ↓ to Wait Valid Delay (Note 2)		170		50	ns
32	TdRD(W)	$\overline{RD}$ ↓ to Wait Valid Delay (Note 2)		170		50	ns
33	TdWRf(REQ)	$\overline{WR}$ ↓ to $\overline{W/REQ}$ Not Valid Delay		170		70	ns
34	TdRDf(REQ)	$\overline{RD}$ ↓ to $\overline{W/REQ}$ Not Valid Delay		170		70	ns
35a	TdWRr(REQ)	$\overline{WR}$ ↓ to $\overline{DTR/REQ}$ Not Valid Delay		4.0TcPc		4.0TcPc	ns
35b	TdWRr(EREQ)	$\overline{WR}$ ↓ to $\overline{DTR/REQ}$ Not Valid Delay		120		70	ns
36	TdRDr(REQ)	$\overline{RD}$ ↑ $\overline{DTR/REQ}$ Not Valid Delay		NA		NA	ns
37	TdPC(INT)	PCLK ↓ to $\overline{INT}$ Valid Delay (Note 2)		500		175	ns
38	TdIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ (Acknowledge) Delay	150		50		ns
39	TwRDA	$\overline{RD}$ (Acknowledge) Width	150		75		ns
40	TdRDA(DR)	$\overline{RD}$ ↓ (Acknowledge) to Read Data Valid Delay		140		70	ns
41	TsIEI(RDA)	IEI to $\overline{RD}$ ↓ (Acknowledge) Setup Time			50		ns
42	ThIEI(RDA)	IEI to $\overline{RD}$ ↑ (Acknowledge) Hold Time			0		ns
43	TdIE(IEO)	IEI to IEO Delay Time				45	ns
44	TdPC(IEO)	PCLK ↑ to IEO Delay				80	ns
45	TdRDA(INT)	$\overline{RD}$ ↓ to $\overline{INT}$ Inactive Delay (Note 2)		450		200	ns
46	TdRD(WRQ)	$\overline{RD}$ ↑ to $\overline{WR}$ ↓ Delay for No Reset	15		10		ns
47	TdWRQ(RD)	$\overline{WR}$ ↑ to $\overline{RD}$ ↓ Delay for No Reset	15		10		ns
48	TwRES	$\overline{WR}$ and $\overline{RD}$ Coincident Low for Reset	150		75		ns
49	Trc	Valid Access Recovery Time (Note 1)	3.5		3.5		TcPc

Notes:

1. Parameter applies only between transactions involving the ESCC, if $\overline{WR}$/$\overline{RD}$ falling edge is synchronized to PCLK falling edge, then TrC = 3TcPc.
2. Open-drain output, measured with open-drain test load.

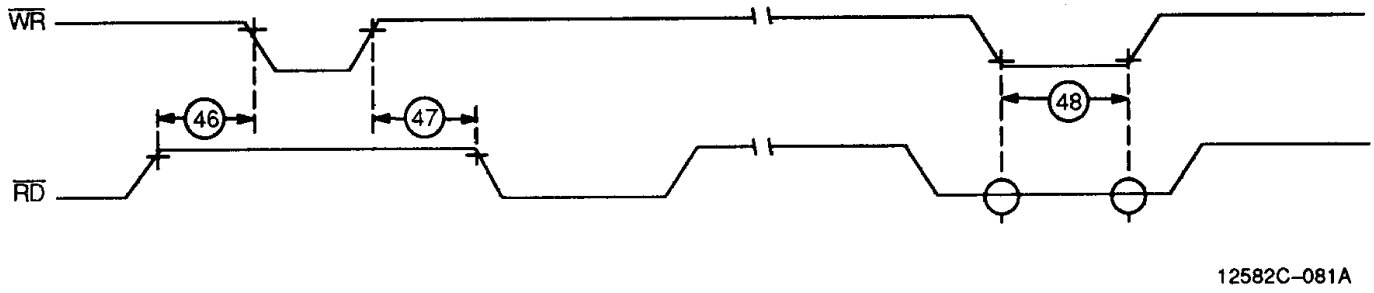


Figure 47. Reset Timing

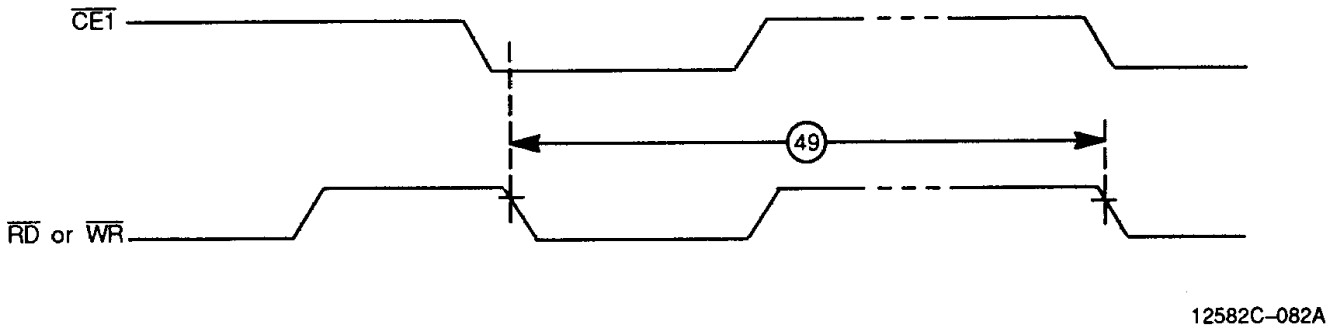


Figure 48. Cycle Timing

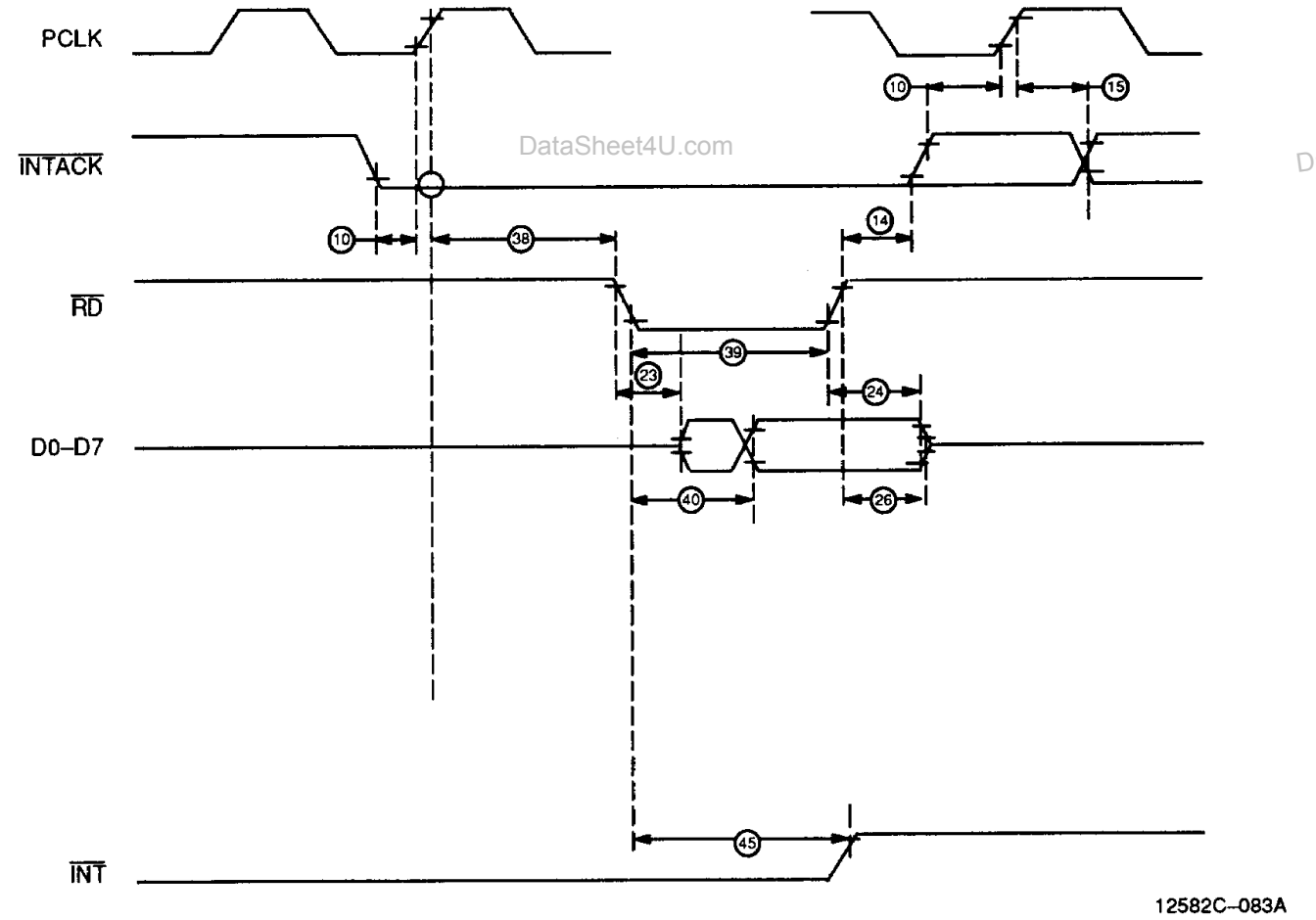


Figure 49. Interrupt Acknowledge Timing

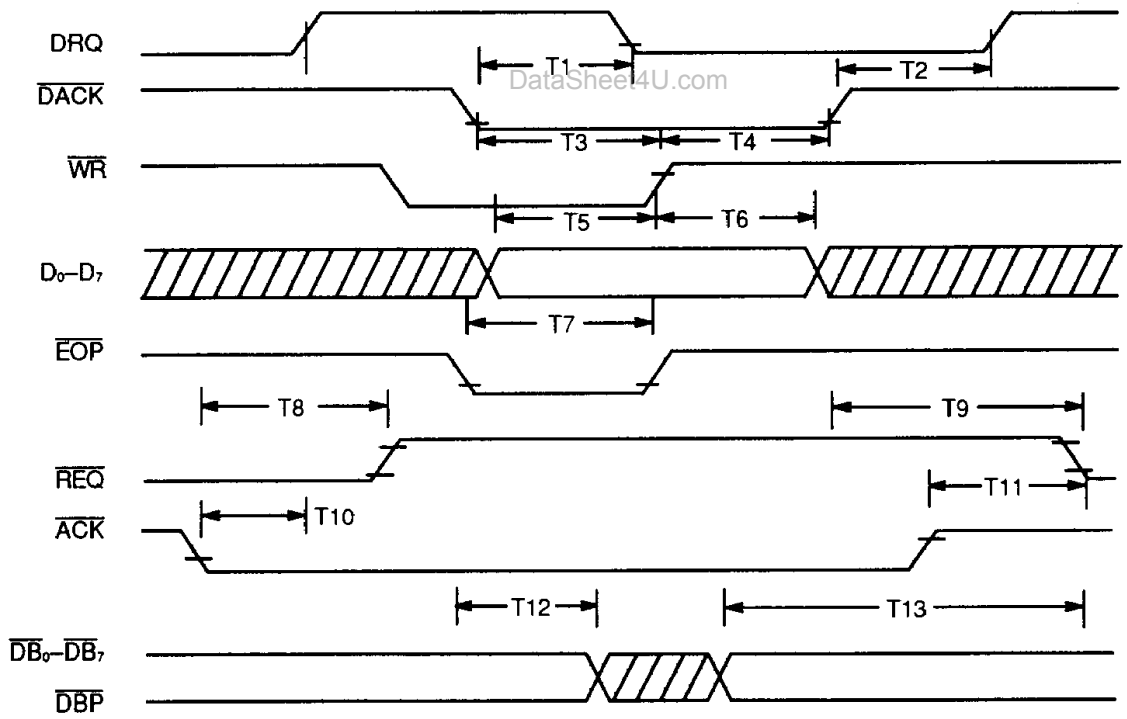
SCSI SWITCHING CHARACTERISTICS/WAVEFORMS
DMA Write (Non-Block Mode) Target Send Cycle (see Figure 50)

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	DRQ False from $\overline{DACK}$ True		100	ns
T2	$\overline{DACK}$ False to DRQ True	30		ns
T3	Write Enable Width*	70		ns
T4	DACK Hold from End of $\overline{WR}$	0		ns
T5	Data Setup to End of Write Enable*	30		ns
T6	Data Hold Time from End of $\overline{WR}$	40		ns
T7	Width of $\overline{EOP}$ Pulse (Note 1)	70		ns
T8	$\overline{ACK}$ True to $\overline{REQ}$ False		125	ns
T9	$\overline{REQ}$ from End of $\overline{DACK}$ ($\overline{ACK}$ False)		120	ns
T10	$\overline{ACK}$ True to DRQ True (Target)		110	ns
T11	$\overline{REQ}$ from End of $\overline{ACK}$ ($\overline{DACK}$ False)		120	ns
T12	Data Hold from Write Enable	*0		ns
T13	Data Setup to $\overline{REQ}$ True (Target)	60		ns

*Write Enable is the occurrence of $\overline{WR}$ and $\overline{DACK}$

Note:

1. $\overline{EOP}$, $\overline{WR}$, and $\overline{DACK}$ must be concurrently True for at least T7 for proper recognition of the $\overline{EOP}$ pulse.



12582C-084A

Figure 50. DMA Write (Non-Block Mode) Target Send Cycle

SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)
DMA Write (Non-Block Mode) Initiator Send Cycle (see Figure 51)

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	DRQ False from $\overline{DACK}$ True		100	ns
T2	$\overline{DACK}$ False to DRQ True	30		ns
T3	Write Enable Width*	70		ns
T4	$\overline{DACK}$ Hold from End of $\overline{WR}$	0		ns
T5	Data Setup to End of Write Enable*	30		ns
T6	Data Hold Time from End of $\overline{WR}$	40		ns
T7	Width of $\overline{EOP}$ Pulse (Note 1)	70		ns
T8	$\overline{REQ}$ True to $\overline{ACK}$ True		110	ns
T9	$\overline{REQ}$ False to DRQ True		110	ns
T10	$\overline{DACK}$ False to $\overline{ACK}$ False		130	ns
T11	$\overline{WR}$ False to Valid SCSI Data		100	ns
T12	Data Hold from Write Enable	*0		ns

*Write Enable is the occurrence of $\overline{WR}$ and $\overline{DACK}$

Note:

1. $\overline{EOP}$, $\overline{WR}$, and $\overline{DACK}$ must be concurrently True for at least T7 for proper recognition of the $\overline{EOP}$ pulse.

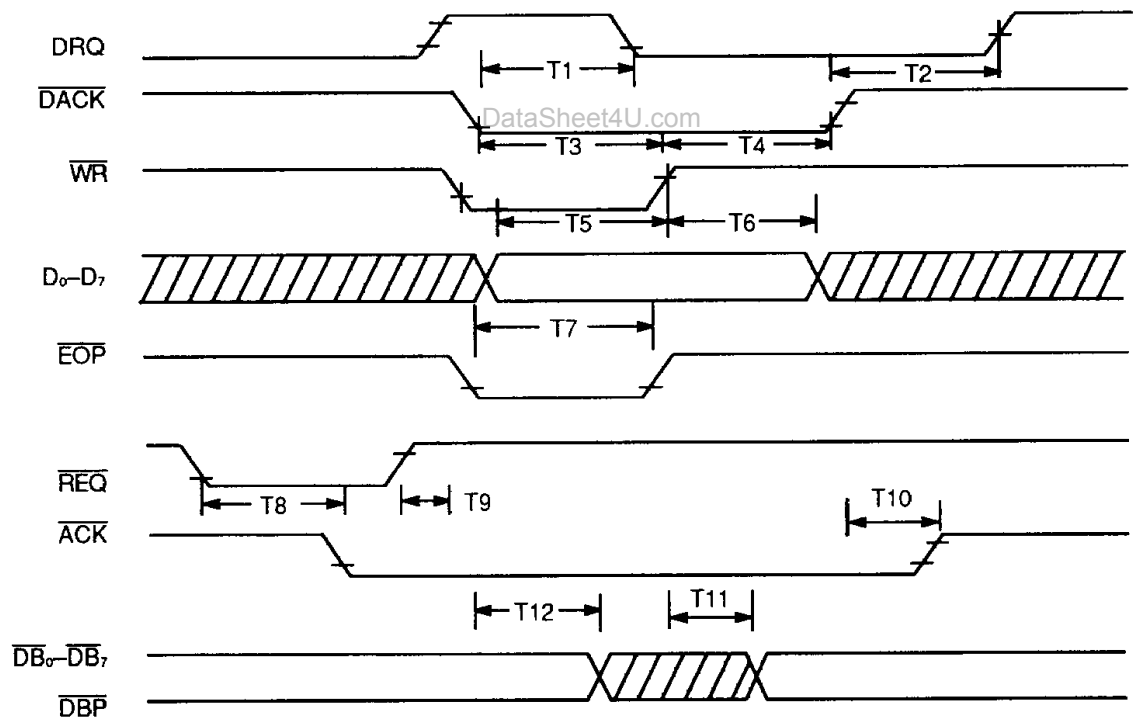


Figure 51. DMA Write (Non-Block Mode) Initiator Send Cycle

SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)
DMA Read (Non-Block Mode) Target Receive Cycle (see Figure 52)

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	DRQ False from $\overline{DACK}$ True		100	ns
T2	$\overline{DACK}$ False to DRQ True	30		ns
T3	$\overline{DACK}$ Hold Time from End of $\overline{RD}$	0		ns
T4	Data Access Time from Read Enable*		100	ns
T5	Data Hold Time from End of $\overline{RD}$	0		ns
T6	Width of $\overline{EOP}$ Pulse (Note 1)	70		ns
T7	$\overline{ACK}$ True to DRQ True		110	ns
T8	$\overline{DACK}$ False to $\overline{REQ}$ True ($\overline{ACK}$ False)		120	ns
T9	$\overline{ACK}$ True to $\overline{REQ}$ False		125	ns
T10	$\overline{ACK}$ False to $\overline{REQ}$ True ($\overline{DACK}$ False)		120	ns
T11	Data Setup Time to $\overline{ACK}$	10		ns
T12	Data Hold Time from $\overline{ACK}$	65		ns

*Read Enable is the occurrence of $\overline{RD}$ and $\overline{DACK}$

Note:

1. $\overline{EOP}$, $\overline{RD}$, and $\overline{DACK}$ must be concurrently True for at least T6 for proper recognition of the $\overline{EOP}$ pulse.

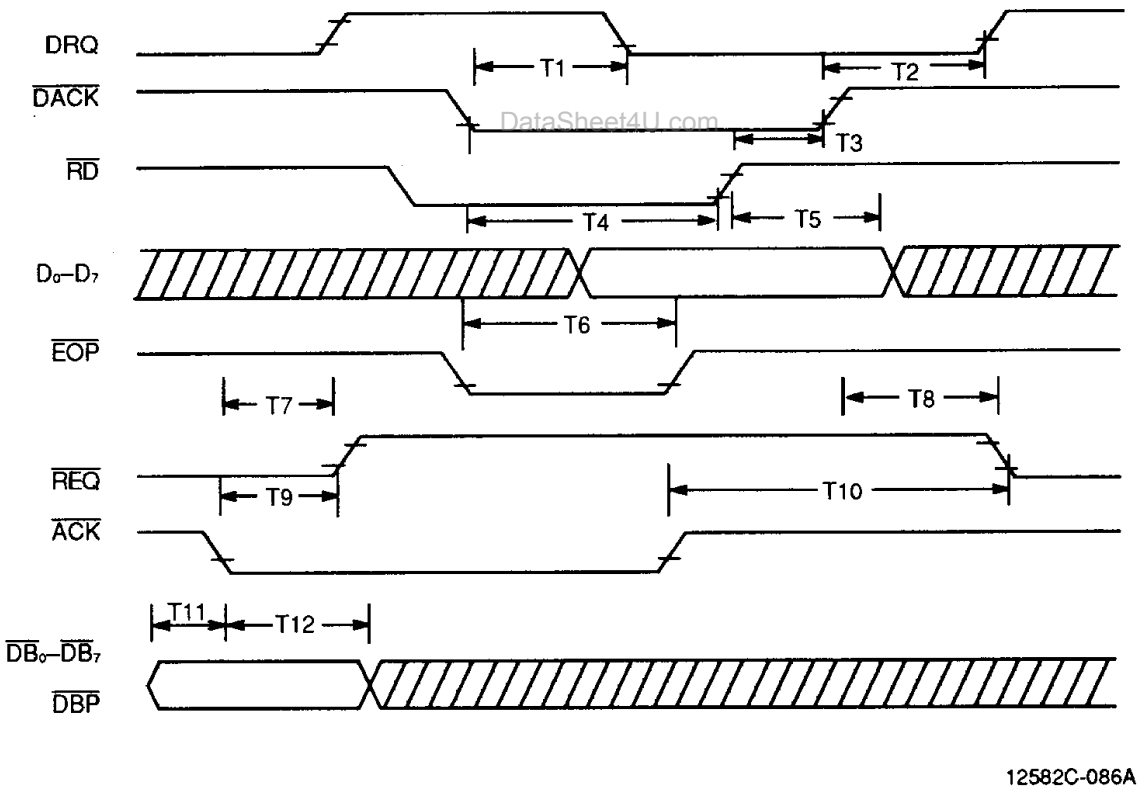


Figure 52. DMA Read (Non-Block Mode) Target Receive Cycle

SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)
DMA Read (Non-Block Mode) Initiator Receive Cycle (see Figure 53)

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	DRQ False from $\overline{DACK}$ True		100	ns
T2	$\overline{DACK}$ False to DRQ True	30		ns
T3	$\overline{DACK}$ Hold Time from End of $\overline{RD}$	0		ns
T4	Data Access Time from Read Enable*		100	ns
T5	Data Hold Time from End of $\overline{RD}$	0		ns
T6	Width of $\overline{EOP}$ Pulse (Note 1)	70		ns
T7	$\overline{REQ}$ True to DRQ True		140	ns
T8	$\overline{DACK}$ False to $\overline{ACK}$ False ($\overline{REQ}$ False)		100	ns
T9	$\overline{REQ}$ True to $\overline{ACK}$ True		110	ns
T10	$\overline{REQ}$ False to $\overline{ACK}$ False ($\overline{DACK}$ False)		100	ns
T11	Data Setup Time to $\overline{REQ}$	10		ns
T12	Data Hold Time from $\overline{REQ}$	65		ns

*Read Enable is the occurrence of $\overline{RD}$ and $\overline{DACK}$

Note:

1. $\overline{EOP}$, $\overline{RD}$, and $\overline{DACK}$ must be concurrently True for at least T6 for proper recognition of the $\overline{EOP}$ pulse.

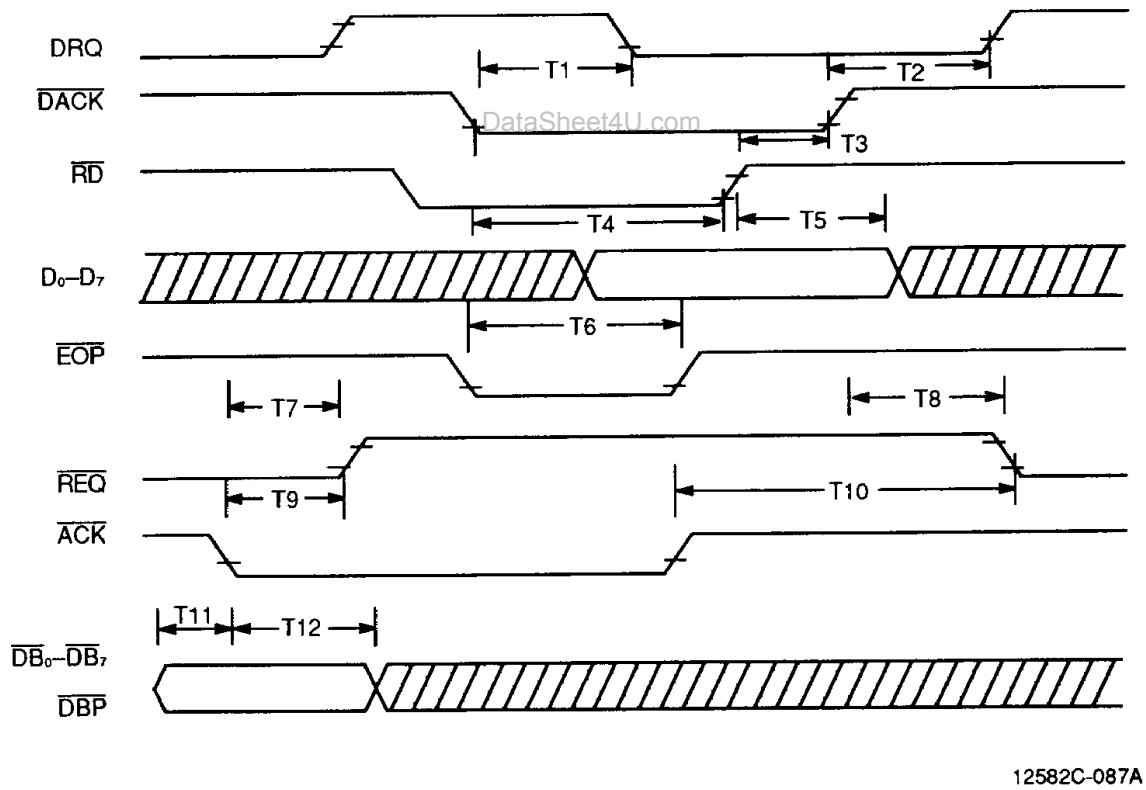


Figure 53. DMA Read (Non-Block Mode) Initiator Receive Cycle

SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)
DMA Write (Block Mode) Target Send Cycle (see Figure 54)

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	DRQ False from $\overline{DACK}$ True		100	ns
T2	Write Enable Width*	70		ns
T3	Write Recovery Time	120		ns
T4	Data Setup to End of Write Enable*	30		ns
T5	Data Hold Time from End of $\overline{WR}$	40		ns
T6	Width of $\overline{EOP}$ Pulse (Note 1)	70		ns
T7	$\overline{ACK}$ True to $\overline{REQ}$ False		125	ns
T8	$\overline{REQ}$ from End of $\overline{WR}$ ($\overline{ACK}$ False)		130	ns
T9	$\overline{REQ}$ from End of $\overline{ACK}$ ($\overline{WR}$ False)		110	ns
T13	Data Hold from $\overline{ACK}$ True	0		ns
T14	Data Setup to $\overline{REQ}$ True	60		ns

*Write Enable is the occurrence of $\overline{WR}$ and $\overline{DACK}$

Note:

1. $\overline{EOP}$, $\overline{WR}$, and $\overline{DACK}$ must be concurrently True for at least T6 for proper recognition of the $\overline{EOP}$ pulse.

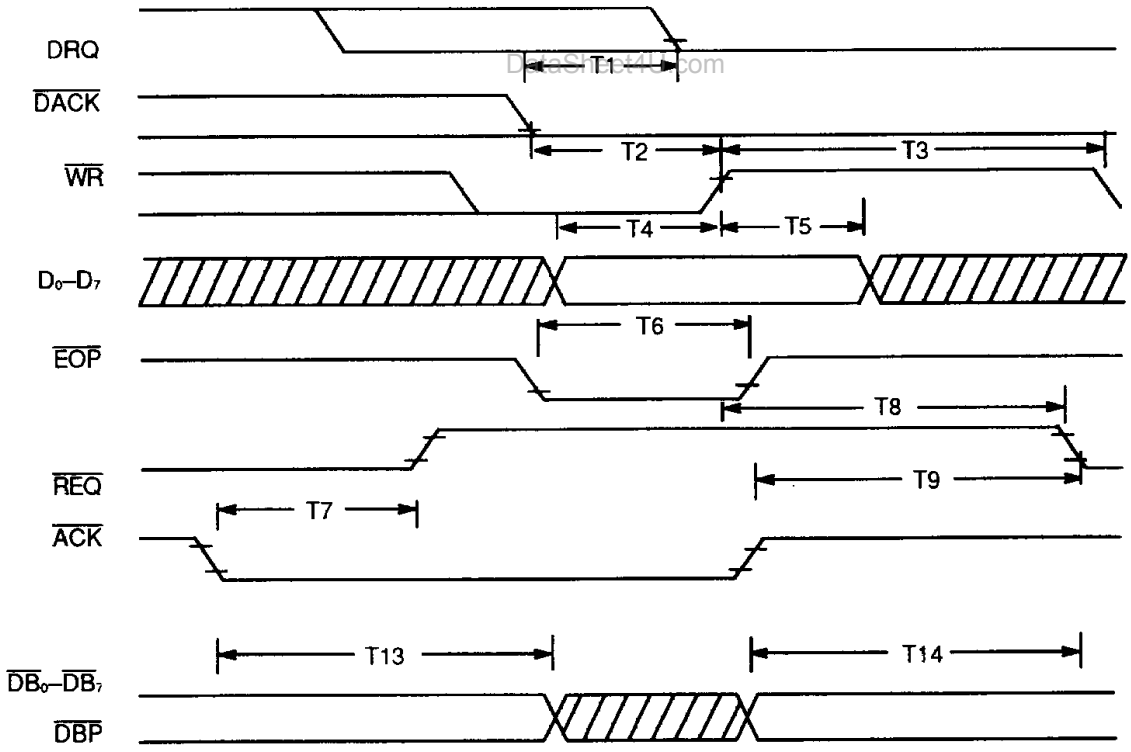


Figure 54. DMA Write (Block Mode) Target Send Cycle

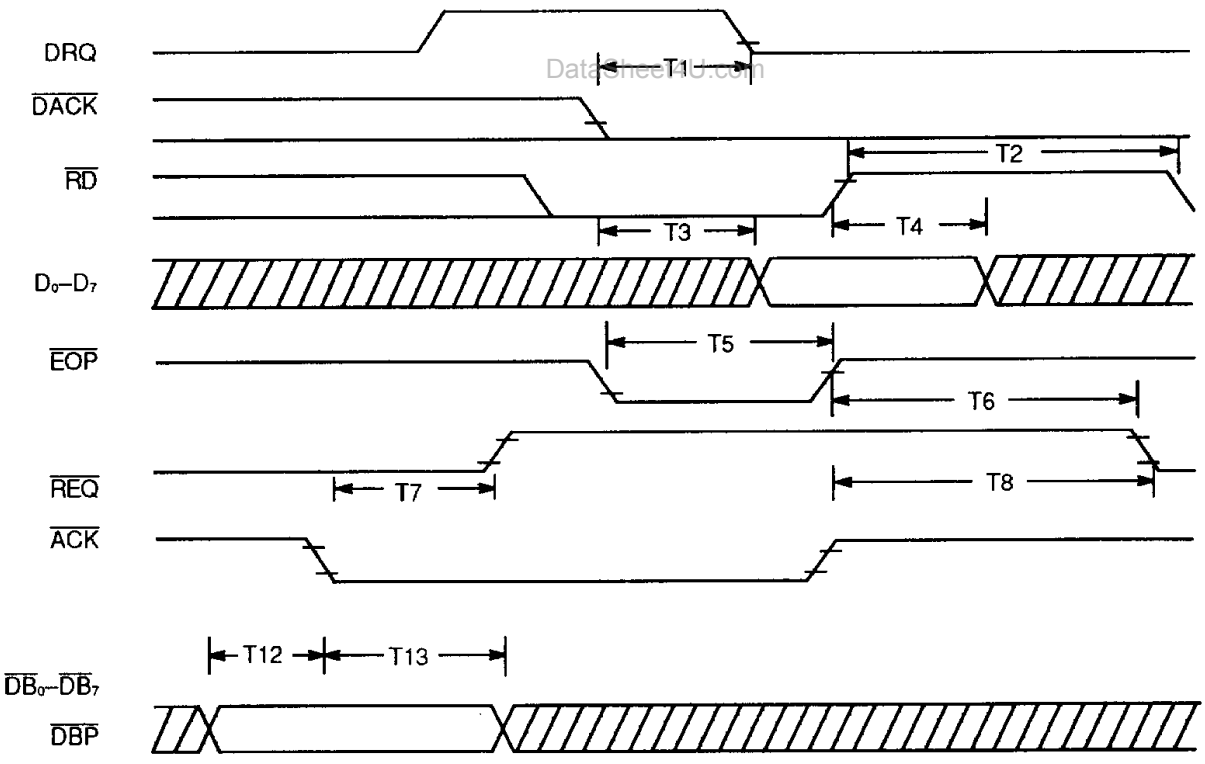
SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)
DMA Read (Block Mode) Target Receive Cycle (see Figure 55)

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	DRQ False from $\overline{DACK}$ True		100	ns
T2	$\overline{RD}$ Recovery Time	120		ns
T3	Data Access Time from Read Enable*		100	ns
T4	Data Hold Time from End of $\overline{RD}$	0		ns
T5	Width of $\overline{EOP}$ Pulse (Note 1)	70		ns
T6	$\overline{RD}$ False to $\overline{REQ}$ True (ACK False)		130	ns
T7	$\overline{ACK}$ True to $\overline{REQ}$ False		125	ns
T8	$\overline{ACK}$ False to $\overline{REQ}$ True ($\overline{RD}$ False)		110	ns
T12	Data Setup Time to $\overline{ACK}$	*10		ns
T13	Data Hold Time from $\overline{ACK}$	65		ns

*Read Enable is the occurrence of $\overline{RD}$ and $\overline{DACK}$

Note:

1. $\overline{EOP}$, $\overline{RD}$, and $\overline{DACK}$ must be concurrently True for at least T5 for proper recognition of the $\overline{EOP}$ pulse.



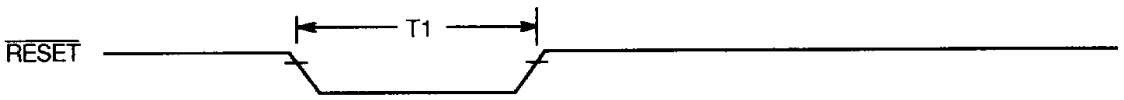
12582C-089A

Figure 55. DMA Read (Block Mode) Target Receive Cycle

SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)

Reset

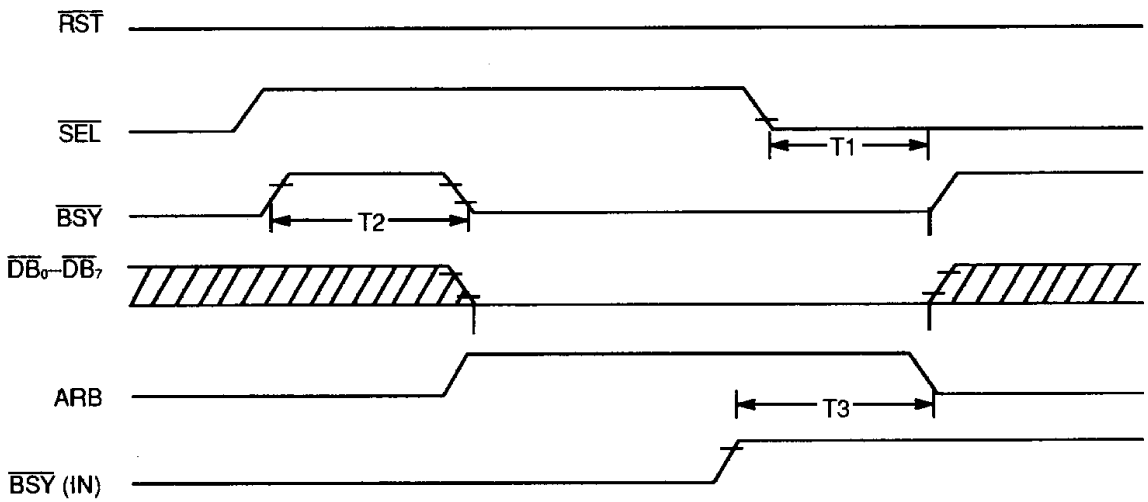
Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	Minimum Width of $\overline{RESET}$	100		ns



12582C-090A

Arbitration

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	Bus Clear from $\overline{SEL}$ True		600	ns
T2	Arbitrate Start from $\overline{BSY}$ False	1200	2400	ns
T3	Bus Clear from $\overline{BSY}$ False		1100	ns



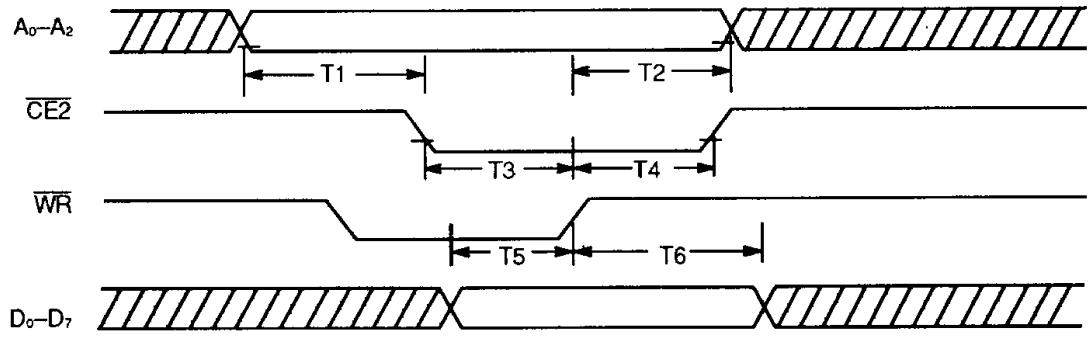
12582C-091A

SCSI SWITCHING CHARACTERISTICS/WAVEFORMS (Continued)

CPU Write Cycle

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	Address Setup to Write Enable*	10		ns
T2	Address Hold from End Write Enable*	0		ns
T3	Write Enable Width*	40		ns
T4	Chip Select Hold from End of $\overline{WR}$	0		ns
T5	Data Setup to End of Write Enable*	20		ns
T6	Data Hold Time from End of $\overline{WR}$	30		ns

*Write Enable is the occurrence of $\overline{WR}$ and $\overline{CE2}$.

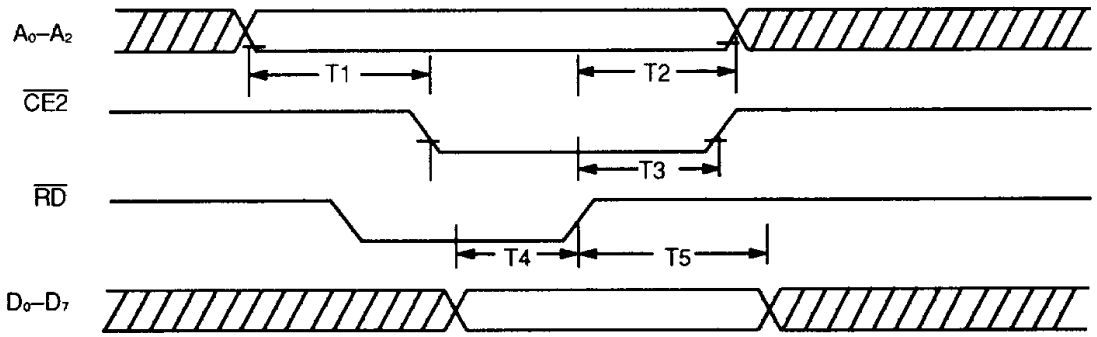


12582C-092A

CPU Read Cycle

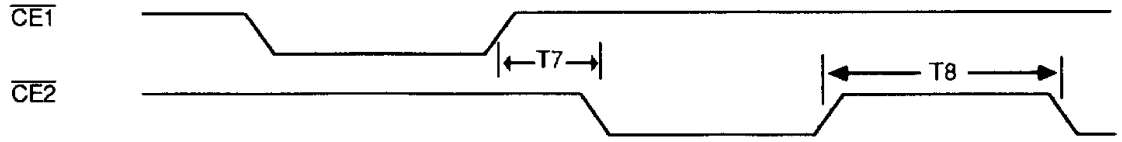
Parameter Symbol	Parameter Description	Min.	Max.	Unit
T1	Address Setup to Read Enable*	10		ns
T2	Address Hold from End Read Enable*	0		ns
T3	Chip Select Hold from End of $\overline{RD}$	0		ns
T4	Data Access Time from Read Enable*		100	ns
T5	Data Hold Time from End of $\overline{RD}$	0		ns

*Read Enable is the occurrence of $\overline{RD}$ and $\overline{CE2}$.



12582C-093A

Parameter Symbol	Parameter Description	Min.	Max.	Unit
T7	$\overline{CE1}$ to $\overline{CE2}$ Recovery	100		ns
T8	$\overline{CE2}$ to $\overline{CE2}$ Recovery	100		ns



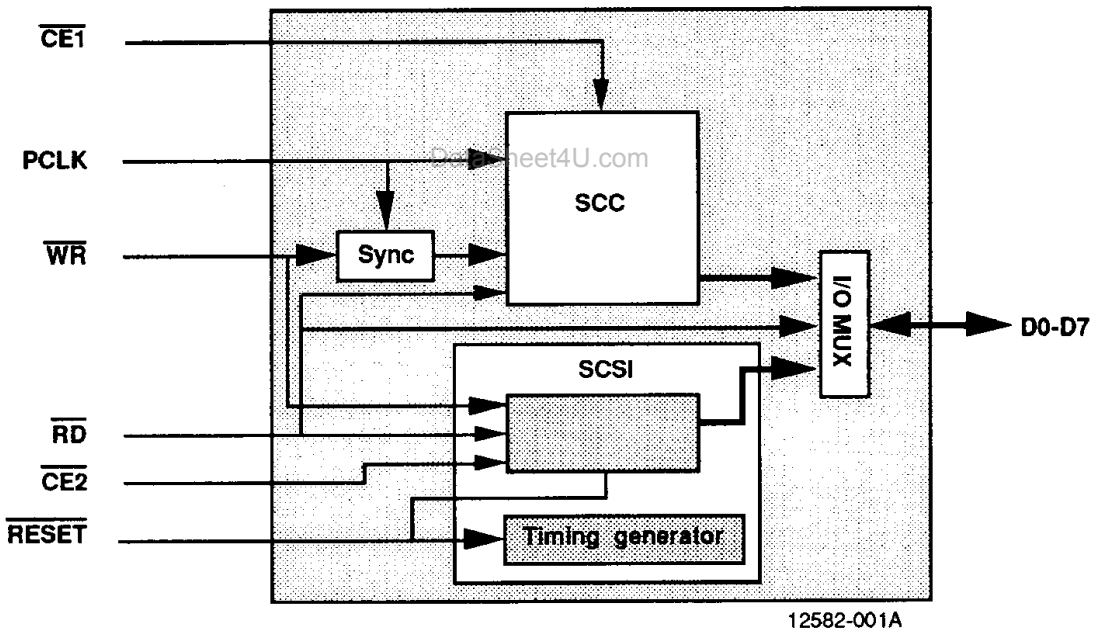
12582C-094A

SLEEP MODE

The Am85C80 has a "sleep mode" feature which is extremely desirable in designing low power systems such as laptop, notebook and tablet PCs. Basically, the "sleep mode" feature saves power consumption by shutting down all the active circuitries when they are not in use, while keeping the necessary register values. While in "sleep mode," any incoming clock has to be stopped, all internal Voltage Controlled Oscillators and Ring Oscillators should be disabled, and to avoid driving any resistive load, the outputs have to be in tri-state. System designers should take necessary precautions to ensure no glitches occur when the system clock is being stopped or started. This is a key design issue for systems with clock stop/start capability.

The Am85C80's sleep mode has been designed so that the two different portions of the device (SCSI and SCC) may be powered down separately or together, depending on user needs. For example, a SCSI disk can be shut down while serial communication is active, or the serial ports can be shut down while data is transferring to/from the disk. To utilize the least current (typical 80 µA), both the SCSI and SCC can be powered down, and this state is called "Deep Sleep." If a user exercises a shut down and wake up session into and out of "Deep Sleep," the Am85C80 will save all its SCC register values. On the other hand, its SCSI register values will be destroyed, and the user has to program the SCSI portion once the device is out of the "sleep mode."

From the design point of view, the clock and host interface of the Am85C80 are crucial. The SCSI timing generator has an internal oscillator that is used for Arbitration and Selection timing defined in the ANSI X3.131-1986 standard. Because $\overline{WR}$ is synchronized with PCLK, the SCC writing registers are blocked when PCLK is stopped. However, any read and write to SCSI is unaffected by PCLK since it is still asynchronous. This feature enables SCSI and SCC portions of Am85C80 to be in "sleep mode" separately or together. Typically, only the SCSI or SCC portion of a Am85C80 will be active. The diagram below is a simplified block diagram of the Am85C80 drawn to address the power down mode.

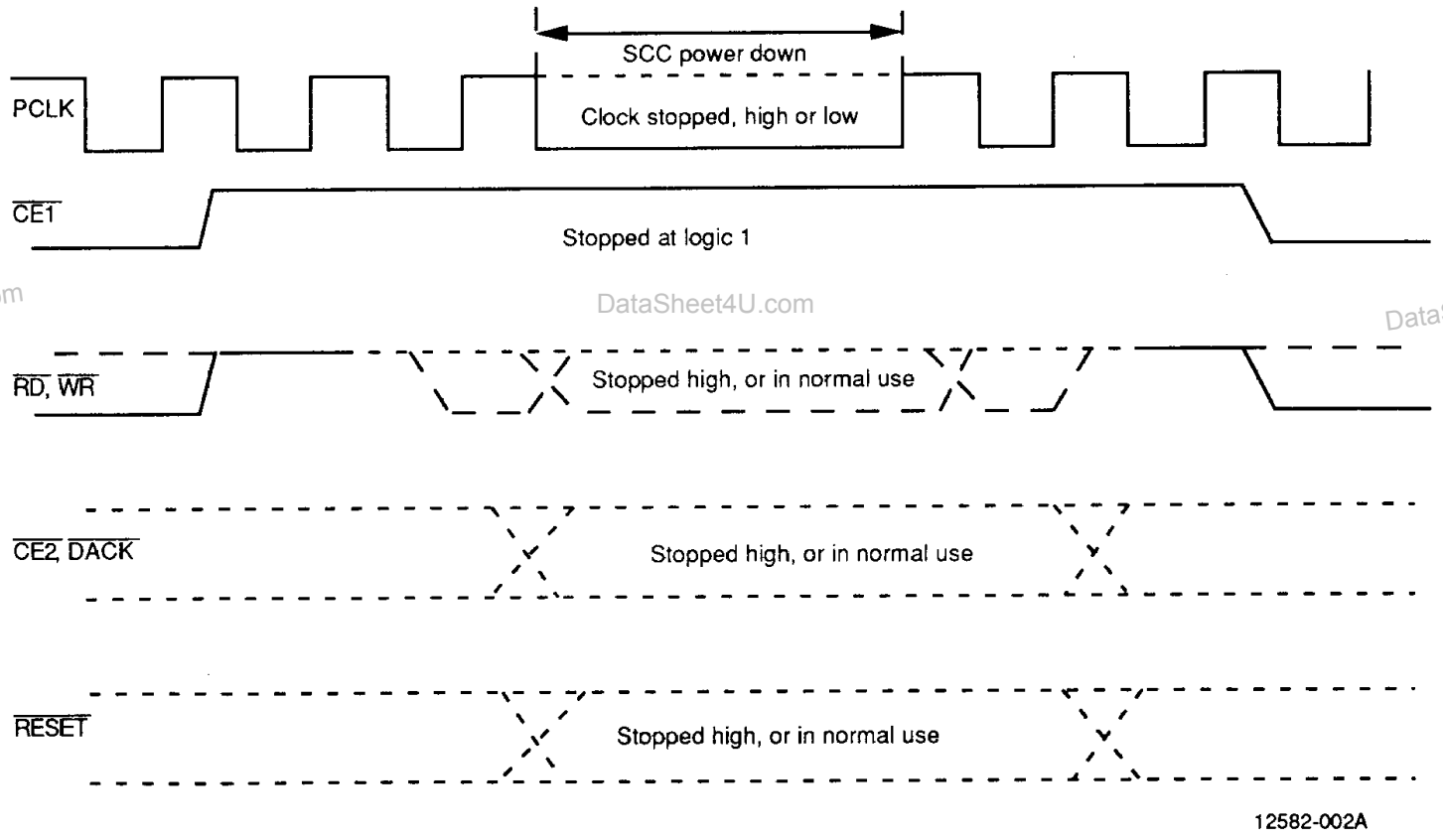


Simplified Block Diagram of the Am85C80

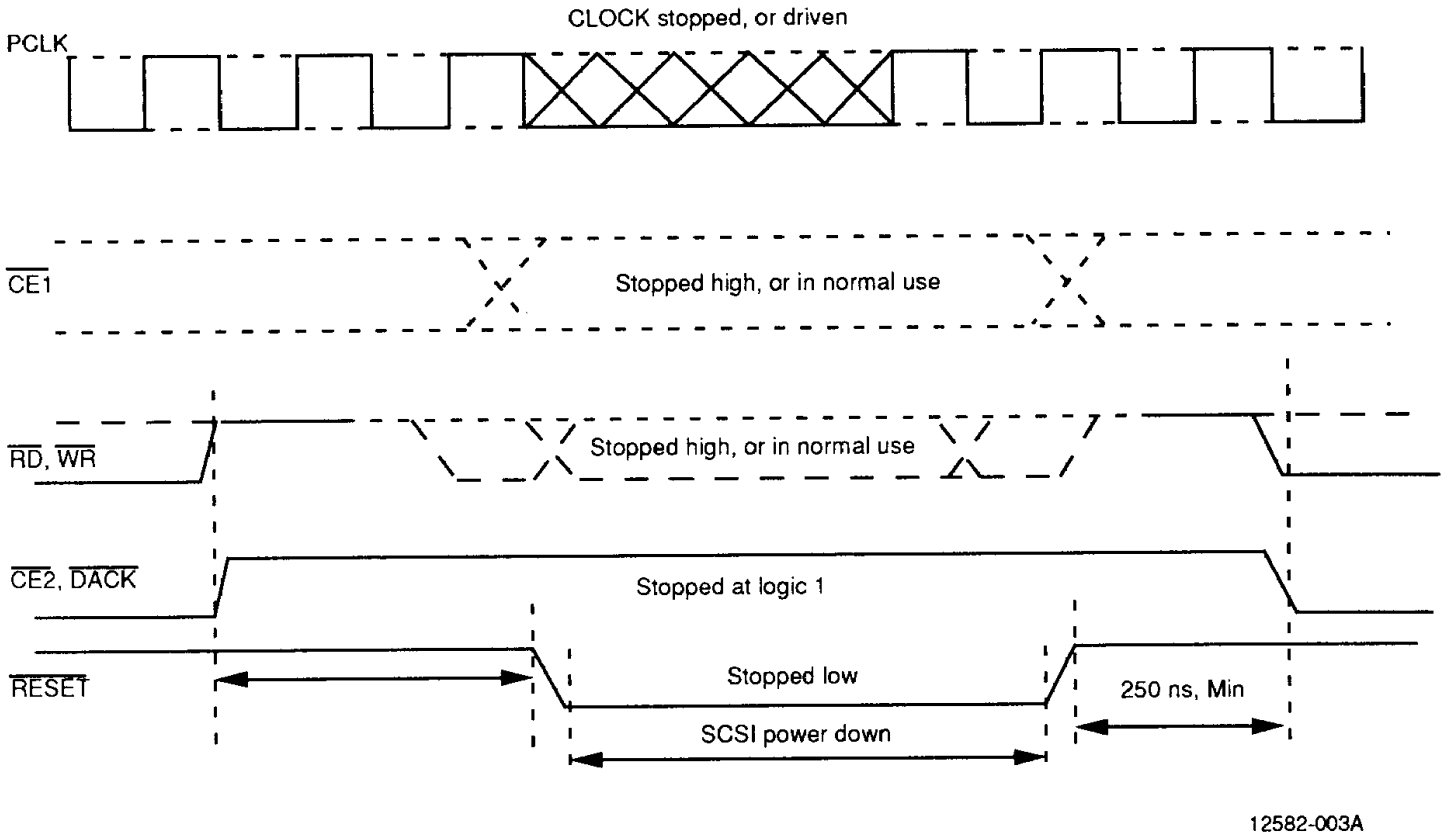
Guidelines for Designing Battery Powered Computers Using the Am85C80:

1. To save power, power-on reset (POR) circuitry is not implemented in the Am85C80. A user needs to do a soft reset, or assert both $\overline{RD}$ and $\overline{WR}$ simultaneously to cause a hard reset to the SCC portion.
2. In order to ensure the outputs (data lines) are inactive, either:
   a) $\overline{RD}$ should be kept at logic 1, or
   b) $\overline{CE1}$ and $\overline{CE2}$ should be kept at logic 1.
3. In order to put the SCC portion into "sleep mode," PCLK should be stopped at a logic 1 or 0 level. Users should ensure PCLK is stopped and started cleanly, with no double clocking or glitches. PCLK should be stopped a minimum of two cycles of PCLK before and after the access to the SCC.
4. In order to put the SCSI portion into "sleep mode," $\overline{RESET}$ should be asserted logic 0 to stop internal timing generator as shown on the diagram. $\overline{RESET}$ signal, which affects only the SCSI portion, can be asserted any time even in the middle of a CPU read or write. (Attention should be given to the software to first terminate any SCSI bus activity by going into bus free phase to avoid troubles on the SCSI bus when $\overline{RESET}$ is asserted.)
5. In order to put the Am85C80 into "Deep Sleep" mode, the user should put both the SCC and SCSI portion into their respective "sleep modes." This is done by both stopping PCLK and asserting $\overline{RESET}$.

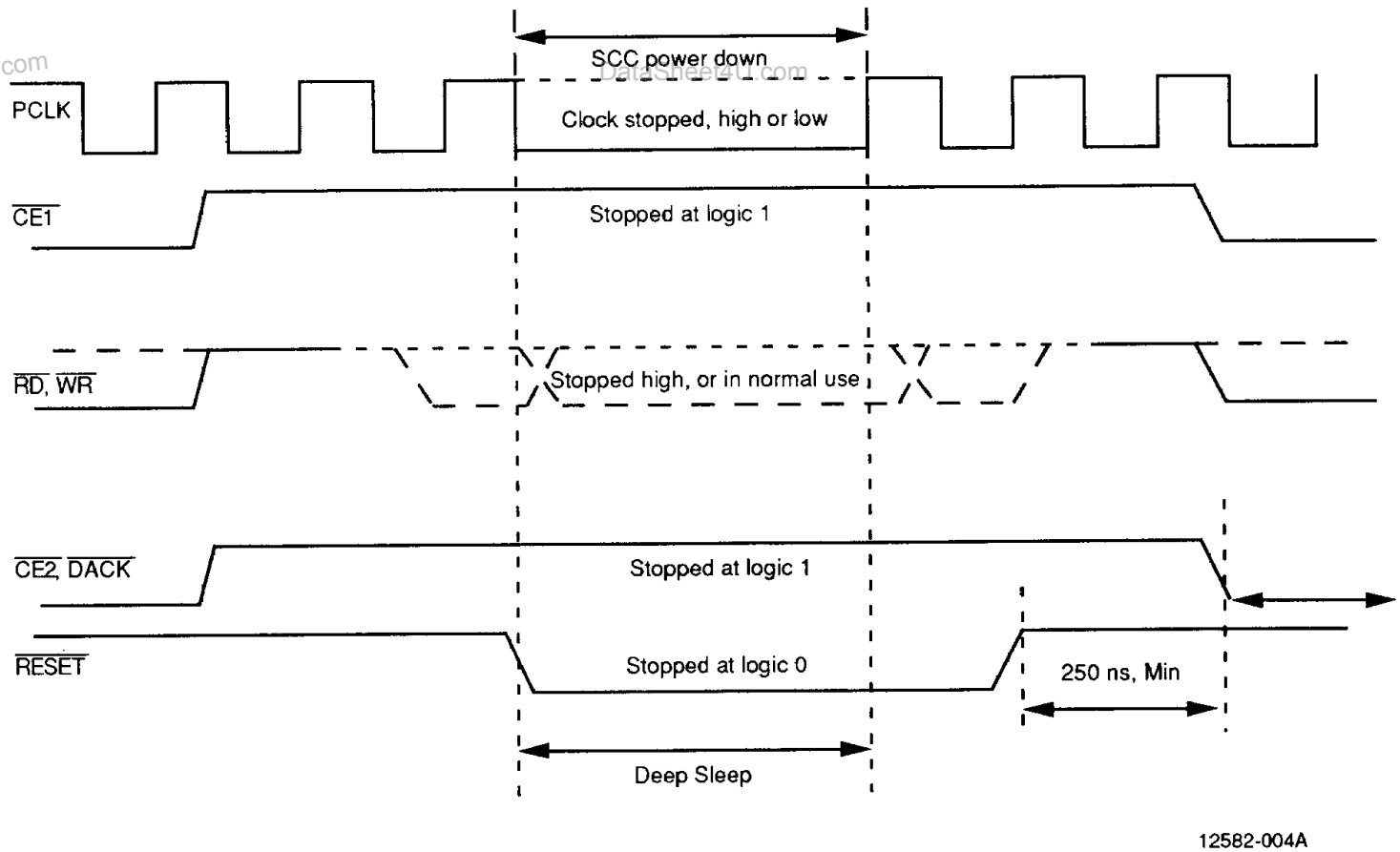
The above guide lines are for designing the laptop computers using Am85C80. The following is a typical timing diagram the one could use. It is recommended that while the chip is in "Deep Sleep" mode, $\overline{CE1}$, $\overline{CE2}$, $\overline{RD}$, and $\overline{WR}$ are at logic 1.



SCC Power Down Mode (SCC PCLK Stopped)



SCSI Power Down (RESET Held Low)



Power Down Mode: "Deep Sleep"