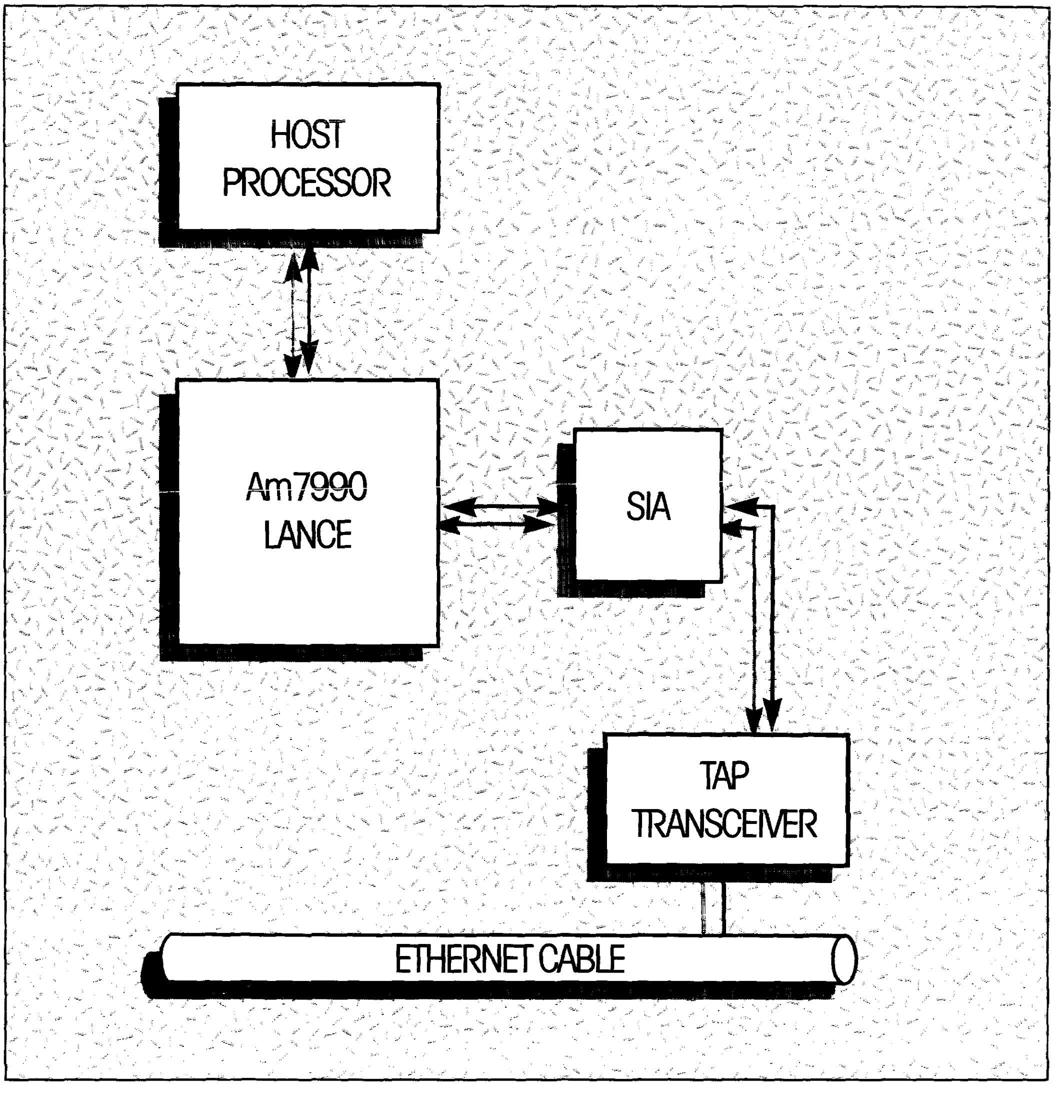
Am7990
Local Area Network Controller (LANCE)

Technical Manual

Advanced
Micro
Devices



Advanced Micro Devices

Local Area Network Controller Am7990 (LANCE)

by Rasoul M. Oskouy, Senior Applications Engineer
Erland Kyllonen, Technical Writer
Harry Lau, Copyeditor
AMD Headquarters Applications

TABLE OF CONTENTS

TABLES

FIGURES

CHAPTER 1
OVERVIEW

The LANCE (Local Area Network Controller for Ethernet) is a 48-pin VLSI device designed to greatly simplify interfacing a microcomputer or minicomputer to an IEEE 802.3/ Ethernet/Cheapernet Local Area Network. In addition to transferring data packets to and from the Ethernet transceiver, the LANCE has the ability to manage the data buffers in memory. Figure 1-1 is a block diagram of the LANCE.

This technical manual discusses the hardware and software considerations that may be useful for the user designing systems around the Am7990 LANCE. It examines the LANCE interfaces with typical microprocessors to show the flexibility under different environments. Bus bandwidth, bus latency, and system performance are discussed for different architectures.

The flowcharts of software drivers, along with some guidelines, are introduced for a better understanding of how the part may be programmed and used in typical applications. No attempt is made to explore software issues relating to the upper layer protocols of the ISO model.

As a communication controller, the LANCE manages descriptor rings and data buffers in memory for both transmitting and receiving data. It shares DMA access to this memory with other master bus devices such as CPUs.

In the transmitting mode, the LANCE transfers data from the current buffer to a 48-byte (FIFO) register in the LANCE called a Silo. The output of the Silo is serialized and then goes to a Serial Interface Adapter (SIA), the Am7992B. Design considerations for the SIA are not discussed in this manual. The output of the SIA is connected to the Ethernet lines through a transceiver. A pin-for-pin compatible IEEE 802.3/ Ethernet/Cheapernet transceiver, Am7995, is available from AMD.

In the receiving mode, the SIA receives data from the Ethernet cable. It transfers the data to the Silo in the LANCE. The LANCE transfers the data from the Silo to the current buffer in local memory. Figure 1-2 is a simplified diagram of the data flow.

To initialize the LANCE, it must be in the Bus Slave mode. When $\overline{CS}$ is active, the LANCE is in the Bus Slave mode.

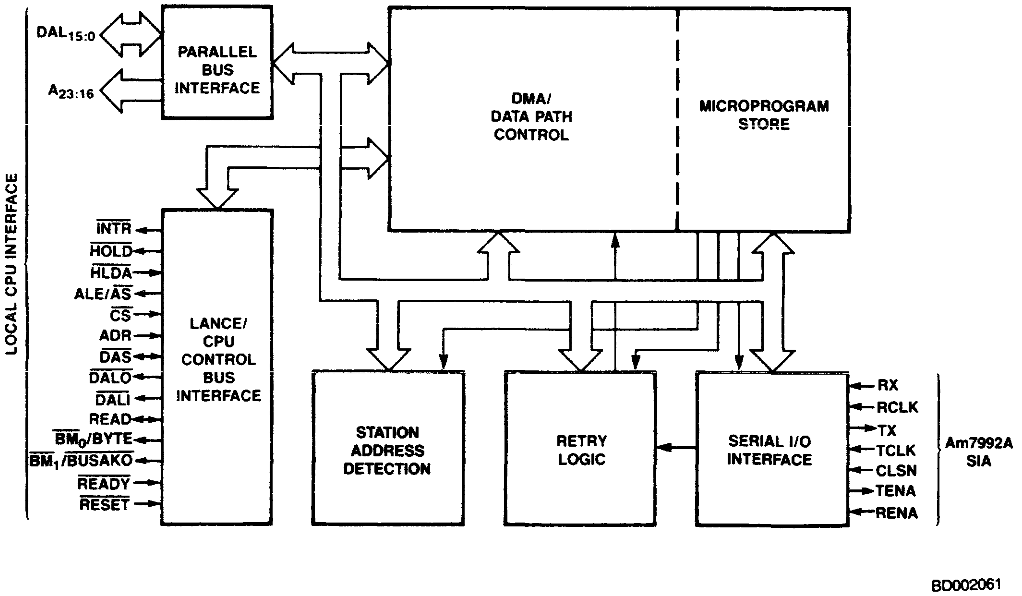


Figure 1-1 LANCE Block Diagram

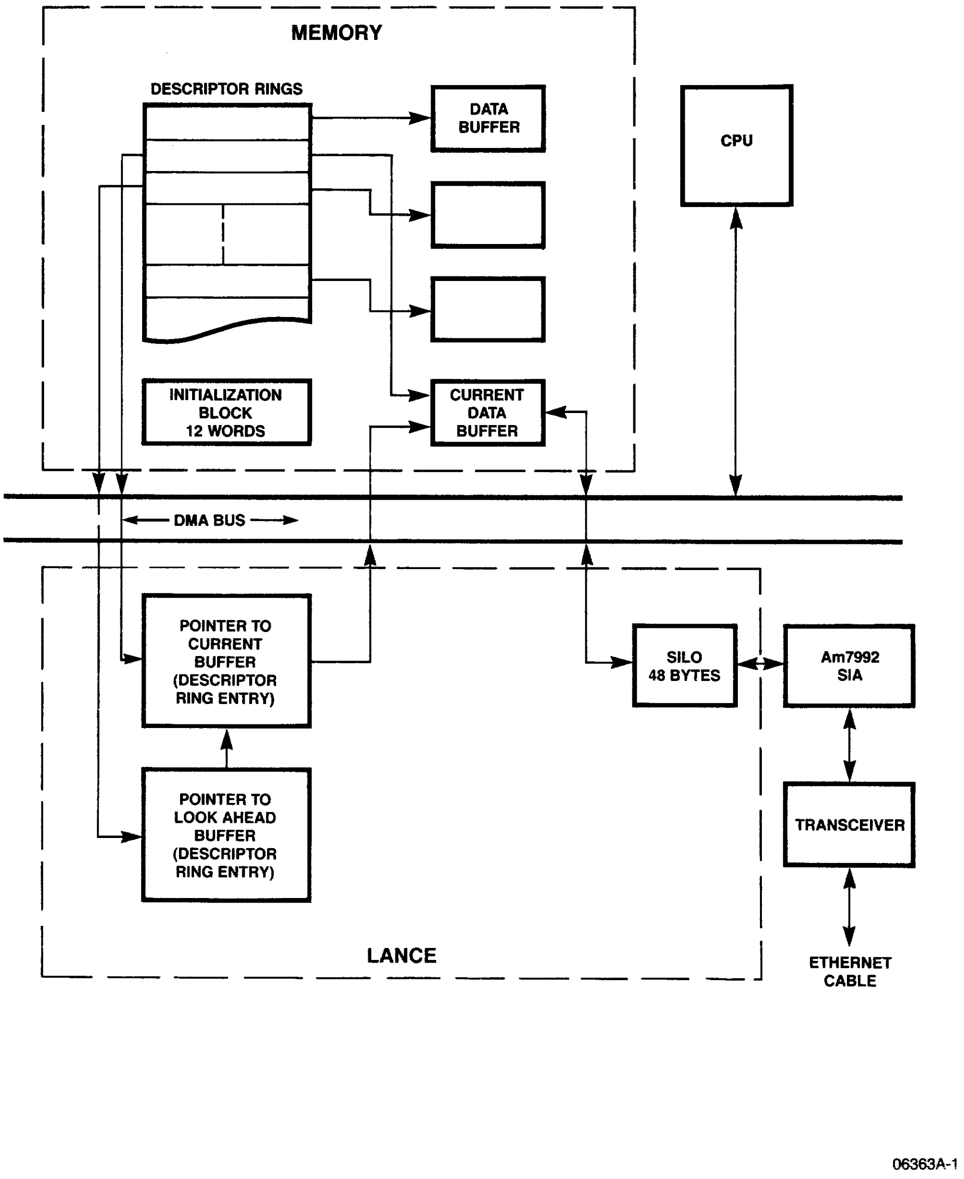

MEMORY
DESCRIPTOR RINGS
DATA BUFFER
CPU
INITIALIZATION BLOCK 12 WORDS
CURRENT DATA BUFFER
DMA BUS
POINTER TO CURRENT BUFFER (DESCRIPTOR RING ENTRY)
SILO 48 BYTES
Am7992 SIA
TRANSCEIVER
POINTER TO LOOK AHEAD BUFFER (DESCRIPTOR RING ENTRY)
LANCE
ETHERNET CABLE
06363A-1

Figure 1-2 Data Flow

The LANCE is initialized by entering configuration parameters into the four Control and Status Registers (CSR) and into an Initialization Block of 12 words. CSR_0 contains interrupt and error status flags and control bits. CSR_1 and CSR_2 contain the address of the Initialization Block. CSR_3 contains control bits to configure Byte Swap (BSWP), ALE Control (ACON), and Byte Control (BCON). The Initialization Block contains pointers to the descriptor rings and their length.

Buffer management is accomplished through message descriptors organized in ring structures in memory. Each message descriptor entry is four words long. There are separate descriptor rings for transmitting and receiving. Packets to be transmitted are loaded into buffers by the CPU and the addresses and lengths of the buffers are stored in consecutive entries in the Transmit Descriptor ring. Then, the ownership bit is set in each entry indicating that the LANCE owns the buffers. The LANCE polls the first descriptor entry and when it finds that it owns it, it proceeds to transmit the packet.

The LANCE maintains a copy of the descriptor entry that points to the current buffer. If the current buffer entry is not the final buffer for the packet being sent, the LANCE maintains a copy of the descriptor ring entry for the next buffer to be transmitted in the look-a-head buffer.

When a Receive Descriptor Ring entry is empty, the ownership bit for it is set meaning that the buffer it points to is available to the LANCE to store an incoming packet. After the LANCE has stored a packet in the buffer or filled the buffer, it resets the ownership bit thus giving control of the buffer to the host. It then polls the next descriptor entry to see if the next buffer is available for additional incoming packets.

The LANCE also enters status information into the descriptor entries.

It is recommended that the reader refer to the LANCE Data Sheet for further detail such as timing.

1.1 EARLY DESIGN STAGE TESTING

As with any other peripherals, a hierarchical approach should be taken to integrate the LANCE into a system for proper operation. The following outline shows the order in which the various LANCE functions should be developed and tested.

— RESET
— BUS SLAVE OPERATION
— INITIALIZATION
— BUS MASTER OPERATION
— DIAGNOSTICS
— NORMAL OPERATION

The initialization, Bus Master operation, and the diagnostic steps are closely tied together, such that each step requires the proper operation or arrangement of the other steps. A brief discussion of significant points for each step follows. A detailed description appears later in this chapter.

1.2 RESET

There are two ways to reset the LANCE: Hard Reset ($\overline{\text{RESET}}$ input to the LANCE), and Soft Reset (by setting the stop bit in CSR_0). Reset causes the LANCE to cease operation and clear its internal logic.

Control Status Registers (CSR_0 CSR_3) contents are affected as follows:

— CSR_0 is cleared. (Stop bit gets set.)
— CSR_3 is cleared.
— CSR_1, CSR_2 undefined by reset.

The LANCE operation may be resumed by setting the STRT bit in CSR_0 after it is stopped. However, the above condition implies that when the LANCE is stopped (Soft Reset), CSR_3 must be reprogrammed if the LANCE interface had been configured for non-default values. If CSR_3 has a non-zero value, it must be reloaded each time that the stop bit is set.

1.3 BUS SLAVE OPERATION

The LANCE enters the Bus Slave mode whenever $\overline{\text{CS}}$ becomes active. This mode is used for accessing the four Control Status Registers (CSR_0, CSR_1, CSR_2, CSR_3), and the Register Address Pointer (RAP). It takes the CPU normally two separate cycles to access CSR_0-CSR_3. The first cycle selects the CSR by writing to RAP, and the second cycle accesses the CSR pointed to by RAP. RAP is a latch and is not changed until it is rewritten; therefore, accessing the same CSR does not require the register select cycle.

During Bus Master Mode operation, the LANCE uses the parameters contained in CSR_1-CSR_3. Therefore, LANCE operation must be stopped by setting the Stop bit in CSR_0 whenever CSR_1-CSR_3 accesses are required. The LANCE and CPU interface can be tested by writing and then reading the Control Status Registers (CSR_0-CSR_3).

The CPU communicates with the LANCE in a handshake sequence by asserting the $\overline{\text{DAS}}$ signal as a request for Data Transfer, and receiving the READY output from the LANCE in return as an acknowledge, (refer to the LANCE Data Sheet for Bus Slave Timing and Read/Write operation). Note that there are two different types of delays involved in accessing the CSRs. The shorter delays refer to accesses to CSR_0, CSR_3, and RAP. The longer delays refer to accesses to CSR_1 and CSR_2. Proper operation of the LANCE in Bus Slave Mode is a prerequisite for the LANCE to operate in Bus Master Mode.

1.4 INITIALIZATION PROCEDURE

The LANCE is initialized (configured) for a desired operation by programming the CSR registers (CSR_1-CSR_3) and providing the parameters required by the LANCE in the Initialization Block. Proper initialization is a prerequisite for the LANCE operation in Bus Master mode. Refer to Figure 1-3 and Figure 1-4.

1.4.1 CSR PROGRAMMING

Depending upon which processor is to be interfaced to the LANCE, ACON, BCON, and BSWP in the Control Status Register 3 (CSR_3) must be programmed when initializing the LANCE.

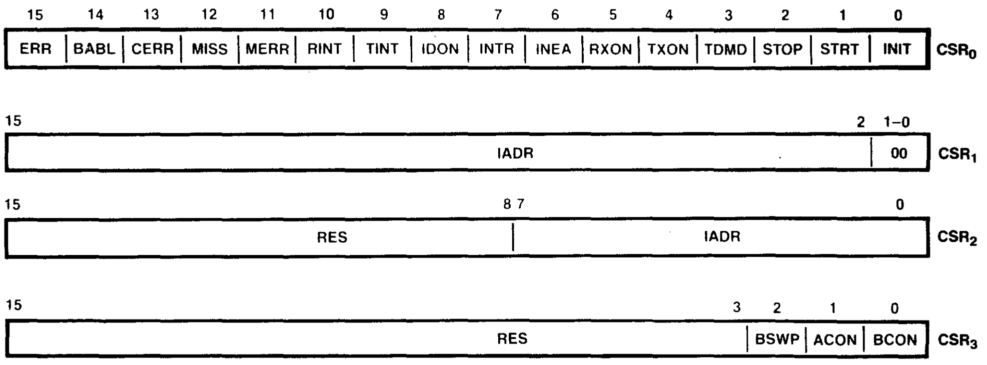
ACON (ALE control) defines the polarity of ALE signal when the LANCE is in Bus Master Mode. Multiplexed buses use this signal (ALE) to latch the least significant part of the Address Bus (A_{15}:A_0).

BCON (byte control) redefines the byte mask I/O pins (pins 15 and 16) for different ways to handle odd address boundaries and redefines $\overline{\text{HOLD}}$ I/O pin 17 to configure the LANCE as a daisy-chain system.

When BCON = 0, pins 15, 16, and 17 are defined as $\overline{\text{BM}}_0$, $\overline{\text{BM}}_1$, and HOLD. Pins 15 and 16 specify byte selection on the DAL lines as whole word, upper byte, lower byte, or none. Pin 17, $\overline{\text{HOLD}}$, is asserted by LANCE whenever it requires access to memory.

When BCON = 1, these pins are defined as BYTE, $\overline{\text{BUSAKO}}$, and $\overline{\text{BUSRQ}}$. The BYTE line (pin 15) is used in conjunction with the least significant bit of the DMA address to specify byte selection as whole word, lower output. If the LANCE is not requesting the bus and it receives $\overline{\text{HDLA}}$ $\overline{\text{BUSAKO}}$ is driven low. If the LANCE is requesting the bus when it receives $\overline{\text{HDLA}}$ $\overline{\text{BUSAKO}}$ will remain high. $\overline{\text{BUSRQ}}$ pin 17, will only be asserted if pin 17 is high prior to assertion when in the daisy chain configuration.

The BSWP bit in CSR_3 determines if the LANCE treats the high byte as the most significant or least significant byte when writing into the Silo (Bus Master Mode, read operation) or when reading it from the Silo (Bus Master Mode, write operation). This function facilitates operation with different 16-bit microprocessors. When BSWP = 1, the LANCE will swap the high and low bytes on DMA transfers between the Silo and and bus memory.



06363A-2

Figure 1-3 Control and Status Registers in LANCE

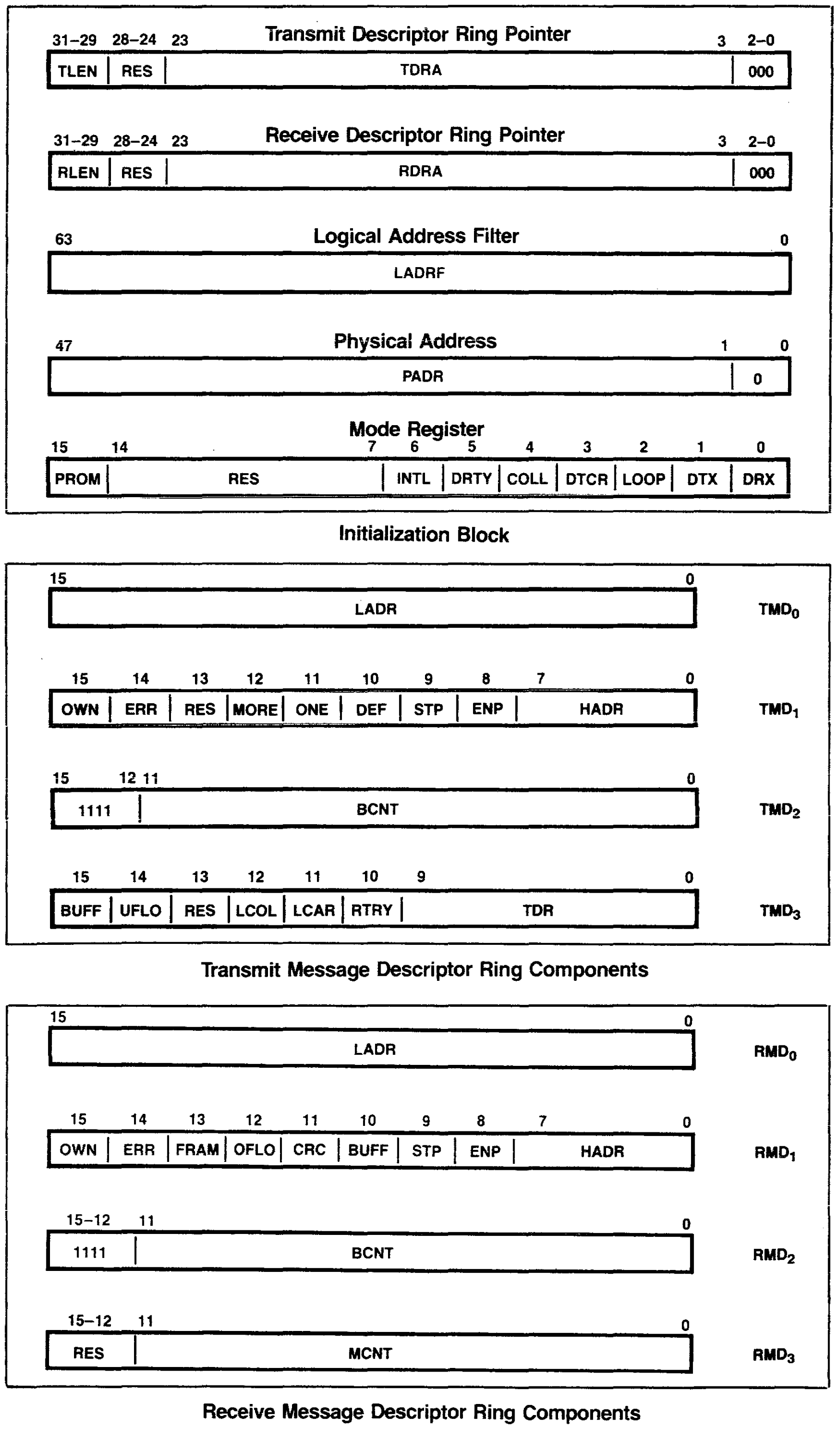


Figure 1-4 Initialization Block and Descriptor Rings

Table 1-1. Processor Interface Requirements

	Z8000	8086	68000	LSI-11
Dedicated buses	**Z-bus (multiplexed)**	**Multibus (multiplexed)**	**Versabus (demultiplexed)**	**Q-bus (multiplexed)**
Address latch enable polarity	$\overline{\text{AS}}$	ALE*	ALE	ALE
Byte addressing of memory	Byte/$\overline{\text{Word}}$ signal	Upper Byte Strobe	Upper and lower strobes	Byte/$\overline{\text{Word}}$ signal
Memory organization	Even address accesses upper byte; odd address accesses lower byte	Even address accesses lower byte; odd address accesses upper byte	Even address accesses upper byte; odd address accesses lower byte	Even address accesses lower byte; odd address accesses upper byte

* The Am7990 Tri-states ALE; the 8086 does not.

Table 1-1 shows how to program CSR_3 when interfaced to different processors. CSR_1 and CSR_2 are programmed to contain a pointer to the Initialization Block.

1.4.2 INITIALIZATION BLOCK

This block contains the operating parameters necessary for device operation. The Initialization Block is 12 words long in contiguous memory starting on a word boundary (See Figure 1-3). The Initialization Block contains:

- Mode of operation (Mode Register)
- Physical Address
- Logical Address Mask
- Location of Transmit and Receive Descriptor Rings
- Number of entries in Transmit and Receive Descriptor Rings.

1.4.3 DIAGNOSTICS

The Mode Register in the Initialization Block is used to set up the LANCE for diagnostic or normal operation. There are four user-programmable diagnostic modes. Each mode requires the user to change bits in the Mode Register and reinitialize the LANCE. The four modes are listed as follows:

1. Internal Loopback
2. CRC Logic Check
3. Collision Detection and Retry Logic
4. External Loopback.

The detailed description of the diagnostic functions and flowcharts appear in a later section of this technical manual.

1.4.4 NORMAL OPERATION

Once the LANCE interface with the system is checked out both in Bus Master and Bus Slave Mode, and the internal logic of the LANCE has been tested in diagnostic mode for both internal and external loopback test, the LANCE can be configured for normal operation via the Initialization Block. The user normally initializes the LANCE once for normal operation. Any change in the LANCE configuration requires the re-initialization of the LANCE. The user must make sure to process all the transmitted or received packets and rearrange the packets queued for transmission in the Transmit Ring before re-initializing the LANCE. This is necessary since, upon re-initialization, the LANCE sets its pointers to the beginning of the Transmit and Receive Rings.

1.5 BUS MASTER OPERATION

The LANCE must be in Bus Master mode to access the external memory. Local memory external to the LANCE is used for the Initialization Block, transmit/receive rings, and transmit/receive buffers. $\overline{\text{HOLD}}$ and $\overline{\text{HLDA}}$ of the LANCE both active indicate that the LANCE is a Bus Master. The $\overline{\text{HLDA}}$ input to the LANCE may be directly connected to the HOLD output from the LANCE. This is desirable in some applications.

To guarantee that the LANCE is not switched into Slave Mode inadvertently, $\overline{\text{CS}}$ must not be asserted when the LANCE in Bus Master Mode ($\overline{\text{HLDA}}$ active). This is usually guaranteed by design when the LANCE and CPU share the same bus. However, in a dual port RAM design in which the LANCE is not on the same bus with the CPU, some external logic is required to monitor and control $\overline{\text{HOLD}}$ and $\overline{\text{HLDA}}$ before a $\overline{\text{CS}}$ is asserted.

Each DMA cycle initiated by the LANCE consists of 6 T-states (600 ns) plus any additional wait states (Tw) which is required for slower memories. Wait states may be added between T4 and T5 of the LANCE DMA cycle. The $\overline{\text{READY}}$ input to the LANCE inserts wait states in 100 ns increments. To guarantee that no Wait State is added to the bus cycle, the $\overline{\text{READY}}$ signal input to the LANCE must be asserted within 80 ns (max), following the falling edge of the ALE signal. Memory chips with 600 ns (max) cycles time and an access time of 280 ns from ALE falling edge, or 355 ns from Address Valid, can be used without adding any Wait States to the LANCE DMA cycles.

It is possible for one or more wait states to be added to the LANCE DMA cycle even with fast memory chips if the timing specification of the LANCE is not met. It is recommended that the memory cycle sequencer be synchronized with the falling edge of DAS and that $\overline{\text{READY}}$ be asserted within 80 ns following the falling edge of ALE in order to assure that no wait states are added.

In the Bus Master Mode, the LANCE has two types of DMA transfers: burst or single DMA cycle. Each burst transfer uses eight DMA cycles and transfers eight 16-bit words unless fewer than eight words remain to be transferred. Separate single word DMA cycles are used to access the transmit/receive rings and the Initialization Block.

The LANCE is equipped with some additional features, including two transceiver control signals, $\overline{\text{DALI}}$ (DAL In) and $\overline{\text{DALO}}$ (DAL Out). When in Bus Master Mode, the LANCE asserts $\overline{\text{DALI}}$ and $\overline{\text{DALO}}$ to enable the data transceivers for Read and Write operations. $\overline{\text{DALI}}$ is asserted only during the data portion of a read transfer. $\overline{\text{DALO}}$ is asserted during the write operation to enable the transceiver for both address and data (multiplexed Address/Data), and also during the read operation to transfer the address (refer to the LANCE Data Sheet). It is important to note that, during the read operation, there is a delay from deassertion of $\overline{\text{DALO}}$ to assertion of $\overline{\text{DALI}}$ or vice versa. This is to prevent bus contention which would cause electromagnetic interference (EMI).

1.6 GENERAL DESCRIPTION

The Am7990 (LANCE) is designed to operate in an environment that includes close coupling with a local memory and a microprocessor (HOST). The Am7990 LANCE is programmed by a combination of registers and data structures resident within the chip and in memory. There are four Control and Status Registers (CSRs) within the chip which are programmed by the HOST device. Once enabled, the chip has the ability to access memory locations to acquire additional operating parameters.

The Am7990 has the ability to do independent buffer management as well as transfer data packets to and from the Ethernet. There are three memory structures accessed by the Chip:

1. Initialization Block—12 words in contiguous memory starting on a word boundary. It also contains the operating parameters necessary for device operation. The Initialization Block is comprised of:
   - Mode of Operation
   - Physical Address
   - Logical Address Mask
   - Location of Receive and Transmit Descriptor Rings
   - Number of Entries in Receive and Transmit Descriptor Rings
2. Receive and Transmit Descriptor Rings—Two ring structures, one each for incoming and outgoing packets. Each entry in the rings is 4 words long and each entry must start on a quadword boundary. The Descriptor Rings are comprised of:
   - The address of a data buffer.
   - The length of that data buffer.
   - Status information associated with the buffer.
3. Data Buffers—Contiguous portions of memory reserved for packet buffering. Data buffers may begin on arbitrary byte boundaries.

In general, the programming sequence of the chip may be summarized as:

1. Programming the chip's CSRs by a host device to locate an Initialization Block in memory. The byte control, byte addressing, and address latch enable modes are defined here also.
2. The chip loads itself with the information contained within the Initialization Block.
3. The chip accesses the descriptor rings for packet handling.

The parallel interface of the Local Area Network Controller for Ethernet (LANCE) has been designed to be "friendly" or easy to interface to a variety of popular 16-bit microprocessors. These microprocessors include the following: Z8000, 8086, 68000 and LSI-11. The LANCE has a 24-bit wide linear address space when it is in the Bus Master Mode allowing it to DMA directly into the entire address space of the above microprocessors.

A programmable mode of operation allows byte addressing in one of two ways: A Byte/Word control signal compatible with the 8086 and Z8000, or an Upper Data Strobe and Lower Data Strobe signal compatible with microprocessors such as the 68000. A programmable polarity on the Address Strobe signal eliminates the need for external logic. The LANCE interfaces with both multiplexed and demultiplexed data buses and features control signals for address/data bus transceivers. See Figure 1-5.

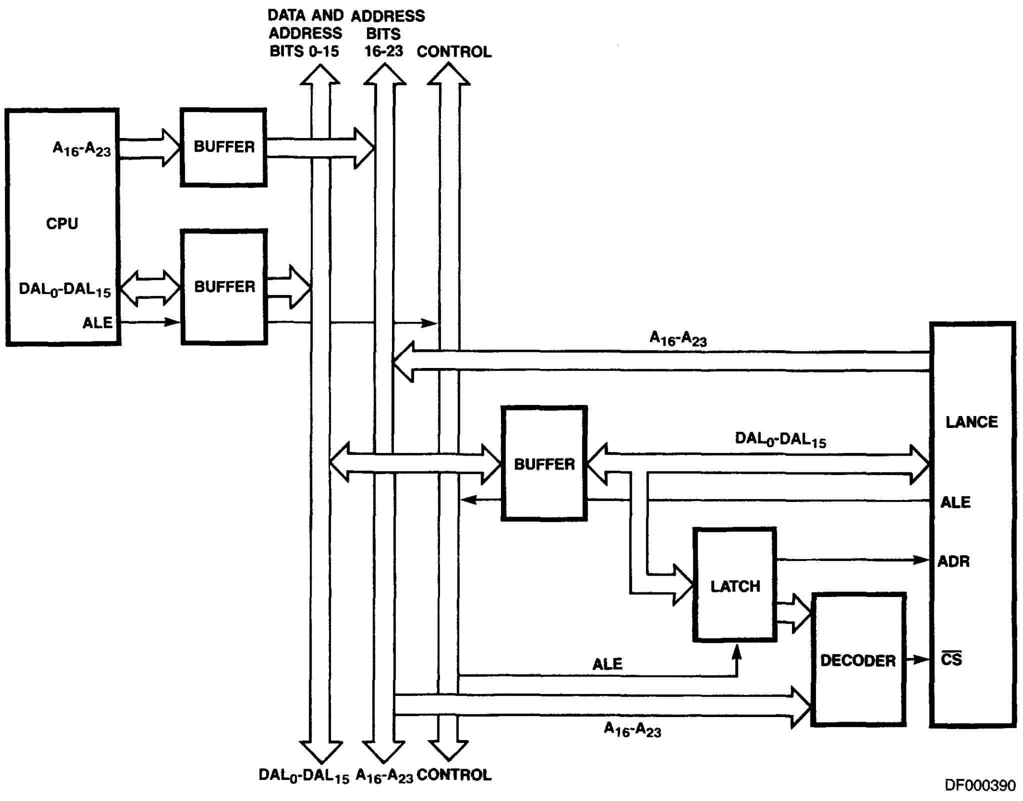
During initialization, the CPU loads the starting address of the Initialization Block into two internal control registers. The LANCE has four internal control and status registers ($\text{CSR}_{0,\ 1,\ 2,\ 3}$) which are used for various functions such as the loading of the Initialization Block address, different programming modes and status conditions. The host processor communicates with the LANCE during the initialization phase—for demand transmission and periodically to read the status bits following interrupts. All other transfers to and from the memory are handled as DMA under microword control.

Interrupts to the microprocessor are generated by the LANCE upon: 1) completion of its initialization routine, 2) the reception of a packet, 3) the transmission of a packet, 4) transmitter timeout error, 5) a missed packet and 6) memory error.

The cause of the interrupt is ascertained by reading CSR_0. Bit (06) of CSR_0, (INEA) enables or disables interrupts to the microprocessor. In systems where polling is used in place of interrupts, bit (07) of CSR_0 (INTR) indicates an interrupt condition.

The basic operation of the LANCE consists of two, distinct modes: transmit and receive. In the transmit mode, the LANCE chip directly accesses data (in a transmit buffer) in memory. It prefaces the data with a preamble, sync pattern, and calculates and appends a 32-bit CRC. This packet is then ready for transmission to the Am7992A SIA. On transmission, the first byte of data loads into the 48-byte FIFO. The LANCE then begins to transmit preamble while simultaneously loading the rest of the packet into FIFO for transmission.

a. Multiplexed Bus



b. Demultiplexed Bus

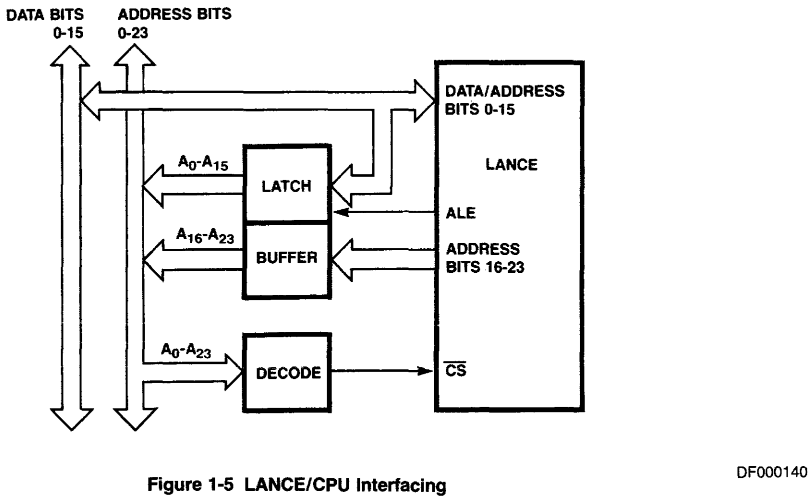


Figure 1-5 LANCE/CPU Interfacing

In the receive mode, packets are sent via the SIA to the LANCE. The packets are loaded into the 48-byte FIFO for preparation of automatic downloading into buffer memory. A CRC is calculated and compared with the CRC appended to the data packet. If the calculated CRC checksum doesn't agree with the packet CRC, an error bit is set.

1.7 PIN DESCRIPTION

DAL_{00}-DAL_{15} **Data/Address Lines (Input/Output 3-State)**
The time multiplexed Address/Data bus. During the address portion of a memory transfer, DAL_{00}-DAL_{15} contains the lower 16 bits of the memory address. The upper 8 bits of address are contained in A_{16}-A_{23}.

During the data portion of a memory transfer, DAL_{00}-DAL_{15} contains the read or write data, depending on the type of transfer.

The LANCE drives these lines as a Bus Master and as a Bus Slave.

A_{16}-A_{23} **High Order Address Bus (Output 3-State)**
Additional address bits that access a 24-bit address space. These lines are driven as a Bus Master only.

READ **(Input/Output 3-State)**
Indicated the type of operation to be performed in the current bus cycle. This signal is an output when the LANCE is a Bus Master.

High — Data is taken off the DAL by the LANCE.

Low — Data is placed on the DAL by the LANCE.

The signal is an input when the LANCE is a Bus Slave.

High — Data is placed on the DAL by the LANCE.

Low — Data is taken off the DAL by the LANCE.

$\overline{BM_0}$/BYTE $\overline{BM_1}$/$\overline{BUSAKO}$ I/O pins 15 and 16 are programmable through bit (00) of CSR_3

$\overline{BM_0}$, $\overline{BM_1}$

If CSR_3 (00) BCON = 0,
I/O Pin 15 = $\overline{BM_0}$ (Output 3-state)
I/O Pin 16 = $\overline{BM_1}$ (Output 3-state)

$\overline{BM_0}$, $\overline{BM_1}$ (Byte Mask). This indicates the byte(s) on the DAL are to be read or written during this bus transaction. The LANCE/ drives these lines only as a Bus Master. It ignores the Byte Mask lines when it is a Bus Slave and assumes word transfers.

Byte selection using Byte Mask is done as described by the following table.

$\overline{BM_1}$	$\overline{BM_0}$	
LOW	LOW	Whole Word
LOW	HIGH	Upper Byte
HIGH	LOW	Lower Byte
HIGH	HIGH	None

BYTE, $\overline{BUSAKO}$

If CSR_3 (00) BCON = 1,
I/O Pin 15 = BYTE (Output 3-state)
I/O Pin 16 = $\overline{BUSAKO}$ (Output)

Byte selection may also be done using the BYTE line and DAL_{00} line, latched during the address portion of the bus cycle. The LANCE drives BYTE only as a Bus Master and ignores it when a Bus Slave selection is done (similar to $\overline{BM_0}$, $\overline{BM_1}$.

Byte selection is done as outlined in the following table.

BYTE	DAL_{00}	
LOW	LOW	Whole Word
LOW	HIGH	Illegal Condition
HIGH	LOW	Lower Byte
HIGH	HIGH	Upper Byte

$\overline{BUSAKO}$ is a bus request daisy chain output. If the chip is not requesting the bus and it received $\overline{HLDA}$, $\overline{BUSAKO}$ will be driven low. If the LANCE is requesting the bus when it received $\overline{HLDA}$, $\overline{BUSAKO}$ will remain high.

Byte Swapping

In an effort to be compatible with the variety of 16-bit microprocessors available to the designer, the LANCE may be programmed to swap the position of the upper and lower order bytes on data involved in transfers with the internal FIFO.

Byte swapping is done when BSWP = 1. The most significant byte of the word in this case will appear on DAL lines 7-0 and the least significant byte on DAL lines 15-8.

When BYTE = H (indicating a byte transfer) the table indicates on which part of the 16-bit data bus the actual data will appear.

Whenever the byte swap is activated, the only data that is swapped is data traveling to and from the Silo.

	Mode Bits	
Signal Line	BSWP = 0 and BCON = 1	BSWP = 1 and BCON = 1
BYTE = L and DAL_{00} = L	Word	Word
BYTE = L and DAL_{00} = H	Illegal	Illegal
BYTE = H and DAL_{00} = H	Upper Byte	Lower Byte
BYTE = H and DAL_{00} = L	Lower Byte	Upper Byte

$\overline{CS}$ **Chip Select (Input)**
Indicates, when asserted, that the LANCE is the slave device of the data transfer. $\overline{CS}$ must be valid throughout the data portion of the bus cycle. $\overline{CS}$ must not be asserted when $\overline{HLDA}$ is LOW.

ADR	**Register Address Port Select (Input)** When LANCE is slave, ADR indicates which of the two register ports is selected. ADR LOW selects register data port; ADR HIGH selects register address port, ADR must be valid throughout the data portion of the bus cycle and is only used by the LANCE when CS is low.
ALE/$\overline{\text{AS}}$	**Address Latch Enable (Output 3-State)** Used to demultiplex the DAL lines and define the address portion of the bus cycle. This I/O pin is programmable through bit (01) of CSR$_3$. As ALE (CSR$_3$ (01), ACON = 0), the signal transitions from a HIGH to a LOW during the address portion of the transfer and remains low during the data portion. ALE can be used by a slave device to control a latch on the bus address lines. When ALE is high the latch is open and when ALE goes low the latch is closed. As $\overline{\text{AS}}$ (CSR$_3$ (01), ACON = 1), the signal pulses LOW during the address portion of the bus transaction. The low to high transition of AS can be used by a slave device to strobe the address into a register. The LANCE drives the ALE/$\overline{\text{AS}}$ line only as a Bus Master.
$\overline{\text{DAS}}$	**Data Strobe (Input/Output 3-State)** Defines the data portion of the bus transaction. $\overline{\text{DAS}}$ is high during the address portion of a bus transaction and low during the data portion. The low to high transition can be used by a slave device to strobe bus data into a register. $\overline{\text{DAS}}$ is driven only as a Bus Master.
$\overline{\text{DALO}}$	**Data/Address Line Out (Output 3-State)** An external bus transceiver control line. $\overline{\text{DALO}}$ is asserted when the LANCE drives the DAL lines. $\overline{\text{DALO}}$ is low only during the address portion if the transfer is a READ. It is low for the entire transfer if the transfer is a WRITE. $\overline{\text{DALO}}$ is driven only when LANCE is a Bus Master.
$\overline{\text{DALI}}$	**Data/Address Line In (Output 3-State)** An external bus transceiver control line. $\overline{\text{DALI}}$ is asserted when the LANCE reads from the DAL lines. It is low during the data portion of a READ transfer, and remain high for the entire transfer if it is a WRITE. $\overline{\text{DALI}}$ is driven only when LANCE is a Bus Master.
$\overline{\text{HOLD}}$ **$\overline{\text{BUSRQ}}$**	**Bus Hold Request (Output Open Drain)** Asserted by the LANCE when it requires access to memory. $\overline{\text{HOLD}}$ is held LOW for the entire ensuing bus transaction. The function of this pin is programmed through bit (00) of CSR$_3$ Bit (00) of CSR$_3$ is cleared when $\overline{\text{RESET}}$ is asserted. When CSR$_3$ (00) BCON = 0 I/O pin 17 = $\overline{\text{HOLD}}$ (Output Open Drain) When CSR$_3$ (00) BCON = 1 I/O pin 17 = $\overline{\text{BUSRQ}}$ (Output Open Drain) $\overline{\text{BUSRQ}}$ will be asserted only if I/O pin 17 is high prior to assertion.
$\overline{\text{HLDA}}$	**Bus Hold Acknowledge (Input)** A response to $\overline{\text{HOLD}}$. When $\overline{\text{HLDA}}$ is low in response to the chip's assertion of $\overline{\text{HOLD}}$, the chip is the Bus Master. $\overline{\text{HLDA}}$ deasserts upon the deassertion of $\overline{\text{HOLD}}$.
$\overline{\text{INTR}}$	**Interrupt (Output Open Drain)** An attention signal that indicates, when active, that one or more of the following CSR$_0$ status flags is set: BABL, MERR, MISS, RINT, TINT or IDON. $\overline{\text{INTR}}$ is enables by bit 06 of CSR$_0$ (INEA = 1). $\overline{\text{INTR}}$ remains asserted until the source of Interrupt is removed.
RX	**Receive (Input)** Receive Input Bit Stream.
TX	**Transmit Enable (Output)** Transmit Output Bit Stream.
TENA	**Transmit (Output)** Transmit Output Bit Stream enable. When asserted, it enables the external transmit output.
RCLK	**Receive Clock (Input)** A 10MHz square wave synchronized to the Receive data and only active while receiving an Input Bit Stream.
CLSN	**Collision (Input)** A logical input that indicates that a collision is occurring on the channel.
RENA	**Receive Enable (Input)** A logical input that indicates the presence of carrier on the channel.
TCLK	**Transmit Clock (Input)** 10MHz clock.
$\overline{\text{READY}}$	**(Input/Output Open Drain)** When the LANCE is a Bus Master, $\overline{\text{READY}}$ is an asynchronous acknowledgement from the bus memory that it will accept data in a WRITE cycle or that it has put data on the DAL lines in a READ cycle. As a Bus Slave, the LANCE asserts $\overline{\text{READY}}$ when it has put data on the DAL lines during a READ cycle or is about to take data off the DAL lines during a write cycle. $\overline{\text{READY}}$ is a response to $\overline{\text{DAS}}$ and will return HIGH after $\overline{\text{DAS}}$ has gone HIGH. $\overline{\text{READY}}$ is an input when the LANCE is a Bus Master and an output when the LANCE is a Bus Slave.
$\overline{\text{RESET}}$	**(Input)** Bus Reset Signal. Causes the LANCE to cease operation, clears its internal logic, and enter an idle state. The stop bit in CSR$_0$ is also set. A 3.3K pull-up resistor should be used at the $\overline{\text{RESET}}$ pin of the LANCE.
V_{CC}	Power supply pin +5 volts ±5%. A 0.1μf and a 10μf decoupling capacitors should be used for V_{CC} to V_{SS}.
V_{SS}	Ground Pins 1 and 24 should be connected together externally, as close to the chip as possible.

1.8 FUNCTIONAL DESCRIPTION

1.8.1 ADDRESSING

Packets can be received using 3 different destination addressing schemes: physical, logical, and promiscuous.

The first type is a full comparison of the 48-bit destination address in the packet with the node address programmed into the LANCE during an initialization cycle.

There are two types of logical address. One is group type mask where the 48-bit address in the packet is put through a hash filter in order to map the 48-bit physical addresses into 1 of 64 logical groups. If any of these 64 groups have been preselected as the logical address, then the 48-bit address is stored in main memory. At this time, a look up is performed comparing the 48-bit incoming address with the prestored 48-bit logical address. The mode can be useful if sending packets to all of a particular type of device simultaneously (i.e., send a packet to all file servers or all printer servers). Additional details on logical addressing can be found in the INITIALIZATION section under "Logical Address Filter". The second logical address is a broadcast address where all nodes on the network receive the packet.

The last receive mode of operation is the so-called "promiscuous mode" in which a node will accept all packets on the coax regardless of their destination address.

1.8.2 COLLISION DETECTION AND IMPLEMENTATION

The Ethernet CSMA/CD network access algorithm is implemented completely within the LANCE. In addition to listening for a clear coax before transmitting, Ethernet handles collisions in a predetermined way. Should two transmitters attempt to seize the coax at the same time, they will collide and the data on the coax will be garbled. The transmitting nodes listen while they transmit, detect the collision, then continue to transmit for a predetermined length of time to "jam" the network and ensure that all nodes have recognized the collision. The transmitting nodes then delay a random amount of time according to the Ethernet "truncated binary backoff" algorithm in order that the colliding nodes don't try to repeatedly access the network at the same time. Up to 16 attempts to access the network are made by the LANCE before reporting back an error due to excessive collisions.

COLLISION JAM

Collisions are detected by monitoring the CLSN I/O pin. If CLSN becomes asserted during a frame transmission, TENA will remain asserted for at least 32 additional bit times (including C:SM synchronization). This additional transmission after collision is called COLLISION JAM. If collision occurs during the transmission of the preamble, the LANCE continues to send the preamble followed by a JAM pattern. If collision occurs after the preamble, the LANCE sends the JAM pattern following the transmission of the current byte. The JAM pattern is any pattern except the CRC bytes.

COLLISION DURING RECEPTION

If CLSN becomes asserted during the reception of a packet, this reception is immediately terminated. The action taken depends on the timing of collision detection. The packet is rejected and the Silo pointer is reset if the collision occurs within six byte times (4.8us) because of an address mismatch. The packet is rejected as a runt packet if the collision occurs within 64 byte times (51.2us). If a collision occurs after 64 byte times (late collision), a truncated packet is written to the memory buffer with the CRC error bit usually set in the Status Word of the receive ring. Late collision error is not recognized in receive mode.

COLLISION DURING TRANSMISSION

When a transmission is terminated due to the assertion of CLSN (a collision within the first 64 byte times of the packet), the LANCE retries it 15 times. The LANCE does not re-read the descriptor entries from the TX ring upon each collision. The descriptor entries for the current buffer are internally saved. The scheduling of the retransmissions is determined by a controlled randomized process called "truncated binary exponential backoff". Upon the completion of the COLLISION JAM interval, the LANCE calculates a delay before retransmitting. The delay is an integral multiple of the slot time. The slot time is 64 byte times (512 bit times). The number of slot times to delay before the nth retransmission is chosen as a uniformly distributed random integer in the range of:

$0 \leqslant r < 2^k$

where: $k = \text{minimum}(n, 10)$

For example, if this is the third retry,

$n = 3$
$k = \min(3,10) = 3.$
$2^3 = 8$

Therefore, r is a random number from zero to eight.

If all 16 attempts fail, the LANCE sets the RTRY bit in the current Transmit Message Descriptor 3 (TMD_3) in memory, gives up ownership (sets the own bit to zero) for this packet, and processes the next packet in transmit ring for transmission. If a late collision occurs, the LANCE does not transmit the same packet again. It terminates the transmission, notes the LCOL error in TMD_3, and transmits the next packet in the ring.

COLLISION - MICROCODE INTERACTION

The microprogram uses the time provided by Collision Jam, Interpacket Delay, and the backoff interval to restore the address and byte counts internally and to start loading the Silo in anticipation of retransmission. To utilize the channel properly, it is important that the LANCE be ready to transmit when the backoff interval elapses.

TIME DOMAIN REFLECTOMETRY

The LANCE contains a time domain reflectometry counter (TDR). The TDR counter is ten bits wide. It counts at a 10MHz rate. It is cleared by the microprogram and counts upon the assertion of RENA during transmission. Counting stops if CLSN becomes true or if RENA goes inactive. The counter does not wrap around. When the counter reaches a value of all ones, that value is held until it is cleared. The value in the TDR is written into memory following the transmission of the packet. TDR is used to determine the location of suspected cable faults.

COLLISION PAIR TEST (HEARTBEAT, SQE)

An open in the cable causes a reflection which the coaxial cable transceiver detects as a collision. A short in the cable causes RENA not to become active or to drop out after becoming active. During the Interpacket Delay following the negation of TENA, the CLSN input is asserted (as a result of collision becoming active) by some transceivers as a self-test called Heartbeat or SQE (Signal Quality Error) test. If the CLSN input is not asserted within 2.0us following the completion of transmission (after TENA goes low), then the LANCE will set the CERR bit in CSR_0. CERR error does not cause an interrupt to occur (INTR = 0).

1.8.3 ERROR REPORTING AND DIAGNOSTICS

Extensive error reporting is provided by the LANCE. Error conditions reported relate to either the network as a whole or to data packets. Network-related errors are recorded as flags in the CSRs and are examined by the CPU following interrupt. Packet-related errors are written into descriptor entries corresponding to the packet.

System-related errors include:

1. Babbling Transmitter — Transmitter attempting to transmit more than 1518 data bytes in one packet.
2. Collision — Collision detection circuitry non-functional.
3. Missed Packet — Insufficient buffer space.
4. Memory Timeout — Memory response failure

Packet-related errors include:

1. CRC — Invalid data
2. Framing — Packet did not end on byte boundary
3. Overflow/Underflow — Abnormal latency in servicing a DMA request
4. Buffer — Insufficient buffer space available

The LANCE performs several diagnostic routines that enhance the reliability and integrity of the system. These include a CRC logic check and two loopback modes (internal/external). Errors may be introduced into the system to check error detection logic. A Time Domain Reflectometer is incorporated into the LANCE to aid system designers to locate faults in the Ethernet cable. Shorts and opens manifest themselves in reflections that are sensed by the TDR.

CYCLIC REDUNDANCY CHECK (CRC)

The LANCE utilizes the 32 bit CRC function used in the Autodin-II network. Refer to the Ethernet specification (Section 6.2.4 Frame Check Sequence Field and Appendix C, CRC Implementation) for more detail. The LANCE requirements for the CRC logic are:

1. TRANSMISSION — Mode bit 2 (LOOP) = 0, Mode bit 3 (DTCR) = 0. The LANCE calculates the CRC from the first bit following the Start bit to the last bit of the data field. The CRC value inverted is appended to the transmission in one unbroken bit stream.
2. RECEPTION — Mode bit 2 (LOOP) = 0. The LANCE performs a check on the input bit stream from the first bit following the Start bit to the last bit in the frame. The LANCE continually samples the state of the CRC check on framed byte boundaries and, when the incoming bit stream stops, the last sample determines the state of the CRC error. Framing error (FRAM) is not reported if there is no CRC error.
3. LOOPBACK — Mode bit 2 (LOOP) = 1, Mode bit 3 (DTRC) = 0. The LANCE generates and appends the CRC value to the outgoing bit stream as in transmission but does not perform the CRC check of the incoming bit stream.
4. LOOPBACK — Mode bit 2 (LOOP) = 1, Mode bit 3 (DTRC) = 1. The LANCE performs the CRC check on the incoming bit stream as in reception but does not generate or append the CRC value to the outgoing bit stream during transmission.

LOOPBACK

The LANCE normally operates as a half-duplex device. However, to provide an on-line operational test of the LANCE, a pseudo-full duplex mode is provided. In this mode, simultaneous transmission and reception of a loopback packet are enabled with the following constraints:

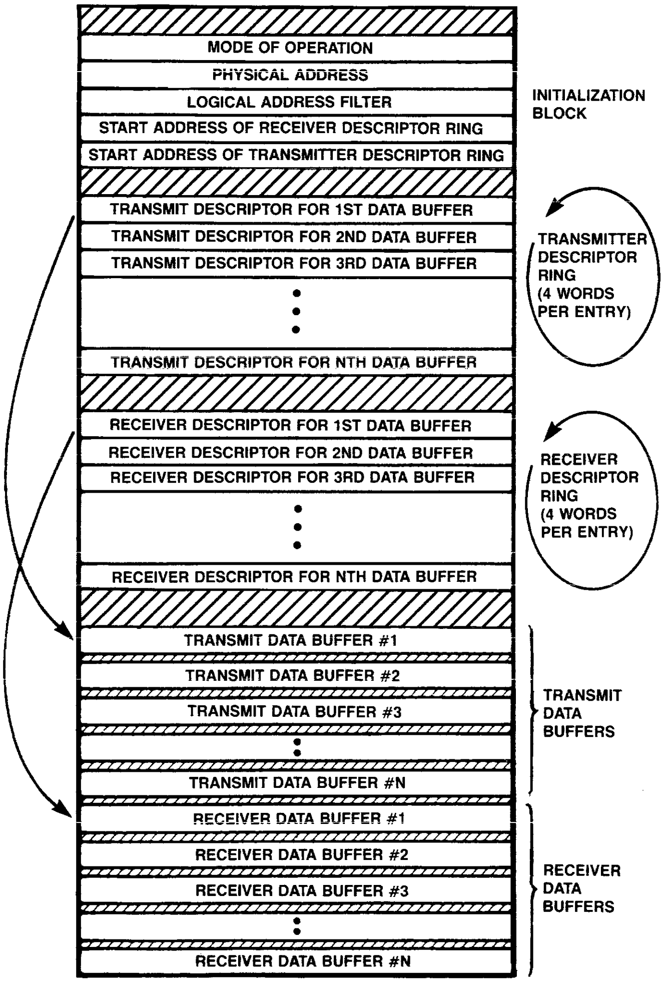
1. The packet length must be from eight to 32 bytes long, exclusive of the CRC bits.
2. Serial transmission does not begin until the Silo contains the entire output packet.
3. Moving the input packet from the Silo to the memory does not begin until the serial input bit stream terminates.
4. CRC may be either generated and appended to the output serial bit stream or may be checked on the input serial bit stream but not both in the same transaction.
5. In internal loopback, the packets should be addressed to the node itself. If not, the chip must be re-initialized with non-matching source and destination addresses after each test.
6. In external loopback, multicast addressing can be used only when DTCR = 1 is in the mode register. In this case, the user needs to append the CRC bytes.
7. In external loopback, the Silo pointers sometimes get misaligned with heavy traffic on the network. It is recommended that the user repeats the test several times after the first time it fails. This can help isolate the failure to Silo pointer misalignment or other causes. For detail External Loopback Test Procedure, please refer to Appendix C.

Loopback is controlled by the INTL, DTCR, and LOOP (bits 2, 3,and 6) of the MODE register.

1.8.4 BUFFER MANAGEMENT

A key feature of the LANCE and its on-board DMA channel is the flexibility and speed of communication between the LANCE and the host microprocessor through common memory locations. The basic organization of the buffer management is a circular queue of tasks in memory called descriptor rings. Refer to Figure 1-6 and Figure 1-7.

The transmit and receive operations are described by separate descriptor rings. Up to 128 tasks may be queued on a descriptor ring awaiting execution by the LANCE. Each entry in a descriptor ring contains a pointer to a data buffer and an entry for the length of the buffer.



MODE OF OPERATION
PHYSICAL ADDRESS
LOGICAL ADDRESS FILTER
START ADDRESS OF RECEIVER DESCRIPTOR RING
START ADDRESS OF TRANSMITTER DESCRIPTOR RING
INITIALIZATION BLOCK
TRANSMIT DESCRIPTOR FOR 1ST DATA BUFFER
TRANSMIT DESCRIPTOR FOR 2ND DATA BUFFER
TRANSMIT DESCRIPTOR FOR 3RD DATA BUFFER
TRANSMIT DESCRIPTOR FOR NTH DATA BUFFER
TRANSMITTER DESCRIPTOR RING (4 WORDS PER ENTRY)
RECEIVER DESCRIPTOR FOR 1ST DATA BUFFER
RECEIVER DESCRIPTOR FOR 2ND DATA BUFFER
RECEIVER DESCRIPTOR FOR 3RD DATA BUFFER
RECEIVER DESCRIPTOR FOR NTH DATA BUFFER
RECEIVER DESCRIPTOR RING (4 WORDS PER ENTRY)
TRANSMIT DATA BUFFER #1
TRANSMIT DATA BUFFER #2
TRANSMIT DATA BUFFER #3
TRANSMIT DATA BUFFER #N
TRANSMIT DATA BUFFERS
RECEIVER DATA BUFFER #1
RECEIVER DATA BUFFER #2
RECEIVER DATA BUFFER #3
RECEIVER DATA BUFFER #N
RECEIVER DATA BUFFERS

DF000130

Figure 1-6 LANCE/Processor Memory Interface

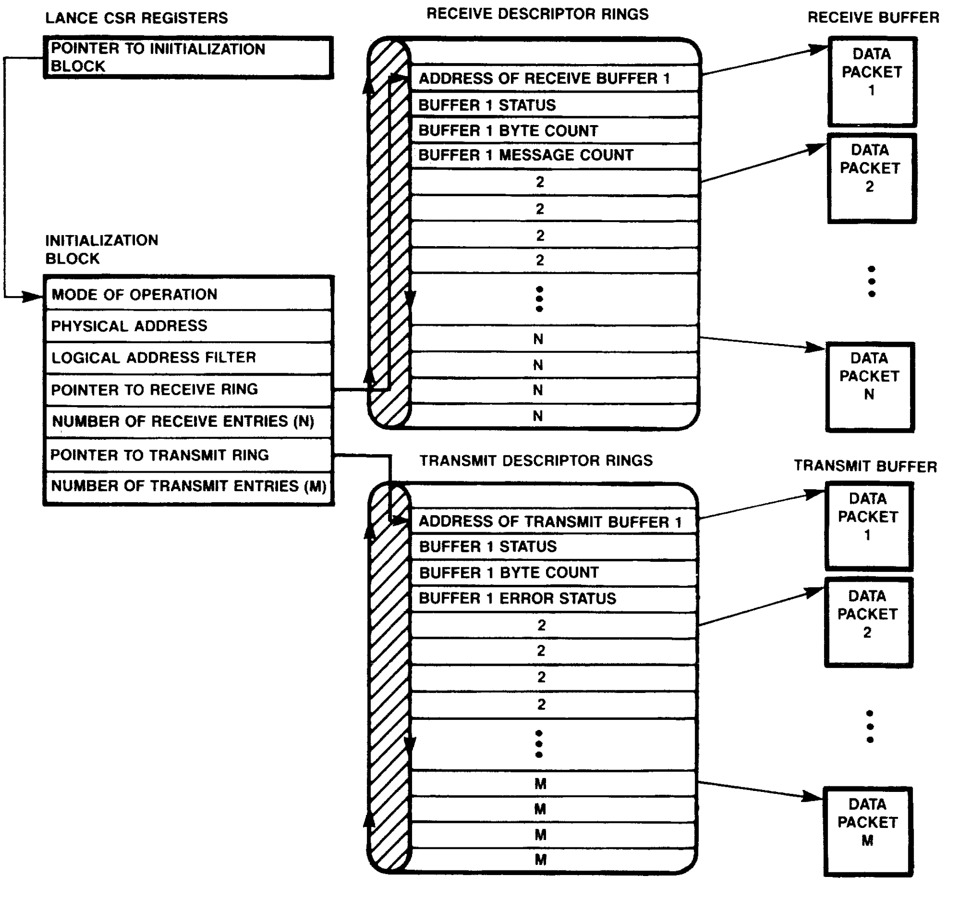


Figure 1-7 LANCE Memory Management

Data buffers can be chained to handle a packet that is longer than the buffer. The LANCE searches the descriptor rings in a lookahead operation to determine the next empty buffer when chaining is needed or when back-to-back packets are being received. As each buffer is filled, an OWN bit in the descriptor ring entry is reset, allowing the host processor to process the data in the buffer.

RING ACCESS BY THE LANCE

After the LANCE is initialized and started, the CPU and the LANCE communicate via transmit and receive rings (in memory) for packet transmission and reception. The LANCE contains eight 16-bit registers. The first four registers store the four entries in the descriptor that points to the current buffer. The other four registers store the same information for the descriptor that points to the next buffer to be processed (the lookahead buffer). Refer to Figure 1-2.

TRANSMIT RING BUFFER MANAGEMENT

After the LANCE has been initialized, whenever the Ethernet is inactive, the LANCE automatically polls the first transmit ring entry in memory. The polling occurs every 1.6ms and consists of reading the status word (TMD_1) of the transmit ring entry until it finds the OWN bit and STP bit set to one. The OWN bit set to one in the transmit ring indicates a buffer ready to be transmitted. The STP bit indicates that this is the start of a packet.

When the LANCE finds that a buffer is to be transmitted, it transfers the low order bits of the buffer address from TMD_0 and byte count from TMD_2 into the current buffer pointer in LANCE. Each of these memory reads is done with a separate arbitration cycle for each transfer. The high order bits of the address are in TMD_1 which have already been transferred.

If the packet is larger than the buffer, the buffers are data-chained and the ENP (end of the packet) is zero in all of the buffers except the buffer containing the end of the packet. If ENP equals zero for the buffer being transmitted, the LANCE does one lookahead for the next buffer. The lookahead is done in between data transfers to or from the Silo. It consists of one single word DMA read (TMD_1) to find out if it owns the buffer and two more reads, (TMD_0 and TMD_2) if it does own it.

If LANCE does own the next buffer, it transfers the address and byte count of this buffer into the lookahead buffer in the LANCE. After the LANCE completes transferring the current buffer into the Silo, it clears the OWN bit for this buffer and immediately starts loading the Silo from the next buffer.

If the LANCE finds during the lookahead that it does not own the next transmit Descriptor Table Entry (DTE), it will transmit the current buffer and update the status of the current ring entry by setting the BUFF bit in TMD_3. BUFF error causes the transmit section of the LANCE to turn off (CSR_0, TXON = 0). The chip has to be re-initialized to turn transmitter on.

While transmitting, if a RTRY or LCOL (late collision) occurs, the LANCE stops transmitting the packet. It clears the OWN bit in TMD_1 and sets the TINT bit in CSR_0 causing an interrupt if INEA = 1 (interrupts enabled). The LANCE does not transmit the remainder of the buffers in this packet. Instead, it clears the own bit and sets the TINT bit in each Descriptor Table Entry it polls until it finds a buffer with both the STP bit and OWN bit set indicating the start of the next packet to send.

When the transmit buffers are not data-chained (current descriptor's ENP = 1), the LANCE does not perform any lookahead operation. It transmits the current buffer and updates the status TMD_1 (and TMD_3 if an error occurred) and then clears the OWN bit in TMD_1. The LANCE immediately checks the next descriptor in the ring to see if another packet is ready to be sent. If it is ready, the LANCE proceeds to read the buffer address and byte count and to transmit the packet. If no packet is ready, it resumes polling the ring every 1.6ms.

After each complete packet has been transmitted from the Silo to the cable, the LANCE updates the status in TMD_1 (and TMD_3 if an error occurred). Then it releases the last buffer of the packet to the CPU. It then immediately polls the transmit descriptor ring for another packet to send. This guarantees the back-to-back transmission of packets when packets are ready to transmit.

RECEIVE RING BUFFER MANAGEMENT

When the receiver is enabled, the LANCE polls the first receive ring entry once every 1.6 ms for buffer ownership so that it will be ready for the next incoming packet. After the LANCE owns a buffer, it reads the remainder of the buffer address (from RMD_0) and the byte count (from RMD_2) into the current buffer in the LANCE. After the LANCE has filled the first buffer with data from a received packet, it polls the next receive descriptor ring entry for the next buffer to fill.

When a packet arrives from the cable, and the LANCE does not own any buffer, it polls the receive ring entry once for the buffer. If it does not get the buffer, it sets the MISS error bit in CSR_0 and does not poll the receive ring entry again until the packet ends. (In the previous revision, Rev.B, the LANCE polled the entry every few microseconds until it either received the buffer or the Silo overflowed.)

Assuming that the LANCE owns a buffer when a packet arrives from the cable, the LANCE does one lookahead operation in between periods of data transfer from the Silo. The LANCE needs to have a buffer ready in case the current buffer needs data-chaining. The lookahead operation consists of one single word DMA read (RMD_1) to find out if it owns the buffer and two more reads, (RMD_0 and RMD_2) if it does own it.

Whether the LANCE owns the next buffer or not, it will transfer data from the Silo to the current buffer in burst mode (eight words for each DMA arbitration cycle). If the packet being received requires data-chaining and the LANCE does not own the next buffer, the LANCE updates the current buffer status by setting the BUFF error bit.

If the LANCE does own the next buffer and needs data-chaining, when the first buffer is full it releases it by clearing the OWN bit. It then starts filling the next buffer. In between data transfers from the Silo, it does one lookahead operation for ownership of the next buffer to be ready for more data-chaining if needed.

After each complete packet has been received and transferred from the Silo to the buffers, the LANCE updates the status in RMD_1 (and RMD_3 if an error occurred). Then it releases the last buffer of the packet to the CPU.

1.8.5 LANCE INTERFACE

CSR bits such as ACON, BCON, and BSWP are used for programming the pin functions used for different interfacing schemes. For example, ACON is used to program the polarity of the Address Strobe signal ALE/$\overline{\text{AS}}$.

BCON is the byte control bit that redefines the BYTE MASK I/O pins to handle odd address boundaries and redefines the $\overline{\text{HOLD}}$ I/O pin for daisy-chain operation when BCON = 1. In addressing word organized, byte addressable memories (BCON = 1), the BYTE signal is decoded along with the least significant address bit to determine odd or even (upper or lower byte) byte address.

When BCON = 1 in a daisy-chain operation, the $\overline{\text{BM}_1}$ pin is used as $\overline{\text{BUSAKO}}$ the $\overline{\text{HOLD}}$ pin is used as $\overline{\text{BUSRQ}}$ and the pin $\overline{\text{HLDA}}$ is used. When a DMA controller is used for arbitration, only the $\overline{\text{HOLD}}$ and $\overline{\text{HLDA}}$ pins are used.

When BCON = 0, the BYTE MASK pins 15 and 16 ($\overline{\text{BM}_0}$ and $\overline{\text{BM}_1}$) are used to specify byte selection on the DAL lines as whole word, upper byte, lower byte, or none.

All data transfers from the LANCE in the Bus Master mode are timed by ALE, $\overline{\text{DAS}}$, and $\overline{\text{READY}}$. The automatic adjustment of the LANCE cycle by the $\overline{\text{READY}}$ signal allows synchronization with variable cycle time memory due either to memory refresh or to dual port access. Bus cycles are a minimum of 600ns duration and can be increased in 100ns increments.

READ SEQUENCE (Master Mode)

The read memory cycle is started by placing valid addresses on DAL_{00}-DAL_{15} and A_{16}-A_{23}. The BYTE MASK signals are asserted to indicate a word, upper byte, or lower byte memory reference. READ indicates the type of cycle. ALE or $\overline{\text{AS}}$ are pulsed and the trailing edge of either can be used to latch addresses. DAL_{00}-DAL_{15} go into a 3-state mode and $\overline{\text{DAS}}$ falls low to signal the beginning of the memory access. The memory responds by placing $\overline{\text{READY}}$ low to indicate that the DAL lines have valid data. The LANCE then latches memory data on the rising edge of $\overline{\text{DAS}}$ which in turn ends the memory cycle and $\overline{\text{READY}}$ returns to high.

Data transfers between the bus transceivers and the LANCE are controlled by the $\overline{\text{DALI}}$ and $\overline{\text{DALO}}$ lines. $\overline{\text{DALI}}$ diverts data toward the LANCE. $\overline{\text{DALO}}$ diverts data or addresses away from the LANCE. During a read cycle, $\overline{\text{DALO}}$ goes inactive before $\overline{\text{DALI}}$ becomes active to avoid "spiking" the bus transceivers.

WRITE SEQUENCE (Master Mode)

The write to memory cycle is similar to the read cycle except that the DAL lines change from containing addresses to containing data after ALE or $\overline{\text{AS}}$ goes inactive. After data is valid on the bus, $\overline{\text{DAS}}$ goes active. Data to memory is held valid for some time after $\overline{\text{DAS}}$ goes inactive.

1.8.6 LANCE DMA TRANSFER (Bus Master Mode)

The LANCE has two types of DMA transfers: Burst Mode and Single Word transfers.

BURST MODE

Burst mode is used for the transfer of packets between the Silo and processor memory. Each burst transfer is eight words long and is done in one DMA arbitration cycle. After the LANCE receives the bus acknowledge ($\overline{\text{HLDA}}$ = low) signal, it does eight word transfers using eight DMA cycles (600 ns per cycle minimum) before it releases the bus request signal ($\overline{\text{HOLD}}$ = low).

Burst transfers are always eight word transfers unless the buffer in memory starts on an odd byte boundary or there are fewer than eight words remaining to be transferred. If the buffer starts on an odd address, one byte and seven words are transferred on the first burst transfer. If fewer than eight words remain to be transferred at the end of a packet, the burst transfer transfers as many full words as remain plus a single byte (if any) as the last transfer in the burst.

SINGLE WORD DMA TRANSFERS

The LANCE initiates single word DMA transfers to access the Transmit Descriptor Ring, Receive Descriptor Ring, and Initialization Block. The LANCE does not initiate any burst transfers when it owns the descriptor and is accessing the descriptor entries (an average of 3 to 4 separate DMA cycles are required for a multi-buffer packet) or when it is reading the Initialization Block.

1.8.7 SILO OPERATION

The Silo provides temporary buffer storage for data being transferred between the parallel bus I/O pins and the serial bus I/O pins in the LANCE. The capacity of the Silo is 48 bytes. It is a first in first out (FIFO) buffer.

TRANSMIT

Data is loaded into the Silo from memory under internal microprogram control. The Silo must have more than eight words (16 bytes) empty before the LANCE requests the DMA bus (by asserting $\overline{\text{HOLD}}$). The LANCE starts sending the preamble to the cable (providing the line is idle) as soon as the first byte is loaded into the Silo from memory. If a collision is detected causing the transmitter to back off, loading the Silo continues until it contains 32 bytes ready for transmission. Reception has priority over transmission during the time that the transmitter is backing off.

RECEIVE

Data from the cable is loaded into the Silo through the serial input shift register during reception. Data is transferred from the Silo into memory under internal microprogram control. The LANCE waits until the Silo has at least eight words (16 bytes) of data before initiating a DMA burst transfer. Neither the preamble nor the sync is loaded into the Silo during reception.

MEMORY BYTE ALIGNMENT

Memory buffers begin and end on arbitrary byte boundaries. Parallel data is byte-aligned between the Silo and DAL lines (DAL_0-DAL_{15}). Byte alignment can be reversed by setting the Byte Swap (BSWP) BIT in CSR_3. Byte Swap (BSWP = 1) is done only on transfers between the Silo and memory. Refer to Table 1-2.

Table 1-2. Silo - Memory Byte Alignment

Word/byte in Memory	BSWP	Silo Byte	DAL Pins
Word from even memory address	0	n	0 - 7
	0	n + 1	8 - 15
	1	n	8 - 15
	1	n + 1	0 - 7
Byte from even memory address	0	n	0 - 7
	1	n	8 - 15
Byte from odd memory address	0	n	8 - 15
	1	n	0 - 7
Word to even memory address	0	n	0 - 7
	1	n + 1	8 - 15
Byte to even memory address	0	n	0 - 7
	0	x	8 - 15
	1	n	8 - 15
	1	x	0 - 7
Byte to odd memory address	0	x	0 - 7
	0	n	8 - 15
	1	x	8 - 15
	1	n	0 - 7

1.8.8 ETHERNET/IEEE 802.3 INTERFACE

The Ethernet (or IEEE 802.3 standard) is the data link and physical level in the communication system. The LANCE performs the encapsulation/decapsulation function of the data link layer and also interfaces the Ethernet through a Serial Interface Adapter (SIA) (Am7992) and a transceiver (Am7995).

A typical Ethernet/IEEE802.3 node is shown in Figure 1-8. This figure also shows a Cheapernet for comparison. Cheapernet costs less because it does not require a drop cable to the transceiver. The transceiver is a part of the DTE. The bulky Ethernet coaxial cable and tap connection is replaced by Rg58 cable and BNC connectors. Cheapernet runs at the same speed as Ethernet.

SERIAL TRANSMISSION

Serial transmission consists of sending an unbroken bit stream from the TX I/O pin of the LANCE. The bit stream has the following format: (Refer to Figure 1-9)

1. Preamble: A stream of alternating ones and zeros 62 bits long supplied by LANCE.
2. Sync.: Two successive ones following the preamble supplied by LANCE.
3. Destination Address: Six byte address supplied by user.
4. Source Address: Six byte address supplied by user.
5. Type: Two byte field for type or length information supplied by user.
6. Data: 46 to 1500 bytes of data serialized with the LSB first supplied by user.
7. CRC: Four byte cyclic redundancy check code supplied by LANCE. It is an inverted 32 bit polynomial calculated from the address, type field, and data field. It is not transmitted if the transmission of the data field is truncated for any reason, CLSN becomes asserted (collision detected), or MODE <3> DTCR = 1 in a normal or loopback transmission mode.

The format for the IEEE 802.3 MAC Frame is shown in Figure 1-10. It is almost identical to the Ethernet format except for the labeling and a pad preceding the CRC. The preamble is shorter and the sync is longer but the code is identical.

Transmission is indicated at the I/O pin by the assertion of TENA with the first bit of the preamble. TENA is negated after the last transmitted bit.

The LANCE starts transmitting the preamble when all of the following conditions are met:

1. There is at least one byte of data to be transmitted in the Silo.
2. No carrier detected. Both RENA and CLSN are inactive.
3. The interpacket delay has elapsed.
4. If a retransmission, the backoff interval has elapsed.

The Interpacket Gap Time (IPG) for back to back transmission is 9.6 to 10.6 microseconds including synchronization. The interpacket delay interval begins immediately after the negation of the RENA signal. During the first 4.1 us of the IPG, RENA is masked internally.

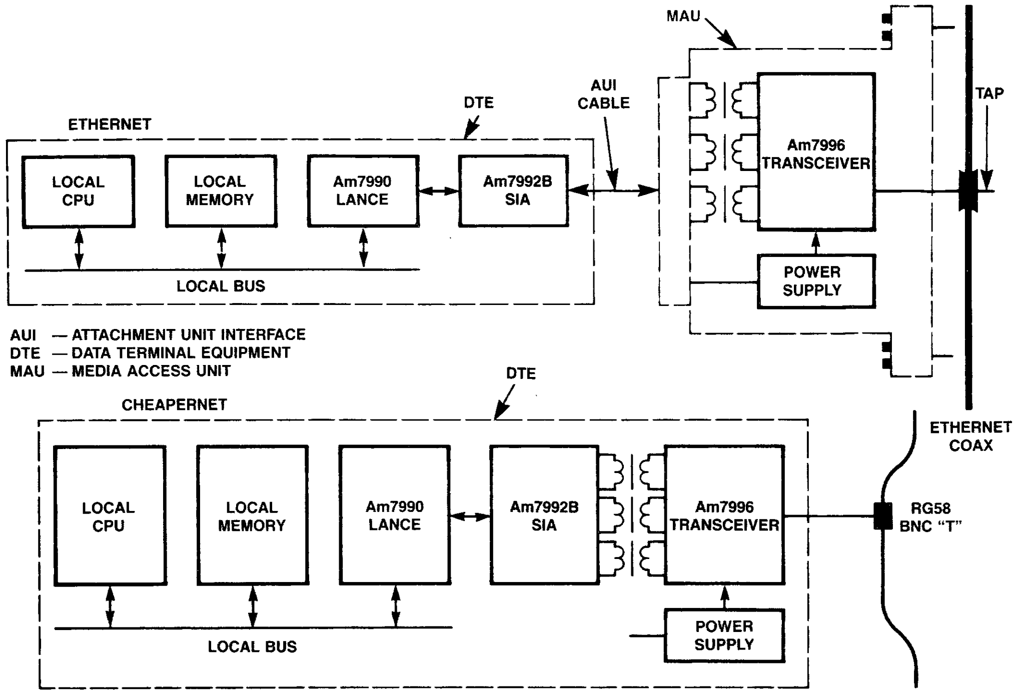
When the LANCE is waiting for the interpacket delay to elapse, it begins transmission immediately after the interpacket delay interval, independent of the state of RENA. However, RENA must be asserted during the time that TENA is high. Otherwise, the LCAR (loss of carrier) bit is set in TMD_3 after the packet has been transmitted.

SERIAL RECEPTION

Serial reception consists of receiving an unbroken bit stream on the RX I/O pin of the LANCE. The Destination Address and data are framed into bytes and enter the Silo. The Source Address and Type field are part of the data which are transparent to the LANCE. They pass through the Silo into memory along with the CRC. The LANCE strips off the preamble and sync bits. Refer to Figure 1-9 and Figure 1-10. The format is:

1. Preamble/Start bit: Two ones occurring a minimum of eight bit times after the assertion of RENA. The last one is the Start bit.
2. Destination Address: The 48 bits (six bytes) following the start bit.
3. Data: The serialized byte stream following the Destination Address. The last four complete bytes are the CRC.

Reception is indicated at the I/O pin by the assertion of RENA and the presence of clock on RCLK while TENA is inactive. After the eighth RCLK, (about 800ns), after RENA is asserted (goes high), the LANCE samples the incoming bits looking for the sync bits "11" following a zero. After it finds the sync bits, it starts calculating CRC on the incoming bits and loading the Silo on byte boundaries.



05998A-2

Figure 1-8 Typical Ethernet/IEEE 802.3 Node

PREAMBLE 1010 . . . 1010	SYNC 1 1	DEST. ADR	SOURCE ADR	TYPE	DATA	FCS
62 BITS	2 BITS	6 BYTES	6 BYTES	2 BYTES	46–1500 BYTES	4 BYTES

05726A-22

Figure 1-9 Ethernet Frame Format

PREAMBLE 1010 . . . 1010	SFD 10101011	DEST. ADR	SOURCE ADR	LENGTH	LLC DATA	PAD	FCS
56 BITS	8 BITS	6 BYTES	6 BYTES	2 BYTES	46–1500 BYTES		4 BYTES

05726A-23

Figure 1-10 IEEE 802.3 MAC Frame Format

The LANCE discards packets with less than 64 bytes (runt packet) and reuses the receive buffer for the next packet. This is the only case when the LANCE discards a packet. A runt packet is normally the result of a collision.

If RENA is asserted and remains asserted during the first 4.1us of IPG following a receive, the LANCE will not receive the packet. If this condition occurs following a transmit, the LANCE starts to look for the sync bits (011) about 800ns (8 bit times) after the 4.1us window has elapsed. Therefore, the packet may be received correctly if at least 8 bits of the preamble are received after the 4.1us window. Otherwise, the packet may contain a CRC error. (The LANCE may be locking to a sync pattern in the middle of data.) The packet may be discarded because the loss of data during the 4.1us window made it a runt packet.

If RENA is asserted after the 4.1us window, the LANCE treats this as the start of a new packet. It starts to look for the sync bits 8 bit times after RENA becomes active.

When a received packet terminates, if it does not end on a byte boundary, a framing error has occurred. The number of bits left over after the last full byte are called dribbling bits. The framing error is reported to the user as follows:

1. If there is no CRC error, then no framing error is reported (FRAM = 0) regardless of the number of dribbling bits.
2. If there is a CRC error, then the framing error is reported (FRAM = 1) if there are one or more dribbling bits.
3. If there are no dribbling bits, there is no framing error. There may or may not be a CRC error.

THE LANCE RECOVERY AND RE-INITIALIZATION

The transmitter and receiver section of the LANCE are turned on via the Initialization Block, Mode Register DRX and DTX bits. The state of the transmitter and the receiver are monitored through the CSR_0 RXON, TXON bits). The LANCE must be re-initialized to transmit/receive if the transmitter and/or receiver has not been turned on during the original initialization. The LANCE must also be re-initialized if either section shuts off because of an error (MERR, UFLO, TX BUFF).

The LANCE should be re-initialized with care. The user should rearrange the descriptors in the transmit and receive rings prior to re-initialization. This is necessary because the transmit and receive descriptor ring pointers are reset to the beginning of the rings during initialization.

To re-initialize the LANCE, the user must stop the LANCE by setting the stop bit in CSR_0 prior to setting the INIT bit (in CSR_0). Because setting the stop bit clears CSR_3, this register must be reprogrammed unless default values are used for BCON, ACON, and BSWP. CSR_1 and CSR_2 must be reloaded when the STOP bit is set.

The LANCE may be restarted without initialization by setting the STRT bit in CSR_0. The STRT bit starts the LANCE in accordance with the parameters setup in the Mode register. If DTX and DRX are set to zero in the Mode Register, the transmitter and the receiver will be turned on again when the STRT bit is set. However, this method of restart is not recommended if the LANCE stopped in the middle of a transmission or reception or if the buffers are data-chained.

ETHERNET/IEEE 802.3 DIFFERENCES

The only differences in the frame formats are:

1. The Ethernet labels the first 62 bits preamble and the next two bits sync. The IEEE 802.3 labels the first 56 bits preamble and the next 8 bits sync (SFD). The coding of the first 64 bits is identical.
2. The Ethernet has a two-byte type field to specify the type of packet. The length is not specified in the packet. The LANCE supplies the packet length during reception. The packet length consists of all the bits starting at the destination address and ending with the last byte of the packet. The Ethernet frame consists of the packet plus the preamble.
3. The IEEE 802.3 has no type field but in its place is a two-byte length field to specify the amount of data in the packet.
4. The IEEE 802.3 has a pad field at the end of the data portion and preceding the CRC. The use of a pad allows the user to send and receive packets that have less than 64 bytes of data. In contrast, Ethernet requires a minimum packet size of 64 bytes for packets that are not data-chained and 100 bytes for packets that are data-chained.
5. Ethernet, Version 2, specifies that the collision detect of the transceiver must be activated during the interpacket gap time.

1.9 LANCE REGISTERS AND BUFFERS

There are four Control and Status Registers (CSRs) resident within the chip. CSR_0 contains the current status information during operation and the Start, Stop, and Initialize control signals. CSR_1 and CSR_2 contain the address in memory of the Initialization Block. CSR_3 contains control bits for Byte Swap, ALE Control, and Byte Control. These registers are described following a description of the two ports used to access the CSRs. The CSRs are accessed through two bus-addressable ports, an address port (RAP) and a data port (RDP).

1.9.1 ADDRESS AND DATA PORTS

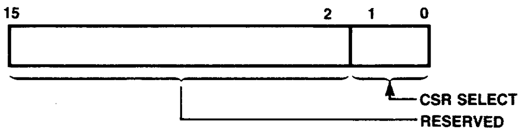
The CSRs are accessed in a two-step operation. The address of the CSR to be accessed is written into the address port (RAP) during a bus slave transaction. During a subsequent bus slave transaction, the data being read from (or written into) the data port (RDP) is read from (or written into) the CSR selected in the RAP.

Once written, the address in RAP remains unchanged until rewritten.

To distinguish the data port from the address port, a discrete I/O pin is provided.

ADR I/O Pin	Port
L	Register Data Port (RDP)
H	Register Address Port (RAP)

REGISTER ADDRESS PORT (RAP)



AF001490

Bit	Name	Description
15:02	RES	Reserved and read as zeroes.
01:00	CSR(1:0)	CSR address select. READ/WRITE. Selects the CSR to be accessed through the RDP. RAP is cleared by Bus RESET.

CSR(1:0)	CSR
00	CSR_0
01	CSR_1
10	CSR_2
11	CSR_3

REGISTER DATA PORT (RDP)

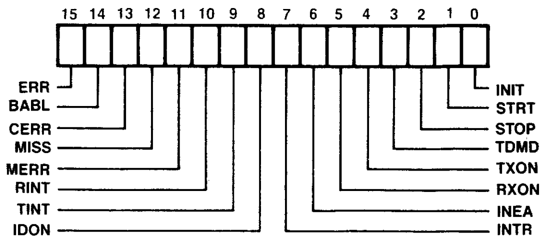
15
0
CSR DATA

AF001450

Bit	Name	Description
15:00	CSR Data	Writing data into RDP writes the data into the CSR selected in RAP. Reading the data from the RDP reads the data from the CSR selected in RAP. CSR_1, CSR_2 and CSR_3 are accessible only when the STOP bit of CSR_0 is set. If the STOP bit is not set while attempting to access CSR_1, CSR_2 or CSR_3, the chip will return READY, but a READ operation will return undefined data. WRITE operation is ignored.

1.9.2 CONTROL AND STATUS REGISTERS

CONTROL AND STATUS REGISTER 0 (CSR_0)



AF000860

The LANCE updates CSR_0 by logical "OR"ing the previous and present value of CSR_0. CSR_0 is accessible when RAP = 00.

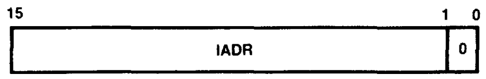
Bit	Name	Description
15	ERR	ERROR summary is set by the "OR" of BABL, CERR, MISS and MERR. ERR remains set as long as any of the error flags are true. ERR is read only; writing to it has no effect. It is cleared by Bus $\overline{\text{RESET}}$, by setting the STOP bit, or clearing the individual error flags.
14	BABL	BABBLE is a transmitter timeout error. It indicates that the transmitter has been on the channel longer than the time required to send the maximum length packet. BABL is a flag which indicates excessive length in the transmit buffer. It is set after 1519 data bytes have been transmitted; the chip continues to transmit until the whole packet is transmitted or there is a failure. When BABL error occurs, an interrupt is generated if INEA = 1. BABL is READ/CLEAR ONLY. It is set by the LANCE and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
13	CERR	COLLISION ERROR is a collision after transmission transceiver test feature. It indicates that the collision input to the LANCE failed to activate within 2us after a LANCE-initiated transmission was completed. This function is also known as heartbeat. CERR is READ/CLEAR ONLY. It is set by the chip and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.

Bit	Name	Description
12	MISS	MISSED PACKET is set when the receiver loses a packet because it does not own a receive buffer and the silo has overflowed, indicating loss of data. Silo overflow is not reported because there is no receive ring entry in which to write status. When MISS is set, an interrupt is generated if INEA = 1. MISS is READ/CLEAR ONLY. It is set by the LANCE and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
11	MERR	MEMORY ERROR is set when the LANCE is the Bus Master and has not received $\overline{\text{READY}}$ within 25.6us after asserting the address on the DAL lines. When a Memory Error is detected, the receiver and transmitter are turned off and an interrupt is generated if INEA = 1. MERR is READ/CLEAR ONLY. It is set by the LANCE and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
10	RINT	RECEIVER INTERRUPT is set when the LANCE updates an entry in the Receive Descriptor Ring for the last buffer received before fall of carrier. When RINT is set an interrupt is generated if INEA = 1. RINT is READ/CLEAR ONLY. It is set by the LANCE and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
09	TINT	TRANSMITTER INTERRUPT is set when the LANCE updates an entry in the transmit descriptor ring for the last buffer sent or transmission is stopped due to a failure. When TINT is set, an interrupt is generated if INEA = 1. TINT is READ/CLEAR ONLY and is set by the LANCE and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
08	IDON	INITIALIZATION DONE indicates that the chip has completed the initialization procedure started by setting the INIT bit. The LANCE has read the Initialization Block from memory and stored the new parameters. When IDON is set, an interrupt is generated if INEA = 1. IDON is READ/CLEAR ONLY. It is set by the LANCE and cleared by writing a "1" into the bit. Writing a "0" has no effect. It is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
07	INTR	INTERRUPT FLAG is set by the "OR" of BABL, MISS, MERR, RINT, TINT and IDON. If INEA = 1, the $\overline{\text{INTR}}$ pin will be low. INTR is READ ONLY; writing this bit has no effect. INTR is cleared by RESET, by setting the STOP bit, or by clearing the condition causing the interrupt.
06	INEA	INTERRUPT ENABLE allows the INTR I/O pin to be driven low when the Interrupt Flag is set. If INEA = 1 and INTR = 1, the INTR I/O pin will be high, regardless of the state of the Interrupt Flag. INEA is READ/WRITE and cleared by RESET or by setting the STOP bit. INEA cannot be set while STOP bit is set. INEA can be set in parallel or after INIT and/or STRT bit is set.
05	RXON	RECEIVER ON indicates that the receiver is enabled. RXON is set when STRT is set if DRX = 0 in the MODE register in the Initialization Block and the Initialization Block has been read by the LANCE by setting the INIT bit. RXON is cleared when IDON is set from setting the INIT bit and DRX = 1 in the MODE register, or a memory error (MERR) has occurred. RXON is READ ONLY; writing this bit has no effect. RXON is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
04	TXON	TRANSMITTER ON indicates that the transmitter is enabled. TXON is set when STRT is set if DTX = 0 in the MODE register in the Initialization Block and the INIT bit has been set. TXON is cleared when IDON is set and DTX = 1 in the MODE register or an error, such as MERR, UFLO or BUFF, has occurred during transmission. TXON is READ ONLY; writing this bit has no effect. TXON is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
03	TDMD	TRANSMIT DEMAND - when set - causes the LANCE to access the Transmit Descriptor Ring without waiting for the polling time interval to elapse. TDMD need not be set to transmit a packet; it merely hastens the chip's response to a Transmit Descriptor Ring entry insertion by the host. TDMD is WRITE WITH ONE ONLY and is cleared by the microcode after it is used. It may read as a "1" for a short time after it is written because the microcode may have been busy when TDMD was set. It is also cleared by $\overline{\text{RESET}}$ or by setting the STOP bit. Writing a "0" in this bit has no effect.

Bit	Name	Description
02	STOP	STOP disables the chip from all external activity when set and clears the internal logic. Setting STOP is the equivalent of asserting $\overline{\text{RESET}}$. The LANCE remains inactive and STOP remains set until the STRT or INIT bit is set. If STRT, INIT and STOP are all set together, STOP overrides the other bits and only STOP will be set. STOP is READ/WRITE WITH ONE ONLY and set by RESET. Writing a "0" to this bit has no effect. STOP is cleared by setting either INIT or STRT.
01	STRT	START enables the LANCE to send and receive packets, perform direct memory access, and do buffer management. STOP bit must be set prior to setting the STRT bit. Setting STRT clears the STOP bit. INIT and STRT must not be set in the same cycle. Set INIT first and wait for IDON (IDON=1) before setting STRT bit. INIT is READ/WRITE WITH ONE ONLY. Writing a "0" into this bit has no effect. STRT is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.
00	INIT	INITIALIZE, when set, causes the chip to begin the initialization procedure and access the Initialization Block. STOP bit must be set prior to setting the INIT bit. Setting INIT clears the STOP bit. INIT and STRT must not be set in the same cycle. Set INIT first and wait for IDON (IDON=1) before setting STRT bit. INIT is READ/WRITE WITH "1" ONLY. Writing a "0" into this bit has no effect. INIT is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit.

CONTROL AND STATUS REGISTER 1 (CSR_1)

READ/WRITE: Accessible only when the STOP bit of CSR_0 is a ONE and RAP = 01.

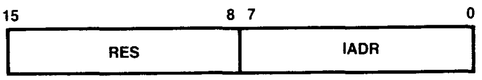


AF000970

Bit	Name	Description
15:01	IADR	The low order 15 bits of the address of the first word (lowest address) in the Initialization Block.
00		Must be zero.

CONTROL AND STATUS REGISTER 2 (CSR_2)

READ/WRITE: Accessible only when the STOP bit of CSR_0 is a ONE and RAP = 10.



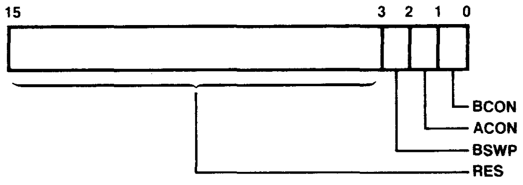
AF000920

Bit	Name	Description
15:08	RES	Reserved.
07:00	IADR	The high order 8 bits of the address of the first word (lowest address) in the Initialization Block.

CONTROL AND STATUS REGISTER 3 (CSR_3)

CSR_3 allows redefinition of the Bus Master interface.

READ/WRITE: Accessible only when the STOP bit of CSR_0 is ONE and RAP = 11. CSR_3 is cleared by $\overline{\text{RESET}}$ or by setting the STOP bit in CSR_0.



AF000900

Bit	Name	Description
15:03	RES	Reserved and read as "0".
02	BSWP	BYTE SWAP allows the LANCE to operate in systems that consider bits (15:08) of data to be at an even address and bits (07:00) to be at an odd address. When BSWP = 1, the LANCE will swap the high and low bytes on DMA data transfers between the silo and bus memory. Only data from silo transfers is swapped; the Initialization Block data and Descriptor Ring entries are NOT swapped. BSWP is READ/WRITE and cleared by $\overline{\text{RESET}}$ or by setting the STOP bit in CSR_0.

Bit	Name	Description
01	ACON	ALE CONTROL defines the assertive state of ALE when the LANCE is a Bus Master. ACON is READ/WRITE and cleared by $\overline{\text{RESET}}$ and by setting the STOP bit in CSR_0.

ACON	ALE
0	Asserted High
1	Asserted Low

Bit	Name	Description
00	BCON	BYTE CONTROL redefines the Byte Mask and Hold I/O pins. BCON is READ/WRITE and cleared by $\overline{\text{RESET}}$ or by setting the STOP bit in CSR_0.

BCON	Pin 15	Pin 16	Pin 17
0	$\overline{BM_0}$	$\overline{BM_1}$	$\overline{\text{HOLD}}$
1	BYTE	$\overline{\text{BUSAKO}}$	$\overline{\text{BUSRQ}}$

All data transfers from the LANCE in the Bus Master mode are in words. However, the LANCE can handle odd address boundaries and/or packets with an odd number of bytes.

1.9.3 INITIALIZATION BLOCK

The Initialization Block consists of five registers; Mode Register, Physical Address Register, Logical Address Filter, Receive Descriptor Ring Pointer, and Transmit Descriptor Ring Pointer. Chip initialization includes the reading of the Initialization Block in memory to obtain the operating parameters.

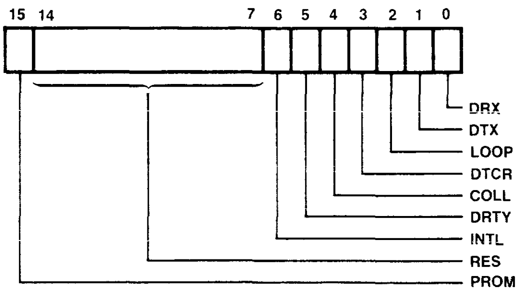
The Initialization Block is read by the chip when the INIT bit in CSR_0 is set. The INIT bit should be set before or concurrent with the STRT bit to insure proper parameter initialization and chip operation. After the chip has read the Initialization Block, IDON is set in CSR_0 and an interrupt is generated if INEA = 1.

The following is a definition of the Initialization Block.

Highest Address	TLEN-TDRA (31:16)	IADR + 22
	TDRA (15:00)	IADR + 20
	RLEN-RDRA (31:16)	IADR + 18
	RDRA (15:00)	IADR + 16
	LADRF (63:48)	IADR + 14
	LADRF (47:32)	IADR + 12
	LADRF (31:16)	IADR + 10
	LADRF (15:00)	IADR + 08
	PADR (47:32)	IADR + 06
	PADR (31:16)	IADR + 04
	PADR (15:00)	IADR + 02
Base Address	MODE	IADR + 00

MODE REGISTER

The Mode Register allows alteration of the chip's operating parameters. Normal operation is with the Mode Register clear.



AF000510

Bit	Name	Description
15	PROM	PROMISCUOUS mode. When PROM = 1, all incoming packets are accepted.
14:07	RES	RESERVED.
06	INTL	INTERNAL LOOPBACK is enabled when the LOOP bit is set. Internal loopback lets the LANCE receive its own transmitted packet. Since this represents full duplex operation, the packet size is 8–32 bytes. INTERNAL LOOPBACK operates in promiscuous mode and when the packets are addressed to the node itself.

If the source and destination are different, the LANCE must be re-initialized after each test.

The LANCE will not receive external packets in INTERNAL LOOPBACK mode.

EXTERNAL LOOPBACK allows the LANCE to transmit a packet through the SIA transceiver cable out to the Ethernet coax. It is used to determine the operability of all circuitry and connections between the LANCE and the coaxial cable. Multicast addressing in EXTERNAL LOOPBACK is valid only when DTCR=1 (user needs to append the 4-byte CRC).

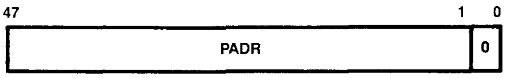
In EXTERNAL LOOPBACK, the Silo READ/WRITE pointers can get misaligned under heavy traffic. The packet may get corrupted, or not be received. Therefore, EXTERNAL LOOPBACK should be executed several times in order to isolate this type of error in receiving the lost packet.

INTL is only valid if LOOP = 1; otherwise it is ignored.

LOOP	INTL	LOOPBACK
0	X	No loopback, normal
1	0	External
1	1	Internal

Bit	Name	Description
05	DRTY	DISABLE RETRY. When DRTY = 1, the LANCE will attempt only one transmission of a packet. If there is a collision on the first transmission attempt, a Retry Error (RTRY) will be reported in Transmit Message Descriptor 3 (TMD_3).
04	COLL	FORCE COLLISION. This bit allows the collision logic to be tested. The LANCE must be in internal loopback mode of COLL to be valid. If COLL = 1, a collision will be forced during the subsequent transmission attempt. This will result in 16 total transmission attempts with a retry error reported in TMD_3.
03	DTCR	DISABLE TRANSMIT CRC. When DTCR = 0, the transmitter generates and appends a CRC to the transmitted packet. When DTCR = 1, the CRC logic is allocated to the receiver and no CRC is generated or sent with the transmitted packet. During loopback, DTCR = 0 will cause a CRC to be generated on the transmitted packet, but no CRC check is done by the receiver since the CRC logic is shared and cannot generate and check CRC at the same time. The generated CRC is written into memory with the data and can be checked by the host software. If DTCR = 1 during loopback, the host software must append a CRC value to the transmit data. The receiver checks the CRC on the received data and reports any errors. Since the CRC generator is used to generate the hash filter, the multicast addressing cannot be used when DTCR = 1.
02	LOOP	LOOPBACK allows the LANCE to operate in full duplex mode for test purposes. The packet size is 8-32 bytes. The received packet can be up to 36 bytes (32 + 4 bytes CRC) when DTCR=0. During loopback, the runt packet filter is disabled because the packet is smaller than the minimum size Ethernet packet (64 bytes). LOOP = 1 allows simultaneous transmission and reception for a message constrained to fit within the Silo. The chip waits until the entire message is in the Silo before serial transmission begins. The incoming data stream fills the Silo from behind as it is being emptied. Moving the received message out of the Silo to memory does not begin until reception has ceased. In loopback mode, transmit data chaining is not possible. Receive data chaining is possible if receive buffers are 32 bytes long to allow time for lookahead.
01	DTX	DISABLE THE TRANSMITTER causes the LANCE to not access the Transmitter Descriptor Ring and therefore no transmissions are attempted. DTX = 1 clears the TXON bit in CSR_0 when initialization is complete.
00	DRX	DISABLE THE RECEIVER causes the chip to reject all incoming packets and not access the Receive Descriptor Ring. DRX = 1 clears the RXON bit in the CSR_0 when initialization is complete.

PHYSICAL ADDRESS REGISTER (PADR)



AF000520

Bit	Name	Description
47:00	PADR	PHYSICAL ADDRESS is the unique 48-bit physical address assigned to the LANCE. PADR (0) must be zero.

LOGICAL ADDRESS FILTER (LADRF)

AF000500

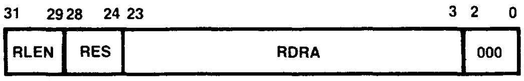
Bit	Name	Description
63:00	LADRF	The 64-bit mask used by the LANCE to accept logical addresses.

If the first bit of an incoming address is a "1" [PADR (0) = 1], the address is deemed logical and is passed through the logical address filter.

The logical address filter is a 64-bit mask that is used to accept incoming Logical Addresses. The incoming address is sent through the CRC circuit. After all 48 bits of the address have gone through the CRC circuit, the high order 6 bits of the resultant CRC are strobed into a register. This register is used to select one of the 64-bit positins in the Logical Address Filter. If the selected filter bit is a "1" the address is accepted and the packet is put in memory. The logical address filter only assures that there is a possibility that the incoming logical address belongs to the node. To determine if it belongs to the node, the incoming logical address that is stored in main memory is compared by software to the physical address that was loaded through the Initialization Block.

The Broadcast address, which is all Ones, does not go through the Logical Address Filter and is always enabled. If the Logical Address Filter is loaded with all zeroes, all incoming logical addresses except broadcast will be rejected.

RECEIVE DESCRIPTOR RING POINTER

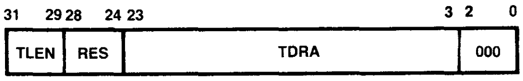


AF000490

Bit	Name	Description
31:29	RLEN	RECEIVE RING LENGTH is the number of entries in the receive ring expressed as a power of two.
28:24	RES	RESERVED
23:03	RDRA	RECEIVE DESCRIPTOR RING ADDRESS is the base address (lowest address) of the Receive Descriptor Ring.
02:00		MUST BE ZEROES. These bits are RDRA (02:00) and must be zeroes because the Receive Rings are aligned on quadword boundaries.

RLEN	Number of Entries
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

TRANSMIT DESCRIPTOR RING POINTER



AF000480

Bit	Name	Description
31:29	TLEN	TRANSMIT RING LENGTH is the number of entries in the Transmit Ring expressed as a power of two.
28:24	RES	RESERVED
23:03	TDRA	TRANSMIT DESCRIPTOR RING ADDRESS is the base address (lowest address) of the Transmit Descriptor Ring.
02:00		MUST BE ZEROES. These bits are TDRA (02:00) and must be zeroes because the Transmit Rings are aligned on quadword boundaries.

TLEN	Number of Entries
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

1.9.4 DESCRIPTOR RINGS

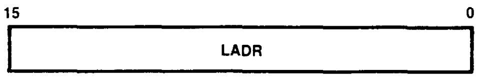
Buffer Management is accomplished through message descriptors organized in ring structures in memory. Each message descriptor entry is four words long. There are two rings allocated for the device: a receive ring and a transmit ring. The device is capable of polling each ring for buffers to either empty or fill with packets to or from the channel. The device is also capable of entering status information in the descriptor entry. Chip polling is limited to looking at the descriptor entry following the current entry.

The location of the descriptor rings and their length are found in the Initialization Block, accessed during the initialization procedure by the chip. Writing a "ONE" into the STRT bit of CSR_0 will cause the chip to start accessing the descriptor rings and enable it to send and receive packets.

The LANCE communicates with a HOST device through the ring structures in memory. Each entry in the ring is either "owned" by the chip or the HOST. There is an ownership bit (OWN) in the message descriptor entry. Mutual exclusion is accomplished by a protocol which states that each device can only relinquish ownership of the descriptor entry to the other device; it can never take ownership, and no device can change the state of any field in any entry after it has relinquished ownership.

Each descriptor in a ring in memory is a 4-word entry. The following is the format of the receive and the transmit descriptors.

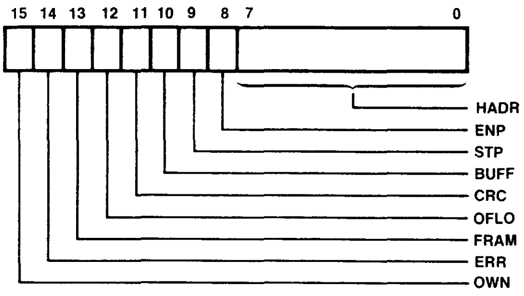
RECEIVE MESSAGE DESCRIPTOR 0 (RMD_0)



AF000940

Bit	Name	Description
15:00	LADR	The LOW ORDER 16 address bits of the buffer pointed to by this descriptor. LADR is written by the host and unchanged by the LANCE.

RECEIVE MESSAGE DESCRIPTOR 1 (RMD_1)

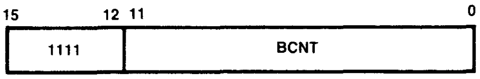


AF000870

Bit	Name	Description
15	OWN	This bit indicates that the descriptor entry is owned by the host (OWN = 0) or by the chip (OWN = 1). The LANCE clears the OWN bit after filling the buffer pointed to by the descriptor entry. The host sets the OWN bit after emptying the buffer. Once the chip or host has relinquished ownership of a buffer, it must not change any field in the four words that comprise the descriptor entry.
14	ERR	ERROR summary is the "OR" of FRAM, OFLO, CRC or BUFF.
13	FRAM	FRAMING ERROR indicates that the incoming packet contains a non-integer multiple of eight bits and there is a CRC error. If the incoming packet has no CRC error, then FRAM will not be set even if the packet has a non-integer multiple of eight bits. FRAM is not valid in internal loopback mode. See Note 1.
12	OFLO	OVERFLOW error indicates that the Silo overflowed and all or part of the incoming packet was lost because no memory buffer was available. See Note 1.
11	CRC	CRC indicates that the receiver has detected a CRC error on the incoming packet. See Note 1.
10	BUFF	BUFFER ERROR is set any time the chip does not wn the next buffer while data chaining a received packet. This can occur in either of two ways: 1) the OWN bit of the next buffer is zero, or 2) silo overflow occurred before the chip received the next STATUS. If a Buffer Error occurs, an Overflow Error may also occur internally in the Silo, but is not reported in the descriptor status entry unless both BUFF and OFLO errors occur at the same time.
09	STP	START OF PACKET indicates that this is the first buffer of a packet. It is used for data chaining buffers.
08	ENP	END OF PACKET indicates that this is the last buffer of this packet. It is used for data chaining buffers. If both STP and ENP are set, the packet fits into one buffer and there is no data chaining. See Note 1.
07:00	HADR	The HIGH ORDER 8 address bits of the buffer pointed to by this descriptor. This field is written by the host and unchanged by the LANCE.

Note 1. Disregard the error status in RMD_1 when both OVFL and ENP are set (no overflow error).
ONFL is valid only when the ENP is "0".
CRC and FRAM are valid only when ENP is "1".

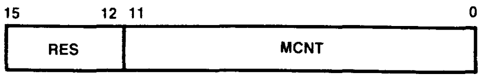
RECEIVE MESSAGE DESCRIPTOR 2 (RMD_2)



AF000930

Bit	Name	Description
15:12		MUST BE ONES. This field (all Ones) is written by the Host and unchanged by the LANCE.
11:00	BCNT	BUFFER BYTE COUNT is the length of the buffer pointed to by this descriptor, expressed as a two's complement number. This field is written by the host and unchanged by the LANCE. Minimum buffer size is 64 bytes for the first buffer packet.

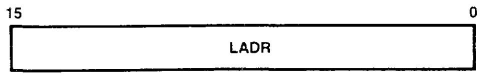
RECEIVE MESSAGE DESCRIPTOR 3 (RMD_3)



AF000950

Bit	Name	Description
15:12	RES	RESERVED and read as zeroes.
11:00	MCNT	MESSAGE BYTE COUNT is the length in bytes of the received message. MCNT is valid only when ERR is clear and ENP is set.

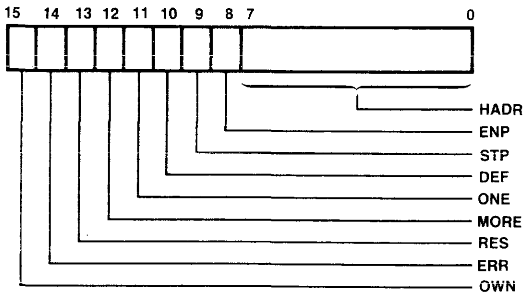
TRANSMIT MESSAGE DESCRIPTOR 0 (TMD_0)



AF000940

Bit	Name	Description
15:00	LADR	The LOW ORDER 16 address bits of the buffer pointed to by this descriptor. LADR is written by the host and unchanged by the LANCE.

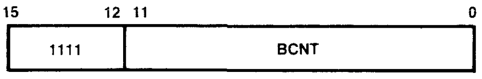
TRANSMIT MESSAGE DESCRIPTOR 1 (TMD_1)



AF000880

Bit	Name	Description
15	OWN	This bit indicates that the descriptor entry is owned by the host (OWN = 0) or by the LANCE (OWN = 1). The host sets the OWN bit after filling the buffer pointed to by this descriptor. The chip clears the OWN bit after transmitting the contents of the buffer. Neither the host nor the chip must alter a descriptor entry after it has relinquished ownership.
14	ERR	ERROR summary is the "OR" of LCOL, LCAR, UFLO or RTRY.
13	RES	RESERVED bit. The LANCE will write this bit with a "0".
12	MORE	MORE indicates that more than one retry was needed to transmit a packet.
11	ONE	ONE indicates that exactly one retry was needed to transmit a packet. One flag is not valid when LCOL is set.
10	DEF	DEFERRED indicates that the chip had to defer while trying to transmit a packet. This condition occurs if the channel is busy when the chip is ready to transmit.
09	STP	START OF PACKET indicates that this is the first buffer of this packet. It is used for data chaining buffers. STP is set by the host and unchanged by the chip. The STP bit must be set in the first buffer of the packet or the LANCE will skip over this descriptor, poll the next descriptor(s) entry until the OWN and STP bits are set.
08	ENP	END OF PACKET indicates that this is the last buffer to be used by the chip for this packet. It is used for data chaining buffers. If both STP and ENP are set, the packet fits into one buffer and there is no data chaining. ENP is set by the host and unchanged by the LANCE.
07:00	HADR	The HIGH ORDER 8 address bits of the buffer pointed to by this descriptor. This field is written by the host and unchanged by the LANCE.

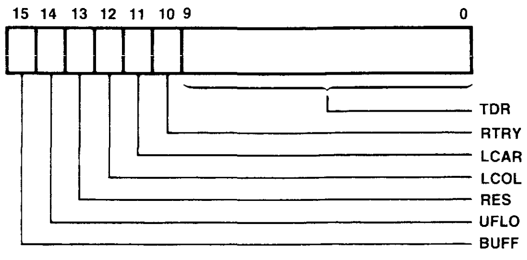
TRANSMIT MESSAGE DESCRIPTOR 2 (TMD_2)



AF000980

Bit	Name	Description
15:12		Must be ones. This field is set by the host and unchanged by the LANCE.
11:00	BCNT	BUFFER BYTE COUNT is the usable length in bytes of the buffer pointed to by this descriptor expressed as a two's complement number. This is the number of bytes from this buffer that will be transmitted by the chip. This field is written by the host and unchanged by the LANCE. The first buffer of a packet has to be at least 100 bytes minimum when data chaining and 64 bytes (DTCR=1) or 60 bytes (DTCR=0) when not data chaining.

TRANSMIT MESSAGE DESCRIPTOR 3 (TMD_3)



AF000890

Bit	Name	Description
15	BUFF	BUFFER ERROR is set by the LANCE during transmission when the chip does not find the ENP flag in the current buffer and does not own the next buffer. This can occur in either of two ways: either the OWN bit of the next buffer is zero, or Silo underflow occurred before the LANCE received the next STATUS signal. BUFF is set by the chip. BUFF error disables the transmitter (CSR_0 = TXON = 0) If a Buffer Error occurs, an Underflow Error will also occur. BUFF error is not valid when LCOL, or RTRY errors are set.
14	UFLO	UNDERFLOW ERROR indicates that the transmitter has truncated a message due to data late from memory. UFLO indicates that the Silo has emptied before the end of the packet was reached.
13	RES	RESERVED bit. The chip will write this bit with a "0".
12	LCOL	LATE COLLISION indicates that a collision has occurred after the slot time of the channel has elapsed. The LANCE does not retry on late collisions.
11	LCAR	LOSS OF CARRIER is set when the carrier input (RENA) to the LANCE goes false during a chip-initiated transmission. The chip does not retry upon loss of carrier. It will continue to transmit the whole packet LCAR until packet is finished. LCAR is not valid in INTERNAL LOOPBACK MODE.
10	RTRY	RETRY ERROR indicates that the transmitter has failed in 16 attempts to successfully transmit a message due to repeated collisions on the medium. If DRTY = 1 in the MODE register, RTRY will set after 1 failed transmission attempt.
09:00	TDR	TIME DOMAIN REFLECTOMETRY reflects the state of an internal LANCE counter that counts from the start of a transmission to the occurrence of a collision. This value is useful in determining the approximate distance to a cable fault. The TDR value is written by the chip and is valid only if RTRY is set.

CHAPTER 2
BUS INTERFACE CONSIDERATIONS

This chapter discusses the requirements of the LANCE in various configurations relating to the bus master or masters, bus bandwidth, and bus latency.

The LANCE may be required to fit into an already designed architecture or a new architecture may be designed around the LANCE. It is obvious that the latter case results in a cleaner design and more reliable system. The following analysis is being developed from a simple architecture where the LANCE and CPU are the only master devices in the system, to a more complex system with multiple master devices.

A system may be configured with:

The LANCE and one CPU as the only bus masters
The LANCE in systems with multiple other masters
The LANCE in a Daisy Chain operation
The LANCE with 8-bit microprocessors

Following a brief definition of bus bandwidth, bus latency, and LANCE DMA cycles, the LANCE bus latency and DMA operation are discussed in each of the above configurations. The bus bandwidth and latency are different for each of these configurations.

2.1 BUS BANDWIDTH DEFINITION

Bus bandwidth is defined as the percentage of the time that the bus is held and controlled by a device. Assuming that no wait states are added to the DMA cycles in a simplified system environment, the LANCE acquires the bus within a few cycles after requesting it. In order to handle the Ethernet transmission data rate of 10 Megabits per second, the LANCE must request the bus approximately every 12.8 us. The LANCE uses the bus for 4.8 us for eight words of DMA transfer. Therefore, 37.5% (4.8/12.8) of bus bandwidth is taken by the LANCE and the remainder is left to the CPU to perform other tasks. Any wait state added to the LANCE 8-word DMA cycle increases the LANCE DMA cycle by 800 ns and increases the bus bandwidth taken by the LANCE by 6.25 percentage points (see Figure 2-1).

2.2 BUS LATENCY DEFINITION

The term "bus latency" is defined as follows:

Bus latency is the amount of time from the time that the DMA device requests the bus (LANCE, HOLD=L) until the time it acquires the bus (LANCE, HLDA=L). Allowable Bus Latency is the amount of time that a DMA device can wait for the bus after requesting it without the loss of data or other functions.

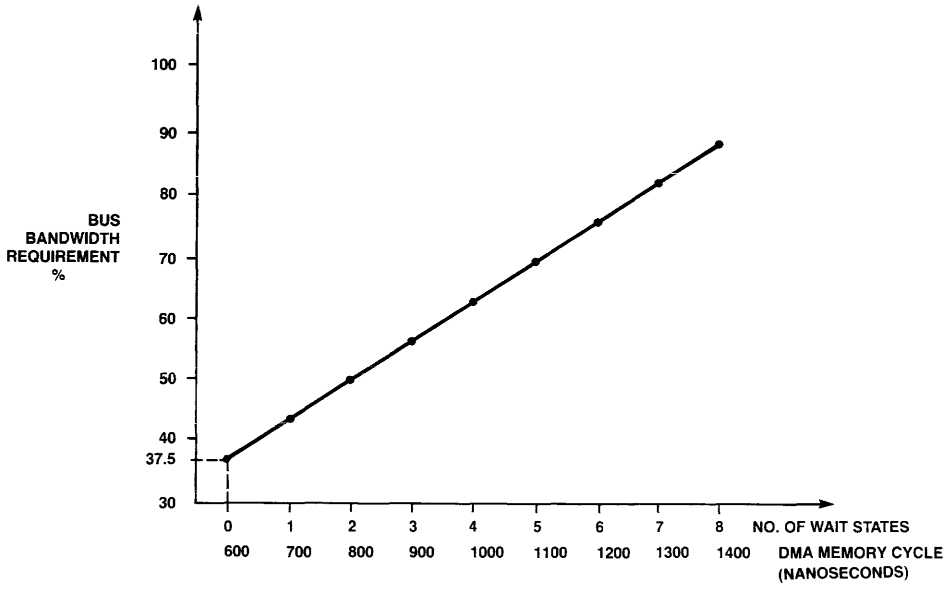


Figure 2-1 Bus Bandwidth Requirement

06363A-4

Bus latency is a limiting factor in assigning peripheral devices to a bus. Devices sensitive to the value of bus latency are: communication controllers, display memory scanners, and dynamic memories when a DMA approach is used.

Communication devices require that data be transferred to/from memory before an overflow/underflow condition occurs. The CRT controllers must scan the display memory to update and/or refresh the screen periodically. Each DMA device on a bus increases the bus latency when sharing the bus with other DMA devices. Therefore, the bus latency of each DMA device should be taken into account in designing a system.

2.3 LANCE DMA CYCLE (BUS MASTER MODE)

The LANCE is capable of two different types of DMA Transfers:

- Burst mode DMA
- Single word DMA

Burst mode DMA is used for transmission of packets from memory to the Silo (read) or for reception of packets from the Silo to memory (write).

The burst transfers are eight consecutive word reads (transmit) or writes (receive) that are done on a single bus arbitration cycle. In other words, after the LANCE receives the bus acknowledge, ($\overline{\text{HLDA}}$ = Low), it will do eight word transfers, or one byte and seven word transfers, (8 DMA cycles, at a minimum of 600 ns per cycle) before releasing the bus request signal by $\overline{\text{HOLD}}$ going high. If there are more than 8 words of data in the Silo in Receive Mode when the LANCE releases the bus ($\overline{\text{HOLD}}$ deasserted), the LANCE will request the bus again within 700 ns ($\overline{\text{HOLD}}$ dwell time). Burst DMAs are always eight cycle transfers unless there are less than eight words left to be transferred to/from the Silo.

The LANCE initiates single word DMA transfers to access the Transmit Rings, Receive Rings, or Initialization Block. The LANCE will not initiate any burst DMA transfer from the time that it gets to own the descriptor to the time it accesses the descriptor entries in the ring (an average of three to four separate DMA cycles for a multibuffer packet) or to the time it reads the Initialization Block.

2.4 LANCE AND ONE CPU ON THE BUS MASTER

In this configuration, the LANCE shares the bus with one master device. A dedicated node processor may be assigned to service the LANCE and the host CPU, or a single processor may be responsible for both the LANCE and the network tasks management. The bus bandwidth requirement should not be critical with only two master devices on the bus. The LANCE will request the bus when there are at least 16 bytes of data available in the Silo (receive mode) or when there are more than 16 locations available in the Silo (transmit mode). The CPU normally acknowledges the bus request within two to four cycles.

Once the LANCE acquires the bus, it will transfer eight words to/from the Silo before it relinquishes the bus. The transfer time is equivalent to about 4.8 us without any wait states. With 8.0 us Dwell time, the LANCE requests the bus every 12.8 us (8.0 + 4.8) until the whole packet is transmitted or received. The ring access is done with a single DMA arbitration cycle, either between the time that the LANCE is transferring data to/from the Silo (data chaining) or when the last buffer of a packet (receive or transmit) has been relinquished to the LANCE. When the LANCE owns the buffer, it does three to four separate DMA cycles, with 1.0 to 2.0 us dwell time for every buffer. The LANCE does not perform any DMA burst between ring access time.

2.4.1 BUS LATENCY WITH ONE OTHER MASTER

The above discussion assumes that the LANCE receives the bus acknowledge ($\overline{\text{HLDA}}$) fast enough (within a few cycles). The large Silo size (48 bytes) allows the LANCE to tolerate a longer bus latency. When the LANCE is in receive mode and there are eight words in the Silo, the time required for the LANCE to receive the bus acknowledge can be stretched to nearly 25.6 us (16 words) before the Silo overflows. However, the very next bus request ($\overline{\text{HOLD}}$) from the LANCE should be acknowledged within 8.0 us because there are already about 19 words in the Silo. The LANCE always releases the bus after eight word transfers, even if there are already eight or more words residing in the Silo. The second bus request will be asserted within 700 ns, when there are eight or more words in the Silo (dwell time).

2.5 LANCE WITH MULTIPLE MASTER DEVICES ON THE BUS

The LANCE is flexible when it is integrated with other master devices on the bus. A large FIFO (also referred to as Silo), 48 bytes deep, and a unique programmable interface make LANCE versatile in a multimaster environment.

The impact of integrating a master device onto an existing bus system requires careful investigation at the start of system development. The following discussion summarizes the areas of concern to the user when designing the LANCE into a multimaster bus.

In designing a bus with multiple master devices, some restrictions are imposed on the master devices. These requirements allow the bus to be utilized efficiently, and each device gets a fair share of the bus without degrading the bus bandwidth.

Some typical system bus requirements are:

1. Each device on the bus is allowed to hold the bus for only a limited time, once it has acquired it. There are many system requirements which must be considered to determine the maximum time any bus master is allowed to remain on the bus. If this time is exceeded, some type of monitoring device generates a bus error and forces the device off the bus.

The LANCE holds the bus for a minimum of 4.8 us during the DMA burst and a minimum of 600 ns when accessing the rings or the Initialization Block. If the memory access time is longer than the LANCE DMA cycle, the LANCE DMA cycle period can be stretched (wait states being added to the LANCE DMA cycle) by the use of some external logic and by controlling the $\overline{\text{READY}}$ input to the LANCE. Eventually, the $\overline{\text{READY}}$ should be granted to the LANCE before the occurrence of either an overflow or underflow error condition, or a 25.6 usec memory timeout error (MERR).

2. Any device on the bus may be kicked off the bus (preemptive DMA) at any time. This is usually enforced by a bus arbiter when a higher priority device needs to acquire the bus. Refer to Figure 2-2.

The $\overline{HLDA}$ to the LANCE cannot be deasserted while the $\overline{HOLD}$ from the LANCE is active. Once the LANCE has acquired the bus, it will not release the bus until it has finished its 8 DMA cycles (or less, if it happens to be the last few bytes of the packet) while transferring data to/from the Silo. However, the external logic pointed out previously may be used to add wait states to the LANCE DMA cycle and to isolate its address/data bus, through buffers or transceivers, from the system bus. Thus preemptive DMA can be supported by the use of external logic.

3. Within an existing bus, the master devices already on the bus may not have enough bus bandwidth left to allow for any new device to be added. The new device may be required to stay off the bus and provide its own dedicated dual port memory.

The LANCE supports a dual port RAM architecture with some external logic. The external logic is needed to allow access to the LANCE internal register, CSR_0, during the normal operation. This register contains the interrupt/control status register bits. The external logic must monitor and control $\overline{HOLD}$, $\overline{HLDA}$ of the LANCE so that the CPU can access the CSR_0 when the LANCE is not in the middle of a series of DMA accesses.

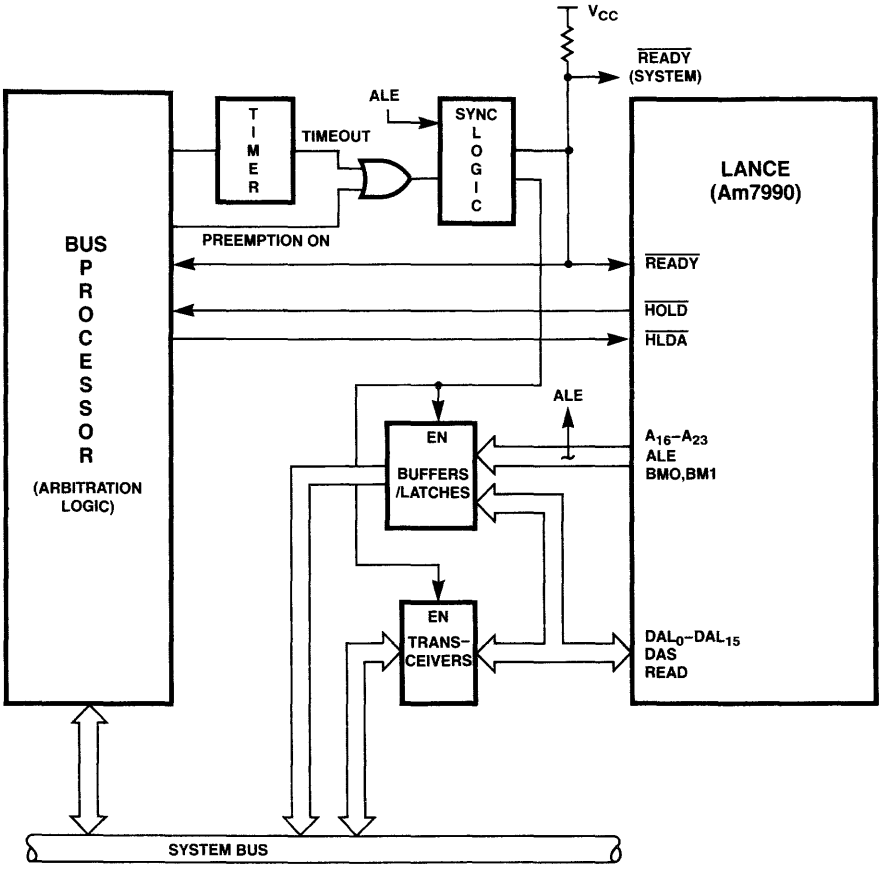


Figure 2-2 Block Diagram of Pre-emption in the LANCE 06363A-5

2.5.1 BUS LATENCY FOR MULTIMASTER ENVIRONMENT

The amount of allowable bus latency for a master device must be evaluated prior to adding a new device into a multimaster environment. We will study the allowable bus latency for the LANCE under two different cases: a single buffer per packet (no data-chaining) and multiple buffer segments per packet (data-chaining). The main difference is that, in data-chaining, the LANCE does a separate lookahead operation (requiring three DMA accesses for each buffer segment) to access the ring between the periods it is doing DMA burst. Therefore, this additional time overhead reduces the allowable bus latency. This is especially significant when short buffer sizes are assigned and long packets are in process.

NO DATA-CHAINING

When the buffers are not data-chained, the single DMA access to the ring per packet may be disregarded especially for large packets. In the receive mode, words enter the Silo from the Ethernet at the rate of eight words every 12.8 microseconds. The LANCE transfers words from the Silo is 8-word bursts requiring 4.8 microseconds per burst.

Therefore the allowable bus latency for LANCE is 12.8 minus 4.8 or 8 microseconds disregarding the single DMA access to the descriptor ring per packet.

The size of the Silo provides some protection against a one time unusual delay in granting the bus after a LANCE bus request. The LANCE automatically requests the DMA bus as soon as eight words are in the Silo. The Silo may accumulate an additional sixteen words (32 bytes) on a one time basis before the LANCE becomes the bus master. This means that the bus latency is 25.6 us (32 x .8) for the first time that the LANCE requests the bus. However, subsequent bus requests must be granted within 8 us without fail to prevent a Silo overflow. The 25.6 us bus latency can apply for one bus access at any time that the Silo has eight or fewer words in it.

The eight microsecond bus latency corresponds to an accumulation of five words in the Silo. The bus latency may be stretched by some amount when the packets are shorter.

WITH DATA-CHAINING

When the buffers are data-chained, the overhead caused by the lookahead operation in accessing the ring must be taken into consideration in calculating the bus latency. This lookahead is performed between the periods in which the LANCE is doing DMA bursts. It consists of three separate single DMA reads when the LANCE owns the buffer. The LANCE does not initiate any DMA burst (8 DMA cycles) between the three single DMA reads.

The lookahead operation may take as long as 5.8 microseconds plus three times the bus latency of a single DMA read. This is calculated as follows: The dwell time (the time interval between the LANCE's request to the bus to access the ring and the acquisition of the bus) is approximately 1.0 to 2.0 us for each access. Two ring accesses are required. Three ring access transfers at .6 microseconds each are required.

The following calculation leads to the worst case allowable bus latency for the LANCE, for large packet (1518 bytes) and small buffer sizes (64 bytes), when the buffers are data-chained.

Assume:

— No wait state is added to the LANCE DMA cycle.

— $\overline{\text{HLDA}}$ assertion to bus driver enable delay is not included in the calculation.

64-byte transfer (1 buffer) time =
4 x 4.8 = 19.2 us (4 bus requests)

Ring access transfer (3 single DMA) =
3 x 0.6 = 1.8 us (3 bus requests)

Ring access dwell time =
2 x 2.0 = 4.0 us

64-byte transfer time from/to Ethernet cable to/from Silo = 64 x 0.8 = 51.2 us

$$\text{ALLOWABLE BUS LATENCY} = \frac{51.2 - (19.2 + 1.8 + 4.0)}{7} = 3.74 \text{ us}$$

Because of the lookahead overhead, the data-chained buffer mode has less allowable bus latency than the single buffer mode which does not require data-chaining.

2.6 LANCE IN A DAISY-CHAIN CONFIGURATION

It may be desirable to integrate the LANCE into a daisy-chained environment (for example, Z-BUS). The BCON bit in CSR_3 can be programmed for daisy-chain operation. As a result, Pins 15, 16, and 17 ($\overline{BM_0}$, BM_1, $\overline{HOLD}$) will function as BYTE, $\overline{BUSAKO}$, and $\overline{BUSRQ}$ respectively. In a daisy chain environment with pre-emption, the LANCE should be assigned to a high priority slot. This is necessary to meet the bus latency requirement under the worst case conditions.

In a daisy-chain environment without pre-emption (Zilog daisy-chain), it is possible for a device, independent of slot position, to wait for the remainder of the devices in the chain to finish their bus accesses before it can request the bus again. (This can occur when the other devices request the bus at the same time.) The impact to the LANCE is an underflow or overflow error condition if the LANCE bus latency requirement is not met. The addition of hardware, to provide preemptive DMA capability and/or to enforce a bus time limit to the devices on the bus, may not be worthwhile if the probability of this happening in a normal system environment is reasonably low.

2.7 LANCE WITH 8-BIT MICROPROCESSORS

Although the LANCE is a 16-bit device that interfaces easily to 16-bit microprocessors, it can also be interfaced to 8-bit microprocessors with some external logic.

In integrating the LANCE with 8-bit microprocessors, there are two possible configurations: 16-bit memory and 8-bit memory. Each solution has trade-offs in cost, performance, and complexity of implementation. Each approach is considered below.

2.7.1 8-BIT MICROPROCESSORS WITH 8-BIT MEMORY

At a glance, this seems to be a simple and not too costly implementation because it requires fewer memory chips. However, as we will see later, the implementation can be quite complex.

For this interface to work properly, it takes twice as long to do the data transfers as with 16-bit memories and a 16-bit microprocessor. The data transfers are between the LANCE and CPU (LANCE in Slave Mode), or the LANCE and 8-bit memories (LANCE in Master Mode). The interface to the LANCE should look like a 16-bit interface. The DMA word transfer cycle between the LANCE and the CPU or memory must be converted to two separate byte transfers. This type of architecture degrades the system performance by one-half.

For some lower cost applications, this may not be a bottleneck. However, it is important to note that the LANCE DMA transfer cycles take twice as long. Therefore, the potential of the LANCE getting into an overflow or underflow error condition is increased, especially if the LANCE bus requests are not acknowledged promptly.

Bus latency for a 16-bit microprocessor interface is based on a 4.8 us DMA cycle. The 8-bit microprocessors must perform twice as many DMA cycles as the 16-bit microprocessor, therefore, the bus latency for the 8-bit microprocessors is based on 9.6 us DMA cycles (assuming that an 8-bit microprocessor runs as fast as a 16-bit which is not a safe assumption). The allowable bus latency for an 8-bit microprocessor is therefore less than for a 16-bit microprocessor and the bus bandwidth is almost doubled. The LANCE must request the bus once every 12.8 us, and since it takes 9.6 us to do an 8-word transfer, the LANCE thus takes up 75% of the bus bandwidth and the allowable bus latency is 3.2 usec.

The bus latency and bandwidth limitations can be overcome by designing a dual-port RAM shared by the 8-bit processor and the LANCE. Some additional logic may be required to provide an 8-bit channel for use by the 8-bit processor, and a 16-bit channel to be used by the LANCE and any other 16-bit controllers.

2.7.2 8-BIT MICROPROCESSORS WITH 16-BIT MEMORY

Using a 16-bit memory is equivalent to having an 8-bit processor on a 16-bit bus. The memory is normally organized into lower and upper bytes. Although this approach is costly in terms of using twice as many memory chips, it is a better solution in terms of performance and ease of implementation.

This architecture is nearly equivalent to the LANCE being interfaced to a 16-bit processor. The LANCE DMA cycles, bus latency, and the bus bandwidth analysis are the same as those discussed involving 16-bit microprocessors. There is still a drawback in the area where the CPU has to access the LANCE. The CPU must go through two separate consecutive cycles for every word transferred to the LANCE; however, this is not significant since the LANCE to CPU interface does not require a fast response time. With enough receive buffers allocated to the LANCE, the packets are received independently of the CPU's interrupt response time.

CHAPTER 3
LANCE-CPU INTERFACE CONSIDERATIONS

3.1 GENERAL LANCE INTERFACE REQUIREMENTS

The following is a list of key points to be considered in designs using the LANCE:

- There are three open drain outputs on the LANCE. They are $\overline{\text{INTR}}$ (Pin 11), $\overline{\text{HOLD}}$ (Pin 17), and $\overline{\text{READY}}$ (Pin 22). These signals must be pulled up external to the LANCE chip by 3.3k or larger resistors.
- $\overline{\text{DAS}}$ and ALE are usually used as strobe signals. It would therefore be a good idea to pull them up to insure that a floating level will not affect other circuits.
- Decoupling capacitors with values of 0.1 and 5.0 uf are connected, in parallel, from the V_{CC} Pin of the LANCE to Ground.
- Voltage level on the $\overline{\text{RESET}}$ Pin of the LANCE should be maintained at above 3.4 volts with a pull-up resistor.

3.1.1 LANCE INTERFACE TO POPULAR MICROPROCESSORS

Several problems are associated with interfacing a peripheral device to a CPU. One major problem involves the various control signals that each chip uses. Unless the two families are designed to be pin-for-pin compatible, there are always going to be minor variations between them. Part of the pin incompatibility involves genuine signal differences while other pins only require name changes. Another problem with interfacing a general purpose peripheral to a CPU is the timing differences.

Most of these timing differences can be resolved by adding external logic. The Am7990 LANCE has been designed to be interfaced easily with the popular 16-bit microprocessors (8086/80186, 68000, Z8000*, LSI-II**). Most of the interface logic is embedded in the chip and is program selectable.

Although the LANCE itself has a multiplexed bus, it can easily be interfaced to demultiplexed buses with a minimal amount of effort. This chapter discusses the interfaces to four popular 16-bit microprocessors (68000, Z8001, 80186, and 8086). These designs assume that the processor and the LANCE reside on the same board. Address buffers and data transceivers are set up to be shared by the processor and the LANCE. All of these designs use PAL*** devices to reduce the parts count.

* Z8000 is a trademark of Zilog, Inc.
** LSI-II is a registered trademark of Digital Equipment Corp.
*** PAL is a registered trademark of Monolithic Memories, Inc.

3.2 68000 TO LANCE INTERFACE

The versatility of the LANCE makes it easier for the user to interface with demultiplexed buses. Figure 3-1 is an example of how to interface the 68000 to the LANCE. The goal of this interface is compatibility with 8-MHz and faster 68000s while minimizing parts count. The two flip-flops are needed to adapt the LANCE bus request handshake to the 68000.

Auto-vectoring is used since the Am7990 does not return a vector during Interrupt Acknowledge cycles. The BYTE and $\overline{\text{DAS}}$ signals of LANCE are used to generate $\overline{\text{UDS}}$ and $\overline{\text{LDS}}$ when LANCE is in Bus Master Mode. It takes two latches to demultiplex the LANCE Address/Data Lines (DAL_0-DAL_{15}) to adapt it to the 68000 Address Bus. The user should program the BSWP, BCON to "1", and ACON to "0" in CSR_3 when initializing the LANCE. The AmPAL22V10 device could also be used to eliminate the two flip-flops shown in Figure 3-1.

3.2.1 68000 TO LANCE INTERFACE PAL

```
PAL16L8                  JOE BRCICH
PAT002                   2 FEB 84
File: 68K90.PAL

68000 TO LANCE INTERFACE
ADVANCED MICRO DEVICES

/AS  RW  BYTE  /HOLD  NC  /BG  AO  NC  /BGACK  GND
/CS  /TB  /UDS  /DAS  /CLR1  BR  /CLR2  /LDS  /RB  VCC

IF (/BGACK) RB = CS*RW*UDS + CS*RW*LDS; 68000 Master

IF (/BGACK) TB = CS*/RW; 68000 Master

IF (BGACK) UDS = DAS*/AO*BYTE + /BYTE*DAS

IF (BGACK) LDS = DAS*AO*BYTE + /BYTE*DAS

IF (/BGACK) DAS = UDS*LDS

CLR1 = /AS*BG

CLR2 = BGACK

/BR = /HOLD
```

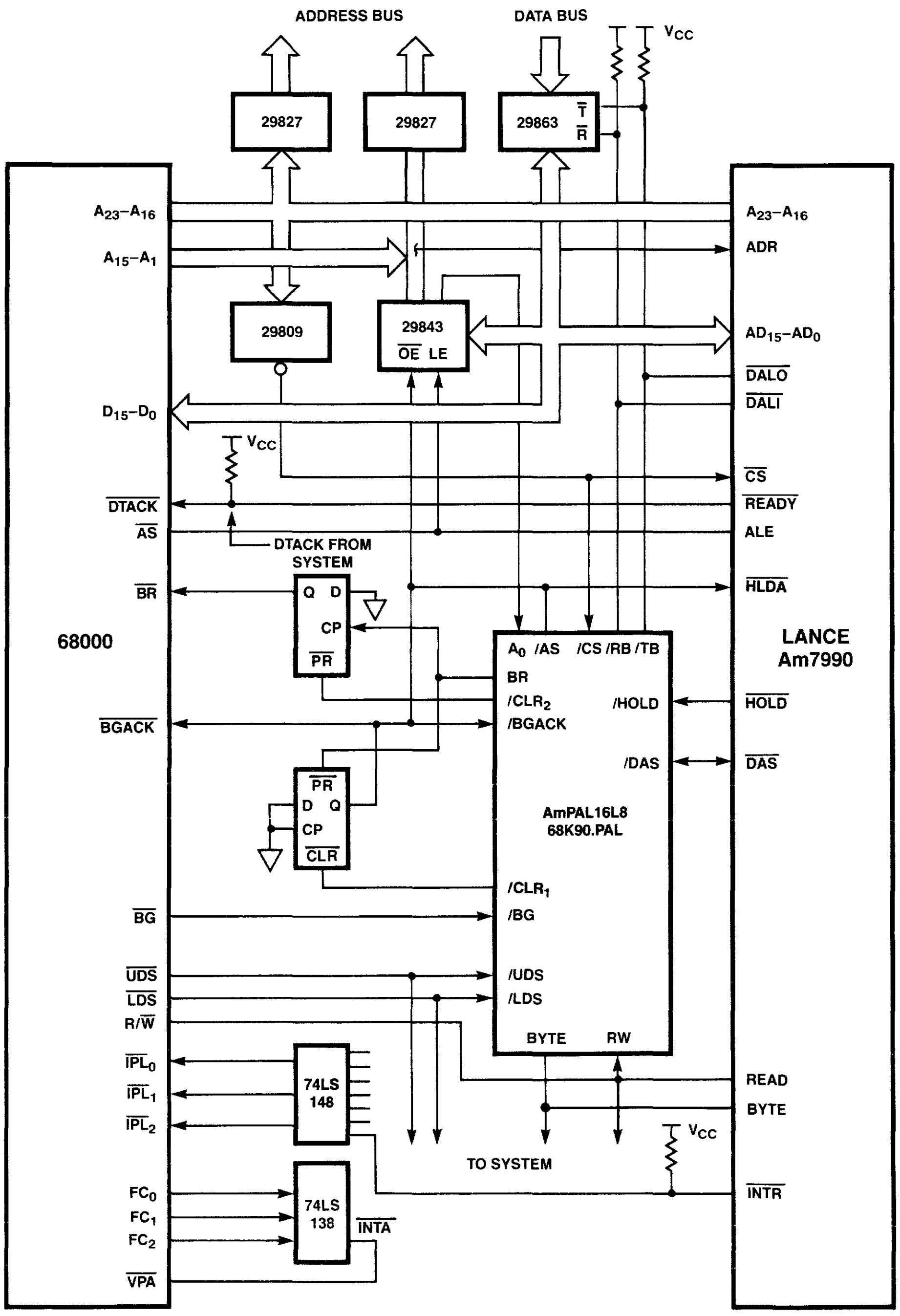


Figure 3-1 68000 to LANCE Interface

06363A-6

3.3 Z8001 TO LANCE INTERFACE

The Z8001 interface to the LANCE is easily accomplished since both have a multiplexed bus and most of the control signals can be directly connected (see Figure 3-2). This design also uses the PAL device, AmPAL16L8, to reduce the parts count. The $\overline{INTR}$ pin of the LANCE is connected to the $\overline{NVI}$ pin of Z8001, since LANCE does not return a vector during the Interrupt Acknowledge cycle. The PAL device uses the status lines ST_3-ST_0 (when Z8001 is the Bus Master) or $\overline{HLDA}$ from the LANCE (when LANCE is the Bus Master) to generate M, the memory request signal. The user should program the ACON, BCON, and BSWP to "1" prior to initializing the LANCE. When the LANCE is Bus Master, $\overline{DALI}$ and $\overline{DALO}$ control the transceiver. When the CPU is Bus Master, T and R are generated from R/$\overline{W}$ and $\overline{DS}$ to control the transceiver.

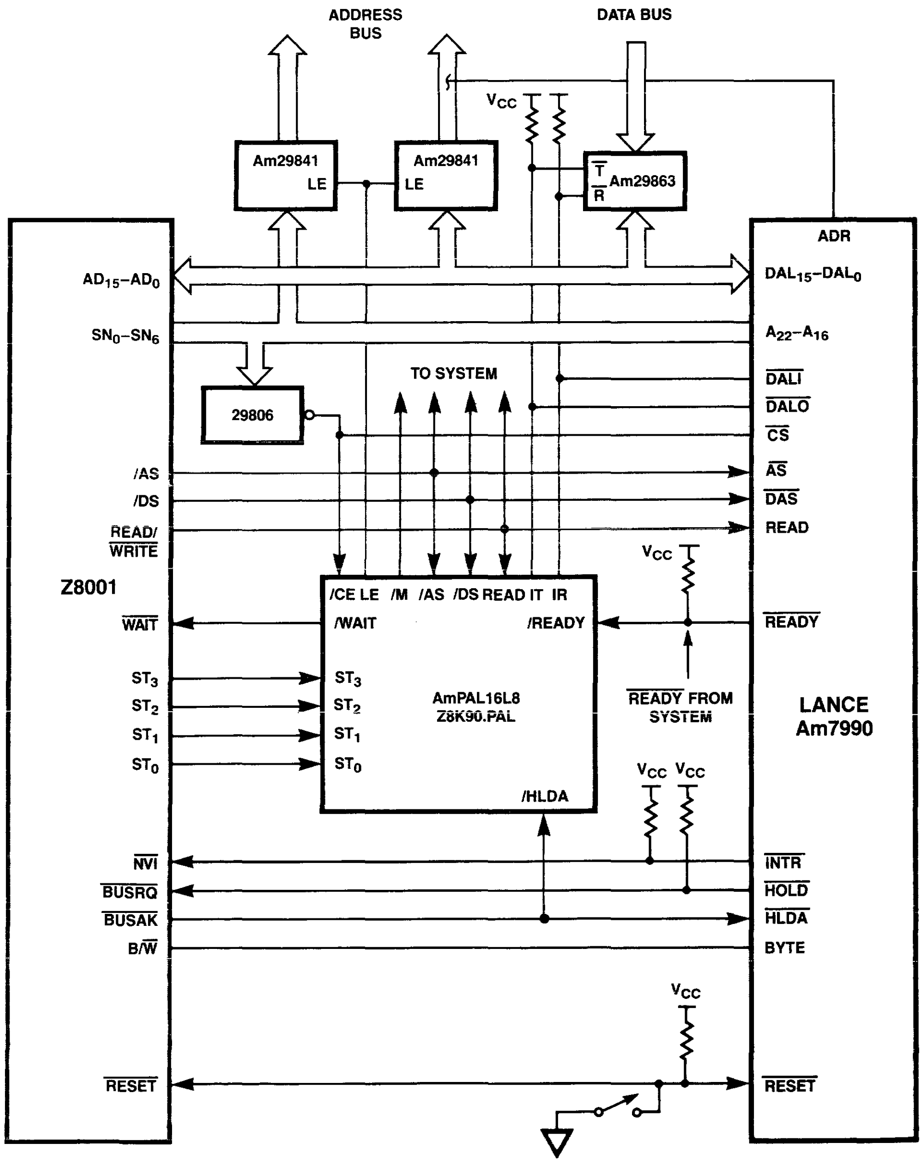


Figure 3-2 Z8001 to LANCE Interface 06363A-7

3.3.1 Z8001 TO LANCE INTERFACE PAL

```
AMPAL16L8                   RASOUL M. OSKOUY
PAT001                      MARCH 13, 1984
File: Z8K90.PAL

Z8001 TO LANCE INTERFACE
ADVANCED MICRO DEVICES

/AS  /HLDA READ /CS   /DS /READY ST3 ST2 ST1 GND

ST0 /T   /M    /WAIT LE /R    NC  NC  NC  VCC

IF /(HLDA)   T = /READ*/CS

IF (/HLDA)   R = READ*DS*/CS

             M = /HLDA*ST3*/ST2*/ST1*/ST0 +

                 /HLDA*/ST2*/ST1*ST0 +

                 /HLDA*ST3*ST2*/ST1 +

                 HLDA

          WAIT = /READY

           /LE = AS
```

3.3.2 DESCRIPTION

Most of the Z8001 and Am7990 control signals can directly be connected. The Z8001 status lines outputs, ST0-ST3, and LANCE (Am7990) $\overline{\text{HLDA}}$ input are used to indicate whether the cycle is a memory or non-memory cycle. The $\overline{\text{INTR}}$ pin of LANCE is connected to $\overline{\text{NVI}}$ pin of Z8001, since the LANCE does not return a vector during the interrupt acknowledge cycle.

Note: Program ACON, BCON, BSWP to "1" in CSR3 Reg.

3.4 80186 TO LANCE INTERFACE

This interface also uses a PAL design to reduce the parts count. The 80186/LANCE address and data buses can be connected directly together since they both have multiplexed buses. It seems natural to program the LANCE for ALE output. However, the PAL equations or a discrete design are easier if $\overline{\text{AS}}$ (CSR_3, ACON=1) is used. This is because the LANCE three-states ALE, the 186 does not. The $\overline{\text{INTR}}$, $\overline{\text{READY}}$, and $\overline{\text{HOLD}}$ signals from the LANCE are open-drain and should be pulled up. The $\overline{\text{BM}}_1$ signal from the LANCE or $\overline{\text{BHE}}$ from the 186 along with A_0 can be used to decode the data transfer type (Word/Byte). The external address buffers and data transceivers are enabled by the LANCE and the 186. The buffers and transceivers are enabled by whichever device is the master. The user should program the BCON, BSWP to "0," and ACON to "1" in CSR_3. Figure 3-3 shows the 80186 to LANCE interface connections.

3.4.1 LANCE IN BUS SLAVE MODE

The LANCE enters the Bus Slave Mode whenever the $\overline{\text{CS}}$ is active. In this example, the 186 becomes the master. $\overline{\text{RD}}$ or $\overline{\text{WR}}$ of the 186 can be used to generate $\overline{\text{DAS}}$ which can be deasserted after the LANCE asserts its $\overline{\text{READY}}$ output. The DTR and DEN of the 186 are used to control the data transceivers. When the LANCE is not selected, the DTR of 186 is used to generate the READ signal to the LANCE since DTR stays valid (High for Read, Low for Write operation) during the whole cycle.

3.4.2 LANCE IN BUS MASTER MODE

The LANCE enters the Bus Master Mode whenever the $\overline{\text{HOLD}}$ and $\overline{\text{HLDA}}$ signals are both active. The READ and $\overline{\text{DAS}}$ outputs are used to generate separate $\overline{\text{RD}}$and $\overline{\text{WR}}$ signals to meet the 186 system requirements. The $\overline{\text{DALI}}$ and the $\overline{\text{DALO}}$ outputs of the LANCE are used to control the data transceivers. The $\overline{\text{DAS}}$ signal asserted by the LANCE remains asserted until either the $\overline{\text{READY}}$ input is received or 25.6 us has elapsed from the time $\overline{\text{DAS}}$ was asserted. The time-out error is noted in CSR_0 (MERR=1) and the 186 is notified via interrupt.

3.4.3 186 TO LANCE INTERFACE PAL

```
PAL16L8                 JOE BRCICH
PAT001                  1 OCT 83
File: 186-90.PAL

80186 TO LANCE INTERFACE
ADVANCED MICRO DEVICES

ALE  /AS  DTR  NC  NC  DEN  NC  /READY  HLDA  GND
/CS  ARDY  READ  /R  /T  /DAS  /WR  /RD  LE  VCC

IF (/HLDA)    DAS = RD + WR

IF (/HLDA)  /READ = DTR

IF (/HLDA)      T = DTR * /CS

IF (/HLDA)      R = /DTR*DEN*/CS

IF (HLDA)      RD = READ*DAS

IF (HLDA)      WR = /READ*DAS

              /LE = /ALE + /AS

            /ARDY = /READY
```

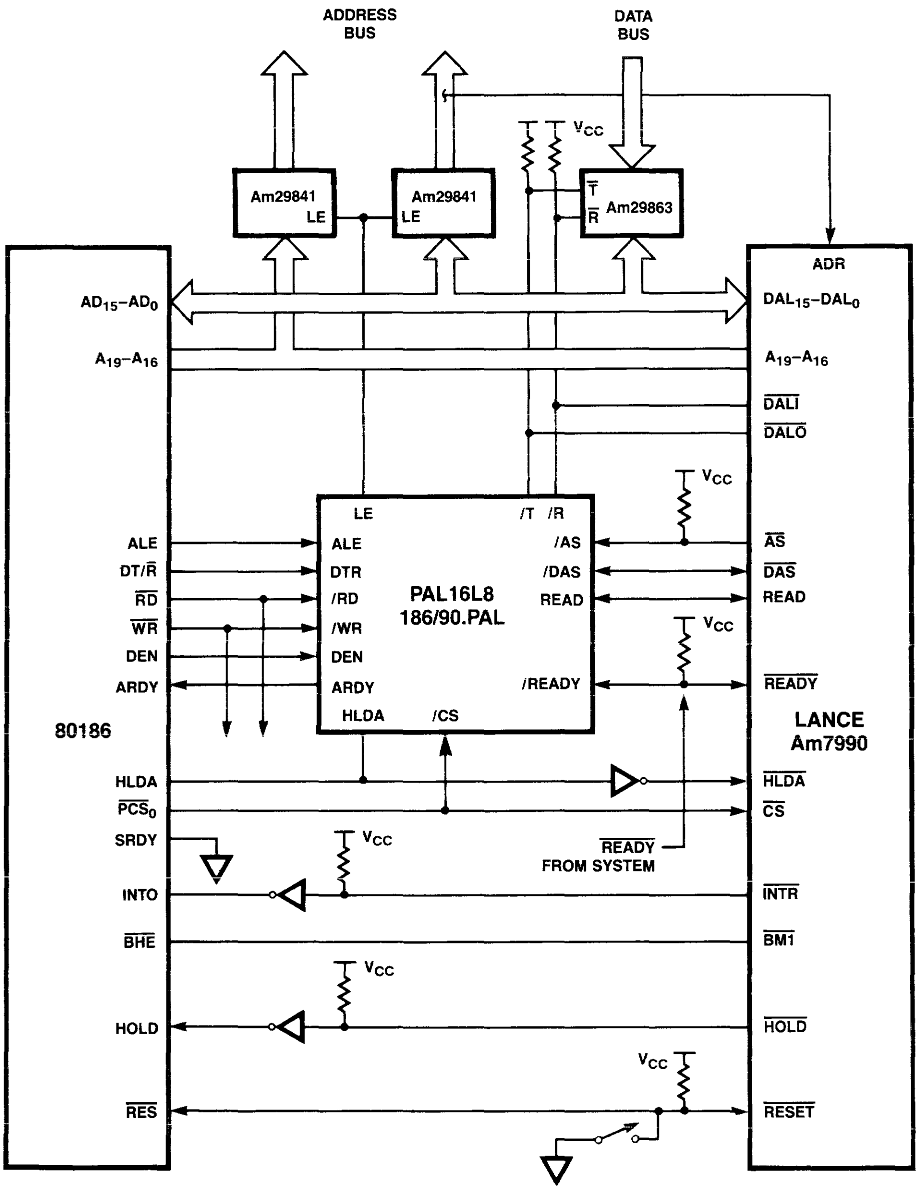
3.4.4 DESCRIPTION

This PAL design assumes that the 186 and LANCE are on the same board. The data bus buffer is only enabled if the LANCE is not selected. It seems natural to program the LANCE for ALE output. However, the PAL equations or indeed a discrete design is easier if $\overline{\text{AS}}$ is used. This is because the LANCE three-states ALE while the 186 does not. Note that data is valid on the falling edge of $\overline{\text{WR}}$ in min mode meeting the apparent requirement of the LANCE in early data sheets. Data set up time is specified with respect to the rising edge of DAS in later data sheets giving the designer more flexibility. All transfers to or from the LANCE must be word transfers.

Note: Program ACON to "1," and BCON, BSWP to "0" in CSR3 REG.

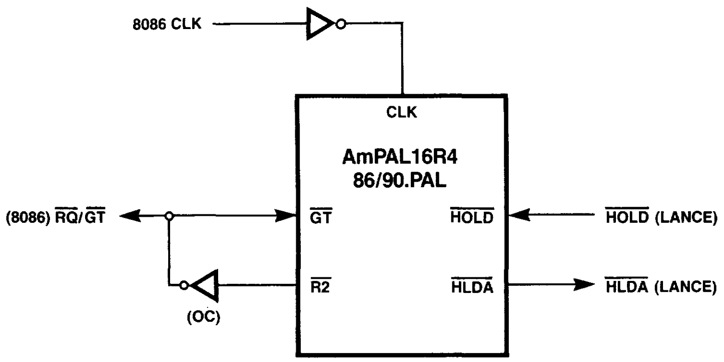
3.5 8086 TO LANCE INTERFACE

The LANCE interface to the 8086 and to the 80186 is quite similar, except that the bus request handshake is different when 8086 is configured in maximum mode. The 8086 has a bidirectional signal for both bus request and bus grant ($\overline{\text{RQ}}$/$\overline{\text{GT}}$). Both the bus request ($\overline{\text{RQ}}$), and bus grant ($\overline{\text{GT}}$) to/from 8086 are one CPU-clock wide and are synchronous to the CPU clock. Figure 3-4 shows a PAL design to do this conversion. Figure 3-5 shows the interface timing for this PAL. This PAL design could also be utilized to include other external logic requirements for interfacing the LANCE to 8086.



06363A-8

Figure 3-3 80186 to LANCE Interface



06363A-9

Figure 3-4 8086 $\overline{RQ}/\overline{GT}$, Am7990 HOLD/HLDA Conversion PAL

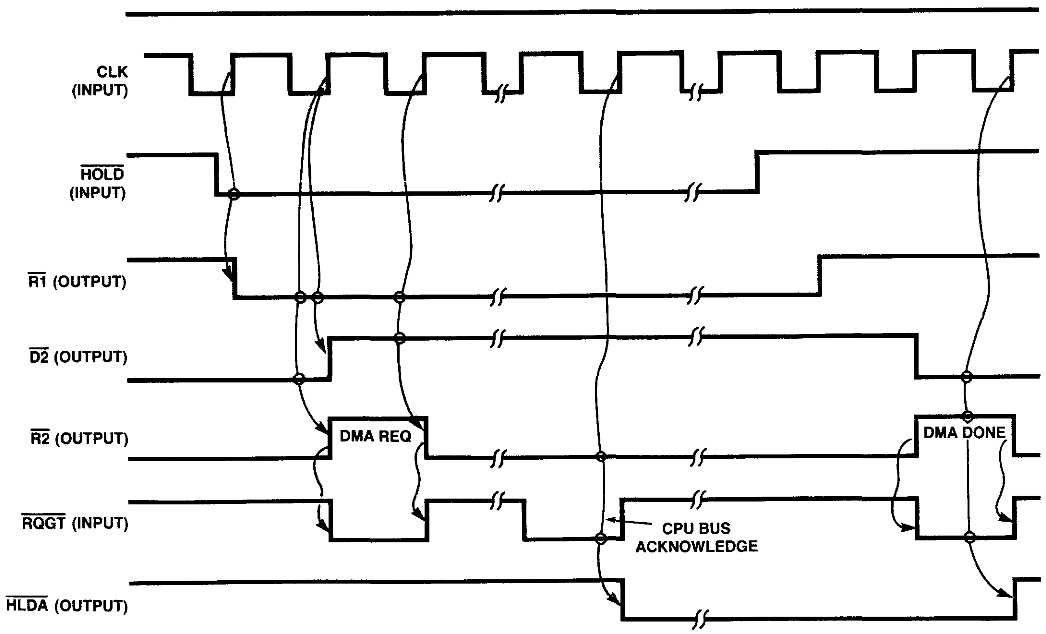


Figure 3-5 AmPAL16R4, 8086 (max. mode)/LANCE Interface Timing Diagram

06363A-10

3.5.1 8086 $\overline{RQ/GT}$ CONVERSION TO LANCE HOLD/HLDA PAL

```
AMPAL16R4                       RASOUL OSKOUY
PAT001                          MARCH 20, 1984

8086 RQ/GT CONVERSION TO LANCE HOLD/HLDA
ADVANCED MICRO DEVICES

CLK NC H̅O̅L̅D̅ G̅T̅   NC NC NC NC NC GND
NC  NC NC   H̅L̅D̅A̅ D̅2̅ R̅2̅ R̅1̅ NC NC VCC

       R1 := HOLD
       D2 := R̅1̅
       R2 := R1*D̅2̅ + R̅1̅*D2
     HLDA := GT*R̅2̅ + HLDA*D̅2̅
```

3.5.2 DESCRIPTION

This PAL converts the request and grant ($\overline{RQ}/\overline{GT}$) of 8086, when configured in maximum mode, to the LANCE Am7990 $\overline{HOLD}/\overline{HLDA}$. Both request (input to 8086) and grant (output from 8086) are one clock wide and are synchronous to the CPU clock.

CHAPTER 4
SOFTWARE CONSIDERATIONS

4.1 SOFTWARE TASKS

First the LANCE interface with the system is checked out both in Bus Master and Bus Slave Mode, and the internal logic of the LANCE is tested in diagnostic mode for both internal and external loopback test. Then, the LANCE is configured for normal operation via the Initialization Block. The user normally initializes the LANCE once for normal operation. Any changes in LANCE configuration require re-initialization.

The data handling tasks include assigning and chaining data buffers and checking the format rings for format errors. Transmit and receive drivers are required to set up the transmit and receive descriptors, buffer pointers, and related tasks. Status monitoring routines are also required. If re-initialization is necessary, the user must first process all the transmitted or received packets and to rearrange the packets queued for transmission in the Transmit Ring. This is necessary since the LANCE sets its pointers to the beginning of the Transmit and Receive Rings upon initialization.

The user must also process the interrupts and report the errors.

These tasks and the software support required are discussed in the following order:

Initialization
Diagnostics
Main Program
Descriptors and Buffers
Interrupts
Transmit Drivers
Receive Drivers
Status Monitoring and Error Processing

The flow charts and discussion outlined below examine one approach to typical driver routines which may be used to operate the LANCE. They are designed to guide the user in programming the LANCE within a typical application. However, the flowcharts are not intended as a direct implementation, which is system dependent.

4.2 INITIALIZATION

The LANCE is initialized (configured) for a desired operation by programming the CSR registers (CSR_{0-3}) and providing the parameters required by the LANCE in the Initialization Block. Refer to Chapter 1 for an overview of the system including the Initialization Block. For detailed information, refer to the LANCE data sheet. Initialization is a prerequisite for the LANCE operation in Bus Master mode.

Upon power-up or hardware reset, the internal logic of the LANCE is cleared, and the Stop bit in CSR_0 gets set. Once the Stop bit is set, either by a hardware reset or by software, all the activity in the LANCE stops. The Stop bit in CSR_0 must be set prior to any access to CSR_1-CSR_3. ACON, BCON and BSWP bits in the CSR_3 are programmed for the desired hardware interface. CSR_1 and CSR_2 get updated to contain the pointer to Initialization Block. (Refer to LANCE data sheet for more description on CSR registers.)

CSR_1 and CSR_2 are normally not affected when the Stop bit is set in CSR_0. (When the Stop bit is set, CSR_3 is cleared and CSR_0 is set to 0004 Hex, Stop bit set.) However, it is recommended to reload CSR_1 and CSR_2 whenever the Stop bit is set.

The Initialization Block contains the parameters (Mode Register, Node Address, Logical Address Filter, Receive, and Transmit Descriptor Ring Pointers) needed for normal or diagnostic operation.

The flow chart shown in Figure 4-1 shows a typical initialization routine which starts by invoking the diagnostic operation with error loop capabilities and if the diagnostic passes are successful, it initializes the LANCE for normal operation.

4.3 DIAGNOSTICS

The Mode Register Initialization Block can be used to set up the LANCE for diagnostic or normal operation. There are four user-programmable diagnostic modes. Each mode requires the user to change bits in the Mode Register and re-initialize the LANCE. The four modes are discussed in the following order:

1. Internal Loopback
2. External Loopback.
3. CRC Logic Check.
4. Collision Detection and Retry Logic.

4.3.1 INTERNAL LOOPBACK

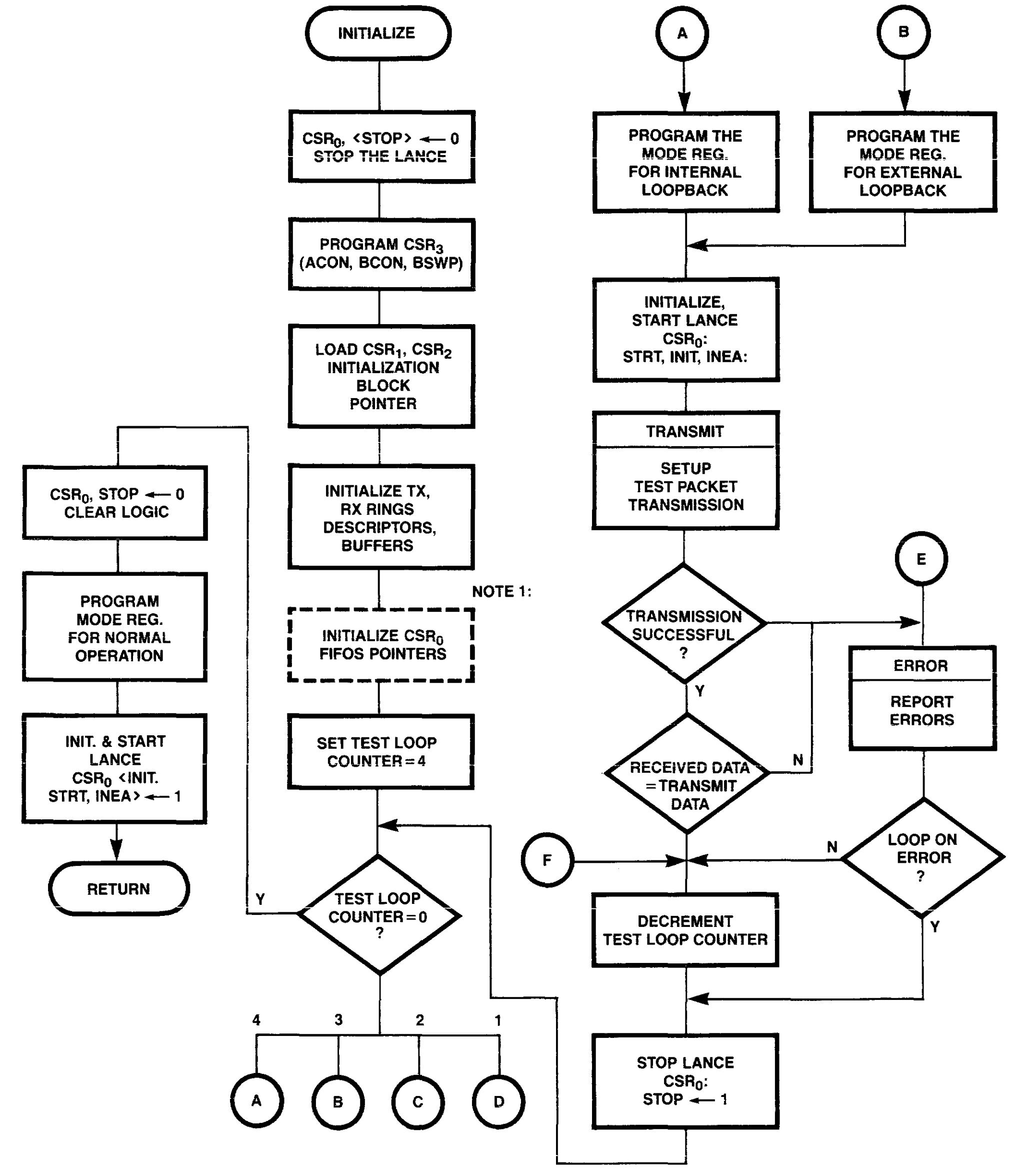
Although Ethernet is half-duplex, when in Loopback Mode the LANCE can operate in full-duplex by limiting the packet size to 8-32 bytes. The LANCE will not start the transmission of the loopback packet unless the whole packet (up to 32 bytes) is in the SILO. All the logic and data paths are tested in internal loopback except the transmit/receive drivers and receivers for the transmit data and receive data.

The LCAR bit should not be set in internal loopback mode since the LANCE does not monitor the RENA signal input to the LANCE. The LCAR bit is set when the sequence described below happens:

a) The LANCE is initialized in internal loopback mode. The loopback test can transmit and receive normally, as long as the transmit address matches the node address.

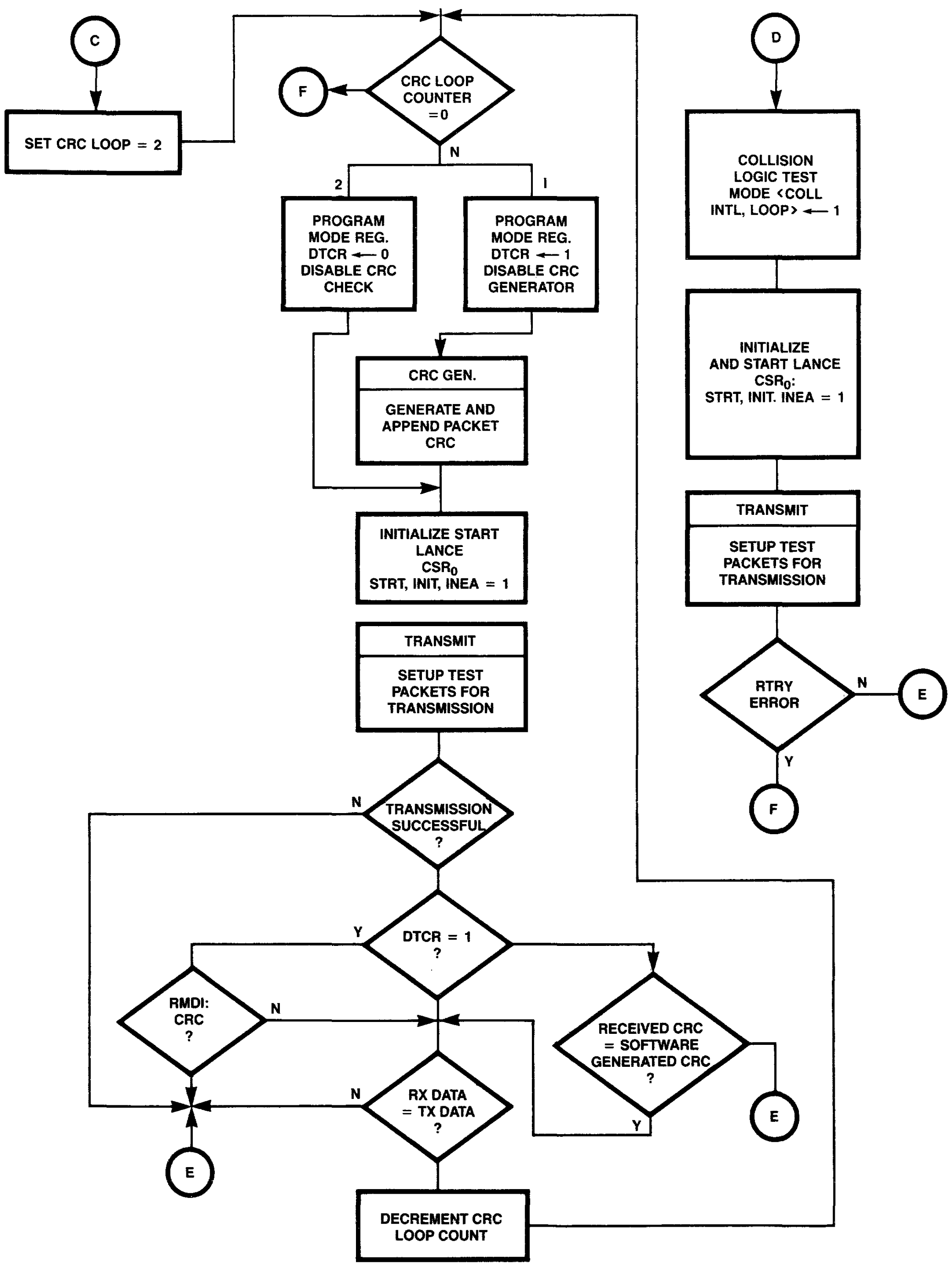
b) If at some point the LANCE transmits a non-matching packet, the packet will be sent but it will not be received. LCAR (lost carrier) will be set.

c) After transmitting a non-matching packet, the LANCE will no longer transmit in internal loopback mode, even if the matching address is put back. If the LANCE is then set-up to send a matching or a non-matching packet, the LANCE reads the descriptors from the Transmit Ring and relinquishes the descriptor to the CPU with LCAR bit set. It never DMAs data into the SILO. It does not actually transmit the packet out. Therefore, it does not receive the packet.


INITIALIZE
CSR0, <STOP> ← 0
STOP THE LANCE
PROGRAM CSR3 (ACON, BCON, BSWP)
LOAD CSR1, CSR2 INITIALIZATION BLOCK POINTER
INITIALIZE TX, RX RINGS DESCRIPTORS, BUFFERS
NOTE 1:
INITIALIZE CSR0 FIFOS POINTERS
SET TEST LOOP COUNTER=4
TEST LOOP COUNTER=0 ?
Y
4
3
2
1
A
B
C
D
CSR0, STOP ← 0
CLEAR LOGIC
PROGRAM MODE REG. FOR NORMAL OPERATION
INIT. & START LANCE CSR0 <INIT. STRT, INEA> ← 1
RETURN
A
B
PROGRAM THE MODE REG. FOR INTERNAL LOOPBACK
PROGRAM THE MODE REG. FOR EXTERNAL LOOPBACK
INITIALIZE, START LANCE CSR0: STRT, INIT, INEA:
TRANSMIT
SETUP TEST PACKET TRANSMISSION
TRANSMISSION SUCCESSFUL ?
Y
RECEIVED DATA =TRANSMIT DATA
N
E
ERROR
REPORT ERRORS
LOOP ON ERROR ?
N
Y
F
DECREMENT TEST LOOP COUNTER
STOP LANCE CSR0: STOP ← 1

NOTE 1: Optional step if CSR stacking is used. Not required for real time Interrupt Processing

Figure 4-1 Initialization/Diagnostic Flowchart (Sheet 1 of 2)


C
SET CRC LOOP = 2
F
CRC LOOP COUNTER =0
N
2
1
PROGRAM MODE REG. DTCR ← 0 DISABLE CRC CHECK
PROGRAM MODE REG. DTCR ← 1 DISABLE CRC GENERATOR
CRC GEN.
GENERATE AND APPEND PACKET CRC
INITIALIZE START LANCE CSR0 STRT, INIT, INEA = 1
TRANSMIT
SETUP TEST PACKETS FOR TRANSMISSION
N
TRANSMISSION SUCCESSFUL ?
Y
DTCR = 1 ?
RMDI: CRC ?
N
RECEIVED CRC = SOFTWARE GENERATED CRC ?
Y
E
N
RX DATA = TX DATA ?
E
DECREMENT CRC LOOP COUNT
D
COLLISION LOGIC TEST MODE <COLL INTL, LOOP> ← 1
INITIALIZE AND START LANCE CSR0: STRT, INIT, INEA = 1
TRANSMIT
SETUP TEST PACKETS FOR TRANSMISSION
RTRY ERROR
N
E
Y
F

06363A-11

Figure 4-1 Initialization/Diagnostic Flowchart (Sheet 2 of 2)

In the internal loopback mode, when a packet is transmitted that is not addressed to the node itself, LCAR is set and indicates this condition. If the address detection logic of the LANCE is being tested for a non-matching test in internal loopback mode, the first non-matching transmit packet always gets transmitted. Only the packets following the transmission of the non-matching packet fall into this condition and never get transmitted. Re-initializing the LANCE is the only option when this problem occurs. This problem will not occur as long as the LANCE is transmitting packets to itself (matching addresses) in internal loopback mode. The external loopback can be used for testing the packets with non-matching addresses.

4.3.2 EXTERNAL LOOPBACK

External Loopback operates in much the same fashion as Internal Loopback, except that the test packet is actually transmitted out, starting from the LANCE to the SIA, to the transceiver, and finally to the coaxial cable. The transmitted data wraps around, in the same fashion, from the coaxial cable to the LANCE. External Loopback test will not interfere with any other node attached to the Ethernet cable, since the packet is normally addressed to the node itself or gets discarded as a runt packet (loopback packet size max = 32 bytes).

The multicast addressing is not operational in External Loopback Mode. The LANCE transmits the packet but it will not receive it. However, DTCR (Disable Transmit CRC) in the Mode Register can be set to "1" when multicast addressing in External Loopback Mode is desired. In this case, the user has to append the correct four byte CRC when transmitting a packet or there will be a CRC error when the packet is received. The multicast addressing can be tested in Internal Loopback Mode, or Receive Mode when another node is transmitting.

4.3.3 CRC GENERATOR/CHECKER LOGIC TEST

During the loopback test (internal or external), the CRC logic may be tested. Since the LANCE operates in half-duplex mode, the CRC generator or checker must be tested separately as follows. The Transmit CRC Generator is tested by programming DTCR=0 in the Mode Register. This enables the transmit CRC generation and disables the Receive CRC Checker. Assuming the test packet is 32 bytes, the LANCE will transmit 32 bytes and store 36 bytes of data in the receive buffer. The extra four bytes are the CRC generated by Transmit CRC Generator. The received CRC bytes are compared against the software generated CRC.

The Receive CRC Checker is tested by setting the DTCR bit in the Mode Register. The user then calculates the CRC for the test packet and appends it to the end of the test packet for transmission (test packet max size = 32 + 4 bytes CRC). The Receive CRC Checker checks the last four bytes of the packet for CRC error and updates the receive descriptors accordingly. The CRC logic may be tested in either External or Internal Loopback Mode.

4.3.4 COLLISION DETECTION/RETRY LOGIC TEST

The collision detection can be tested by forcing collision, setting the COLL, LOOP, and INTL bits of the Mode Register) to "1." The LANCE is then initialized and started (CSR_0: INIT,STRT,INEA=1). The collision bit, COLL, being set will cause the LANCE to detect a collision on each loopback attempt. The result is 16 total transmission attempts with a retry error reported in TMD_3.

The SIA/transceiver collision interface to the LANCE is checked by placing the LANCE in External Loopback. The LANCE will look for a collision detection within 2 us following the completion of a transmission. This is a transceiver self test feature which is known as SQE (Signal Quality Error) test or "Heartbeat." The CERR bit gets set if the Heartbeat is missing within the specified time. The failure can be caused by the external collision path (transceiver-to-SIA-to-LANCE) and/or the transceiver itself.

4.4 MAIN PROGRAM

The LANCE is configured for normal operation via the initialization Block. LANCE is initilized once for normal operation. However, any configuration change requires re-initialization. Figure 4-2 shows a flow diagram of the main program for normal operation.

4.5 DESCRIPTOR AND BUFFER PITFALLS

4.5.1 TRANSMIT MESSAGE DESCRIPTORS, TMD1 AND TMD3

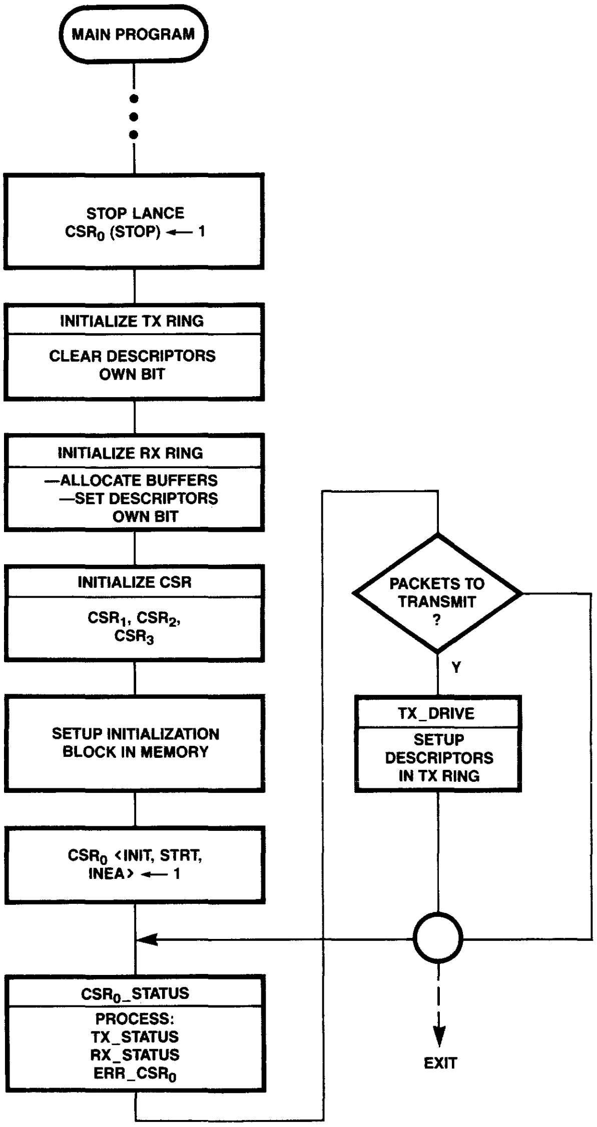
"One" flag in TMD_1 is not valid when "LCOL" is set in TMD_3.

4.5.2 TRANSMIT DESCRIPTOR RING FORMAT

When the LANCE is polling the Transmit Ring, it will skip over Transmit Ring entries having a bad format. For the first buffer in a packet, when the LANCE owns the buffer and STP (Start of Packet) is not set, it will simply turn the ownership back over to the host. The LANCE will then generate a TINT interrupt and go on to the next buffer. The LANCE continues to skip over transmit buffers (by clearing the OWN bit and setting TINT bit) until it finds a buffer with a good format (both STP and OWN bit set) to transmit. When the LANCE finds a transmit buffer owned by the host (OWN = 0), it means that there are no more buffers to transmit.

4.5.3 DATA CHAINING

During either a receive or a transmit with data chaining, the LANCE generates a done flag (RINT or TINT) and interrupt only at the end of the chain. It will not generate any interrupt flags whenever a buffer of the chain is filled or transmitted, unless there is an error. The ownership bits, however, are turned over to the host as each buffer is completed. This can cause a problem if the first transmit buffer is short as explained below.



06363A-12

Figure 4-2 Main Program for the LANCE in Normal Operation

4.5.4 TRANSMIT BUFFERS

When in Transmit Mode, the LANCE can transfer up to 32 bytes of data into the Silo before transmission is started if the cable is busy at the time. A collision could occur at any time during the transmission of the first 64 bytes (first slot time). During this transmission, 64 additional bytes could be read from the buffer. Therefore, the first transmit buffer must be at least 96 bytes long to ensure that the LANCE does not relinquish the first buffer before it has sent the first 64 bytes of the packet over the Ethernet cable. Including a small extra margin, 100 bytes was chosen as the minimum size required for the first buffer when data chaining to ensure proper handling of collisions.

At the beginning of a transmission, the LANCE internally stores the address and byte count of the first transmit ring entry, in case a retransmission is required. In data-chaining, after each buffer is read into the Silo, the buffer is returned to the host by resetting the "own" bit. As explained above, if the first transmit buffer is too short, a collision could occur after the first buffer was completed and the ownership bit turned over to the host.

Since a collision causes an automatic retry and since the LANCE has the address of the first buffer, the LANCE will attempt to retransmit the buffer that it had just given back to the host. This violates the mutual exclusion rule of the ring ownership by accessing a ring entry that it does not own. This can also cause incorrect and unexpected transmit ring status changes as follows: if the host is polling the ownership bit, it would see ownership of the first transmit buffer and start to service it. At the same time, the LANCE may be retransmitting it and even updating the status for that buffer.

When data-chaining is not used, the LANCE does not return the buffer to the host until the entire packet has been sent without collision or other error. Therefore, for buffers that are not data-chained, the minimum buffer size for the first buffer is the minimum packet size of 64 bytes required by Ethenet to ensure proper handling of collisions. However, the LANCE is capable of transmitting a packet as small as one byte.

To summarize, when the buffers are data chained in Transmit Mode, the minimum size for the first transmit buffer is 100 bytes. When the buffers are not data-chained, the minimum size for the transmit buffer is 64 bytes to conform to Ethernet standards. These minimums prevent the LANCE from violating the mutual exclusion principle of the ring ownership bits as a result of a collision. The mutual exclusion rule states that the LANCE must not access a ring entry it does not own. The slot time is the time it takes to transmit 64 bytes over Ethernet. A collision can occur at any time during the first slot time.

4.5.5 RECEIVE BUFFERS

The minimum size for the first receive buffer is specified as 64 bytes. If this spec is not adhered to, the LANCE could cause the violation of the mutual exclusion principle of the ring ownership bits.

This happens if the receive buffers are less than 64 bytes each and a packet that is larger than the first buffer but less than 64 bytes (a runt packet) arrives. The LANCE fills the first buffer, updates that buffer's status and turns over the ring ownership to the host. When the packet terminates as a runt, the LANCE discards it. The LANCE backs up its internal receive ring address pointer and byte count registers to their values at the beginning of the reception so that it can use this buffer for the next packet. However, this buffer has already been turned over to the host.

When the next packet arrives, the LANCE will reuse the buffer that it filled during the previous runt packet although it is now owned by the host. This violates the mutual exclusion rule of the ring ownership bits.

If the host had been polling the ownership bits, it could have already used the data in that receive buffer although it was not valid data. The host expects the next receive packet to start at the next receive buffer but the LANCE starts at the previous buffer. If the next receive packet is a legal packet, the LANCE will data chain at least two buffers because the first buffer is too small. The host then starts looking at the second buffer of the chain because it has already serviced the previous buffer when the ownership bit was turned over during the previous runt packet. Under these conditions, the host will not see the expected packet format (the destination address, source address, etc.) in the receive buffer.

4.6 INTERRUPTS

The interrupt pin is simply an OR of the interrupt causing conditions. The occurrence of another interrupt after the interrupt pin is asserted does not generate another transition on the interrupt pin. Interfaces that require a transition per interrupt may use the following software routine to force a transition on the interrupt pin.

READ CSR_0 AND STORE IN MEMORY OR A REGISTER

CLEAR THE INTERRUPT ENABLE (BIT 06) IN THE STORED CSR_0

WRITE CSR_0 WITH THE STORED VALUE (BIT 06 IS NOW CLEAR)

SET THE INTERRUPT ENABLE BIT IN CSR_0

At the time the interrupt is first serviced, this routine first reads CSR_0 to capture the register and store its value in memory or a register. Then the stored value is written to CSR_0 with Bit 06 clear, clearing the interrupt-causing conditions present at the time the register was read. They are cleared because they are all "write-one-to-clear" bits. This clears out the old interrupt flags and preserves any new ones set after the register was read. It causes the LANCE to deassert the interrupt pin because interrupt enable is now clear. The last step is to set interrupt enable again. This must be done with a Write Only instruction, such as MOVE, so that any new interrupt flags are not cleared by a Read-Modify-Write operation. At this time, if there were any new interrupt flags set after CSR_0 was first read, the interrupt pin will be reasserted generating a new transition for the new interrupt flags.

It is possible to have more than one interrupt-causing condition for each interrupt generated. For example, if an interrupt is received and the RINT bit is set, there could have been more than one receive buffer turned over to the host, because other receive buffers could have been filled before the host services the interrupt. This is also true for transmit interrupts. In fact, a receive buffer could have been turned over to the host after a transmit interrupt has been generated. This happens if a transmit interrupt was generated, if the host software read CSR_0, and if a receive buffer was filled before the host software finishes the transmit routine. Therefore, it is wise to check both the Receive and Transmit Rings for buffers owned by the host whenever an interrupt occurs.

LANCE asserts the interrupt signal ($\overline{INTR}$) every time it updates the internal register, CSR_0, provided the INEA bit is set. Interrupt remains asserted until one of the following conditions is met:

1. Source of interrupt is removed.
2. $\overline{RESET}$ is asserted.
3. Stop bit gets set in CSR_0.

There are four types of interrupts noted in CSR_0:

1. Initialization done.
2. Fatal and non-fatal errors.
3. Receive Descriptor update.
4. Transmit Descriptor update.

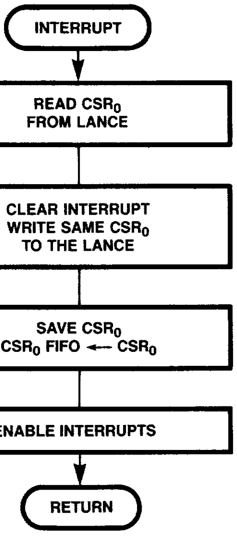
It is important and necessary to acknowledge the interrupts in real time so that the segment of the LANCE status being updated is not lost. This also enables the processing of transmit and receive buffers promptly.

There is only one interrupt condition, MISS error, which requires a response. When MISS error occurs, it must be cleared by the user, or no more packets will be received, even if some receive buffers have been relinquished to the LANCE. (The MISS error requirement applies only to the LANCE Rev. B. The LANCE Rev. C does not require the MISS bit to be cleared to receive the following packets.) However, there is enough time (67.2 us) between the MISS error bit getting set and the arrival of another packet. The 67.2 us is measured based on back-to-back short packets (64 bytes) addressed to this node, and the size of the SILO (48 bytes).

No other interrupt bits in CSR_0 will result in the loss of any packet. If interrupt is not serviced before the next interrupt occurs, then the new status bits get ORed with the old ones before CSR_0 is updated. It is possible for the user to lose the sequence of status being updated.

The discussion below describes one way of handling a long interrupt latency. Since some software overhead may be involved in responding to the interrupts, a buffering scheme (software FIFO) may be used to ensure that all the events are recognized and serviced in the order of occurrence.

A software FIFO as large as Transmit/Receive Ring size is allocated to store the CSR_0s in the order of interrupt occurrence. The interrupt service routine merely reads the CSR_0 and stores it for later processing by status processing routines thus making it as short as possible. (See Figure 4-3.) This approach will assure that all the events are captured in the order of occurrence.



INTERRUPT
READ CSR0 FROM LANCE
CLEAR INTERRUPT WRITE SAME CSR0 TO THE LANCE
SAVE CSR0 CSR0 FIFO ← CSR0
ENABLE INTERRUPTS
RETURN

06363A-13

Figure 4-3 Interrupt Service Routine

4.7 TRANSMIT DRIVER CONSIDERATIONS

The host communicates with the LANCE, via the Transmit Ring, for packet transmission. There can be up to 128 descriptor entries, four words per entry, in the Transmit Ring.

The user should have two pointers associated with the Transmit Ring:

1. TXDESW — Transmit Descriptor Write Pointer
2. TXDESR — Transmit Descriptor Read Pointer

TXDESW points to the first descriptor of the packet queued for transmission by the host (OWN=1, STP=1). TXDESR points to the first descriptor of the packet just relinquished by the chip (OWN=0, TINT=1).

TXDESR and TXDESW originally point to the same location, to the beginning of the Transmit Ring. Once the packet transmission starts, the TXDESW pointer normally leads the TXDESR pointer. The user sets up the descriptors by flowing/wrapping around the ring, and LANCE chases the user sequentially to send the buffers out.

Transmit data buffers are usually scattered around in the memory. Before initiating a transmission, the user needs to invoke the lower level transmit driver routine to change the size of large and small buffers scheduled for transmission. One approach, as illustrated in Figure 4-4, is to shrink down the large buffers to 1500 bytes each and chain the small buffers together by taking advantage of LANCE's buffer management data chaining scheme. Three types of lists can be associated with transmit routines to transmit a large block of data (for example, 10K bytes).

List 1:
This list contains pointers to buffers (any location in memory), with different buffer sizes (no limitation in buffer sizes), to be transmitted as one block in the order defined sequentially in the list. LIST1PTR is the pointer to this list. Each entry on the list designates the buffer pointer and buffer size. End of list is recognized by an end of list flag (for example, FFFF...).

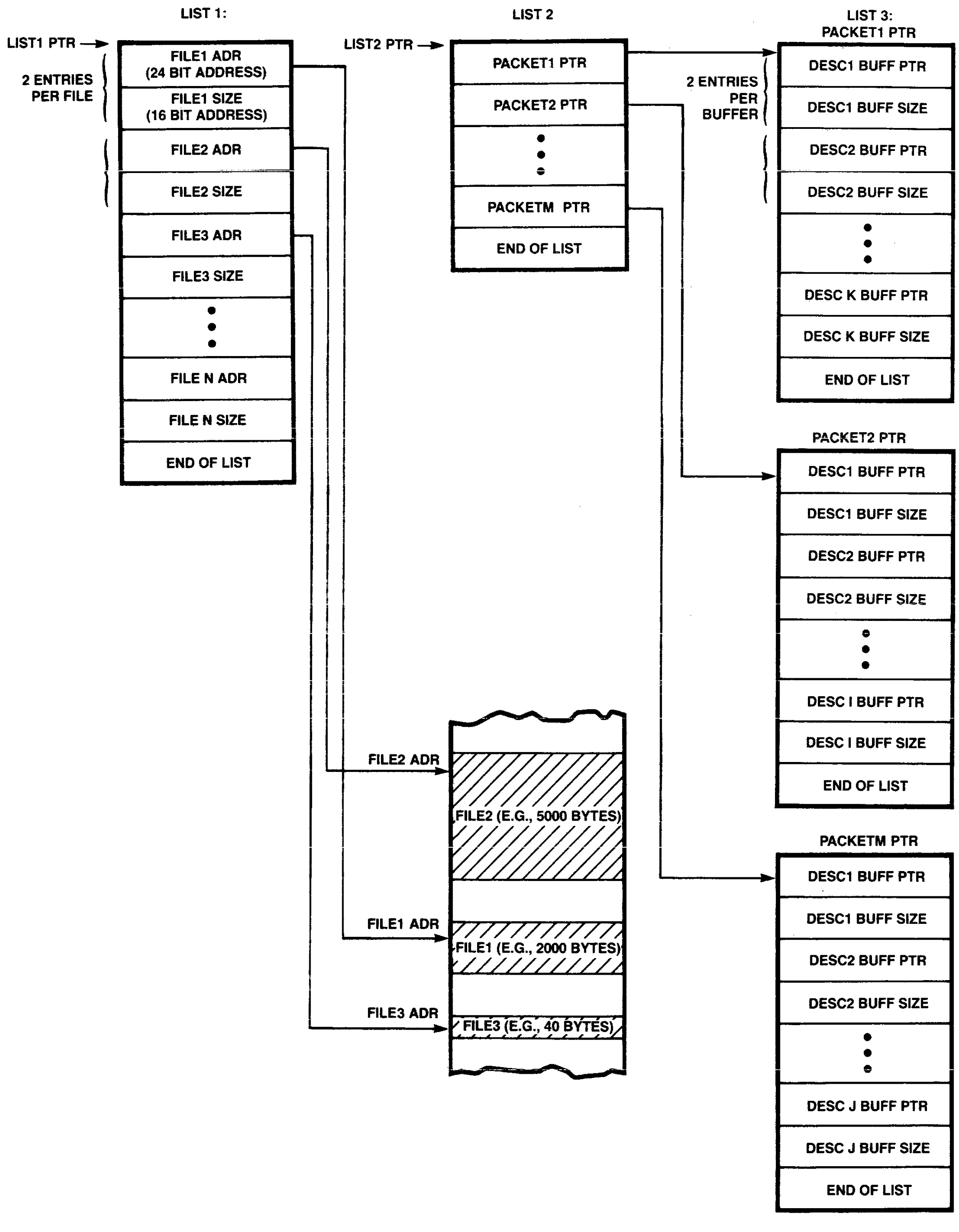


Figure 4-4 Transmit Block Buffer Packing

List 2:
One of the transmit routines uses List 1 to generate List 2. Each entry on List 2 is a pointer to a packet (List 3). List 2 is a translation of a block of data (List 1) with large and/or small buffers to one or more packets. End of list is recognized by the end of list flag (for example, FFFF...). LIST2PTR is a pointer as an input to the lower level transmit driver routine to transmit the whole block (all the packets in List 2).

List 3:
Each entry on this list designates the buffer pointer and the buffer size (<=1514 bytes; see Note 1) for each buffer that is chained together to be transmitted as one packet. At the end of the packet, a list flag is used to designate the end of the list. The lower level transmit driver routine transfers both the buffer pointers and buffer sizes directly to the transmit ring descriptors. The buffer pointed to by the first entry in list 3 must contain the header information (source address, destination address, and type (or length) field.

NOTE 1

The user should be cautious about address and type field insertion for transmission of each packet. Since LANCE does not insert destination or source addresses or type field, the user must append the address and type field (14 bytes) to the beginning of each packet.

The transmit driver routine, TXDRIVE, shown in Figure 4-5 uses LIST2PTR as the pointer to the packet list to set-up the transmit descriptors for transmission. Transmit driver routine (Figure 4-5) and TX_Status routine (Figure 4-6) keep track of the two pointers, TXDESW and TXDESR by using a software FIFO. The transmit driver routine writes the pointer, TXDESW, to the FIFO and the TX_Status routine reads the pointer, TXDESR, from the FIFO. The FIFO size should be as large as the transmit ring size. This approach allows the transmission process of the packet to be performed in the correct order and minimizes the software complexity of managing the Transmit Ring.

As pointed out earlier, the TXDESW pointer normally leads the TXDESR. If these two pointers point to the same location (TXDESR = TXDESW), it is an indication of having either a small ring (much less than 128) or the LANCE transmitter has been turned off because of some particular error(s). Should the transmitter shut off, the user is notified via interrupt. CSR_0 Register, TXON Bit, may then be checked to see if the transmitter is on.

The LANCE normally polls the Transmit Ring descriptor once every 1.6 ms until it finds a packet (defined/addressed in the descriptor) to be sent out. However, the user may demand faster polling time (every 1-2 us) by continuously setting the TDMD bit in CSR_0, after it gets cleared by the LANCE. The user should normally set the TDMD bit every time it relinquishes (sets the OWN bit) the first descriptor of a packet to the LANCE.

NOTE

As shown in the transmit routine flowchart, the user should not relinquish the descriptors (buffers) until all the descriptors for the packet have been set up. (ENP bit is set in the last descriptor.) After the descriptors are set up, the user should clear the OWN bit of each descriptor in the reverse order (last descriptor first, first descriptor last). This technique will prevent transmit buffer error (TMD_3, BUFF error).

As shown in Figure 4-6, there are two variables, TXDESR and TINTCNT, used for processing the packets that LANCE has transmitted. TINTCNT corresponds to the number of packets that have been transmitted (times that the TINT bit has been set in CSR_0) but not processed by the CPU to update their status. TXDESR is the pointer to the first descriptor of the packet relinquished by the LANCE and not processed by the CPU. This mechanism is used to assure servicing the packets in the order of transmission and to prevent TINT being set twice without the CPU awareness that two packets have been transmitted by the LANCE.

When checking for errors, the user needs to check both the ERR and BUFF bits, since the ERR does not include the BUFF error. When processing errors, BUFF or UFLO errors should be treated as fatal errors since these errors are due to a malfunction in hardware and/or software designed around the LANCE and it is under user's control. LCOL error should also be treated as a fatal error. In a maximum Ethernet configuration, a collision might occur within the first 64 bytes of the packet but not beyond. LCOL error can also be due to a fault (open or short) in the coaxial cable.

LCAR error can be due to a malfunction in the receive section at the transceiver or fault between transceiver and SIA (Receive ± pair) or SIA and LANCE (RENA input to the LANCE). The LANCE requires RENA to become and remain active before TENA is deasserted; otherwise an LCAR error will be set. If RENA drops out while transmitting (TENA High), an LCAR error will get set and the LANCE will continue to transmit the packet. A fault (short) in the coaxial cable can also cause an LCAR error.

The RTRY error is set when the LANCE attempts 16 times to transmit the packet and does not succeed. This error is less likely to occur. However, this error can occur in a very heavily loaded network, a failure of a node in the network, or an open in the coaxial cable.

As shown in the flowchart in Figure 4-6, after the packet has been transmitted, in particular when the buffers are data chained, the user should check the status of all descriptors for errors (ERR bit). This is because RTRY error is updated in the first descriptor, and LCOL and UFLO can occur anywhere from the first descriptor to the last descriptor of the packet. When late collision occurs (LCOL=1), the LANCE continues to transmit the packet.

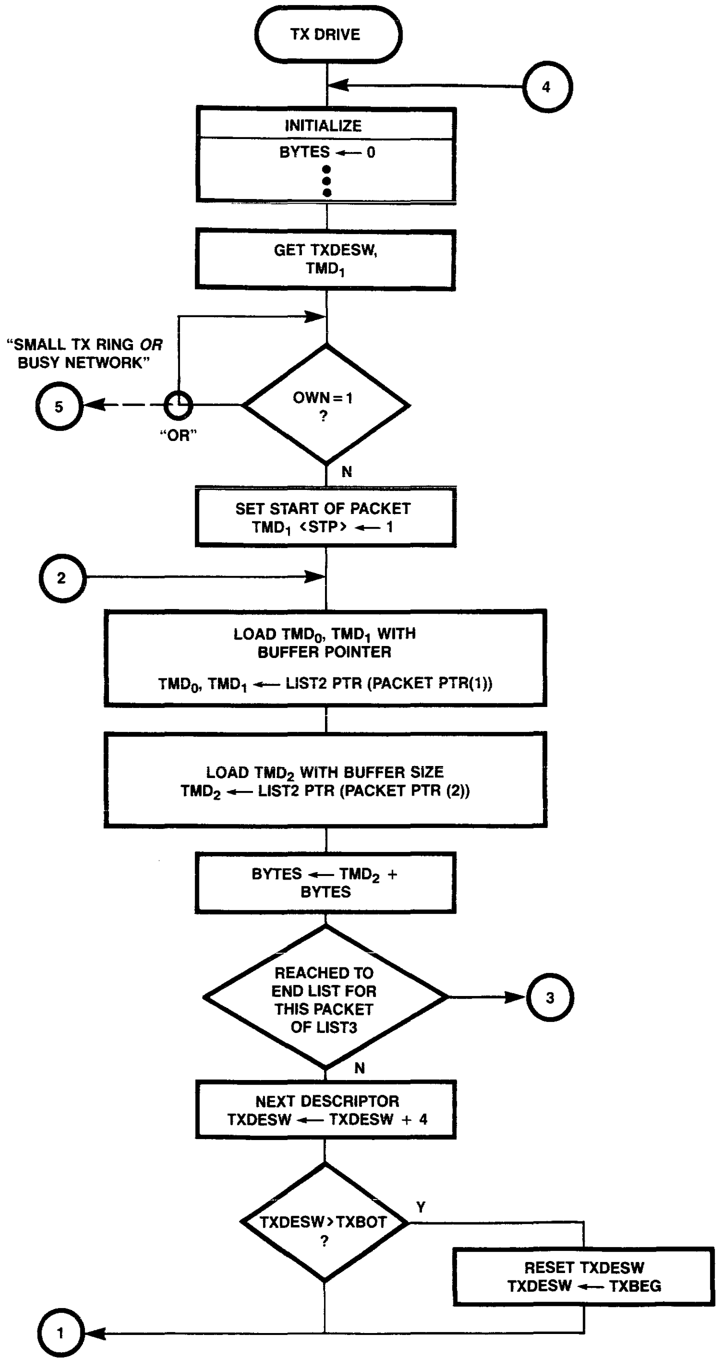
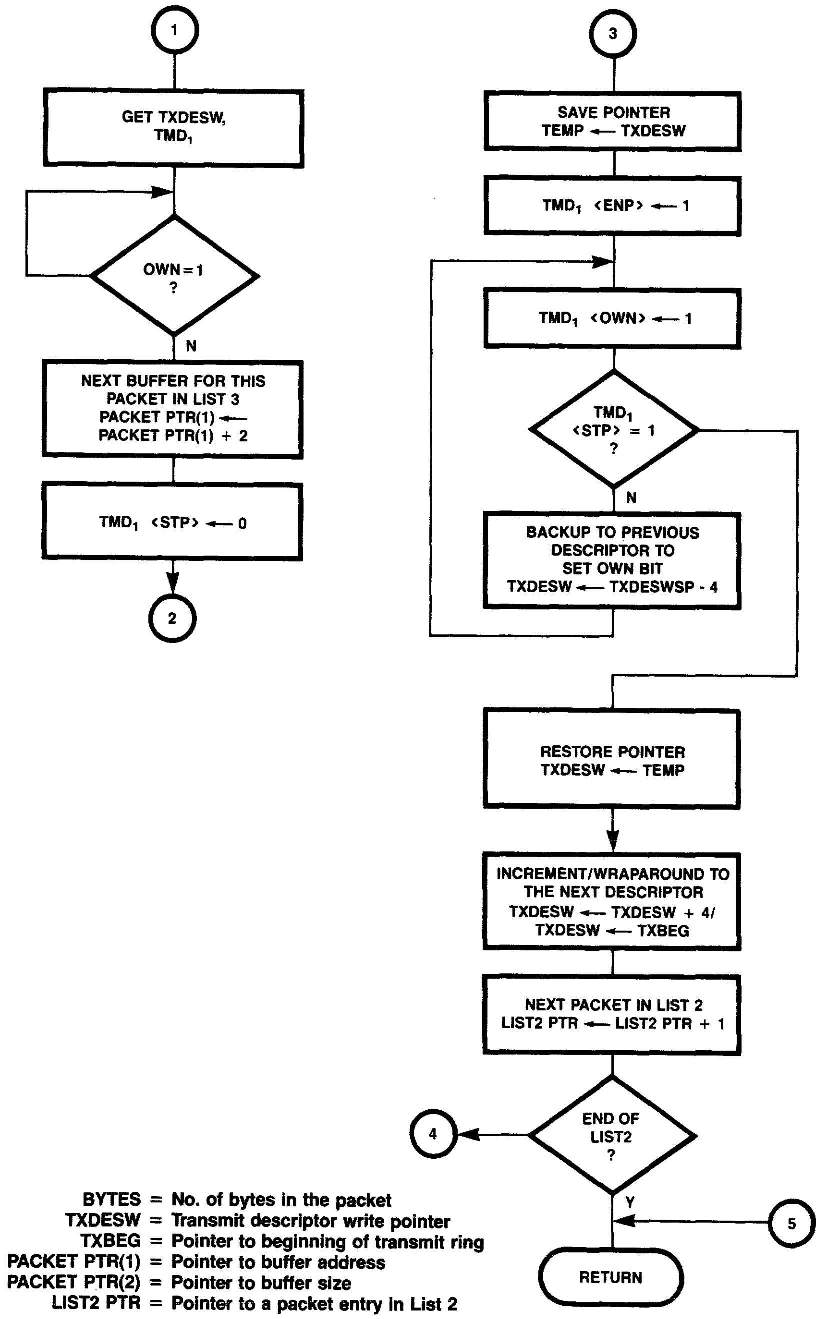

TX DRIVE
4
INITIALIZE
BYTES ← 0
GET TXDESW, TMD1
"SMALL TX RING OR BUSY NETWORK"
5
"OR"
OWN=1 ?
N
SET START OF PACKET TMD1 <STP> ← 1
2
LOAD TMD0, TMD1 WITH BUFFER POINTER
TMD0, TMD1 ← LIST2 PTR (PACKET PTR(1))
LOAD TMD2 WITH BUFFER SIZE
TMD2 ← LIST2 PTR (PACKET PTR (2))
BYTES ← TMD2 + BYTES
REACHED TO END LIST FOR THIS PACKET OF LIST3
3
N
NEXT DESCRIPTOR TXDESW ← TXDESW + 4
TXDESW>TXBOT ?
Y
RESET TXDESW TXDESW ← TXBEG
1

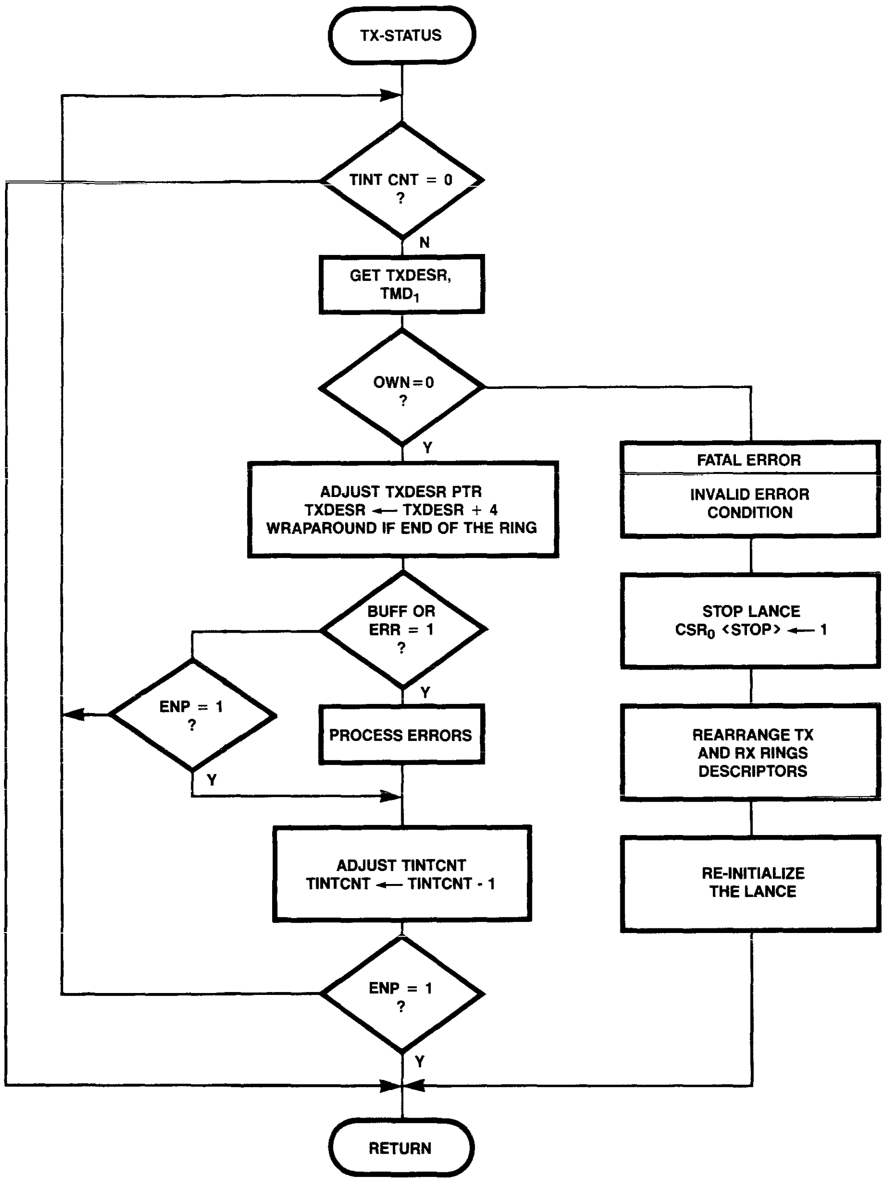
Figure 4-5 Transmit Routine (TXDRIVE) to Setup Packets for Transmission (Sheet 1 of 2)


1
GET TXDESW, TMD1
OWN=1 ?
N
NEXT BUFFER FOR THIS PACKET IN LIST 3 PACKET PTR(1) ← PACKET PTR(1) + 2
TMD1 <STP> ← 0
2
3
SAVE POINTER TEMP ← TXDESW
TMD1 <ENP> ← 1
TMD1 <OWN> ← 1
TMD1 <STP> = 1 ?
N
BACKUP TO PREVIOUS DESCRIPTOR TO SET OWN BIT TXDESW ← TXDESWSP - 4
RESTORE POINTER TXDESW ← TEMP
INCREMENT/WRAPAROUND TO THE NEXT DESCRIPTOR TXDESW ← TXDESW + 4/ TXDESW ← TXBEG
NEXT PACKET IN LIST 2 LIST2 PTR ← LIST2 PTR + 1
END OF LIST2 ?
4
Y
5
RETURN

BYTES	=	No. of bytes in the packet
TXDESW	=	Transmit descriptor write pointer
TXBEG	=	Pointer to beginning of transmit ring
PACKET PTR(1)	=	Pointer to buffer address
PACKET PTR(2)	=	Pointer to buffer size
LIST2 PTR	=	Pointer to a packet entry in List 2

06363A-15

Figure 4-5 Transmit Routine (TXDRIVE) to Setup Packets for Transmission (Sheet 2 of 2)



TINTCNT	=	No. of TINT interrupts
TXDESR	=	Pointer to the first descriptor of the packet relinquished by the LANCE and not processed by the CPU
TXBOT	=	Pointer to the bottom of the ring
TXBEG	=	Pointer to the beginning of the ring.

06363A-16

Figure 4-6 Transmit Packet Status Processing (TX-STATUS)

4.8 RECEIVE DRIVER CONSIDERATIONS

The host communicates with the LANCE, via the Receive Ring, for packet reception. There can be up to 128 descriptor entries, four words per entry, in the Receive Ring.

The user needs to allocate some buffers through the descriptor entries in the Receive Ring for the arrival packets, if the receiver is enabled (turned on via Initialization Block). Multiple buffers with different sizes scattered around in the memory may be allocated to the LANCE for reception. Receive driver routines and buffer management tasks can be relatively complex when the user needs to keep up with incoming back-to-back packets.

The user has 3 major tasks to handle at all times:

1. Respond fast enough to LANCE's DMA requests
2. Always have enough available receive buffers for arriving packets
3. Provide fast interrupt response time and short interrupt service routine to capture/distinguish the events in the order of occurrence for each packet. However, the packets will not be lost if interrupt processing is not in real time; CSR_0 update is an OR operation of the new CSR_0 and the previous CSR_0.

The user must keep track of the incoming receive packets, process the packets (check status and empty buffers), and allocate enough available receive buffers to store all incoming packets. A global pointer associated with the Receive Ring is needed to process the received packets sequentially in the order of arrival. This pointer, Received Descriptor Pointer (RXDPTR), is used in RX-Status routine (Figure 4-7) to locate the buffers for the received packet, check the integrity of the packet, and update the value of RXDPTR (pointer to the first descriptor of the next packet).

The node processor and the host CPU software may use a semaphore approach to communicate with each other. This is accomplished by allocating a software FIFO to be a Write Only by the node processor, and a Read Only by the host CPU. This FIFO has multiple entries each of which contains two locations: a pointer to the first descriptor for a packet (RXDPTR) and the number of descriptors (buffers) for this packet (RXDPSNO). The host CPU is interrupted by the node processor every time an entry is added to the list (FIFO). The host CPU, in return, reads the entry from FIFO to empty the buffer(s) and relinquish the buffers to the LANCE using the RXDPTR and RXDESNO. This technique speeds up the received packet processing time and diminishes the probability of missing packets as a result of no receive buffers being available.

The data chaining scheme in the LANCE makes efficient use of the memory during the packet reception. It is practical to assign small buffers (for example, 256 bytes) for receive buffers allocated to the LANCE. With small buffers, there is less unused space in each buffer when storing packets that are short. For packets larger than the buffer, data-chaining is used. Storing small packets in large buffers results in large unused and unusable areas of memory and results in the data being in fragmented areas scattered around in the memory.

RINTCNT variable shown in Figure 4-7 contains the number of times that the RINT bit has been set in CSR_0 without being processed by the user. This value corresponds to the number of packets which have been received with or without the error and are awaiting processing. Figure 4-7 uses RINTCNT and RXDPTR to process back-to-back packets in the order of arrivals.

As it implies in Figure 4-7, the user should start checking the RMD_1 of each descriptor until it finds either ENP or ERR bit set. IF ERR bit is set and ENP bit is not, it indicates that an error (such as BUFF or OVFL) occurred before the packet ended. If ENP bit is set, it indicates that the packet has ended (cable has gone idle) and the current descriptor is the last descriptor for this packet. When ENP and ERR are both set, it is an indication that there is a CRC or both CRC and framing error.

It is important to note that the LANCE interrupts the user when it updates the last descriptor (clears the OWN bit) for current received packet. It does not interrupt (does not set RINT) for each descriptor update when the packet consists of multiple descriptors.

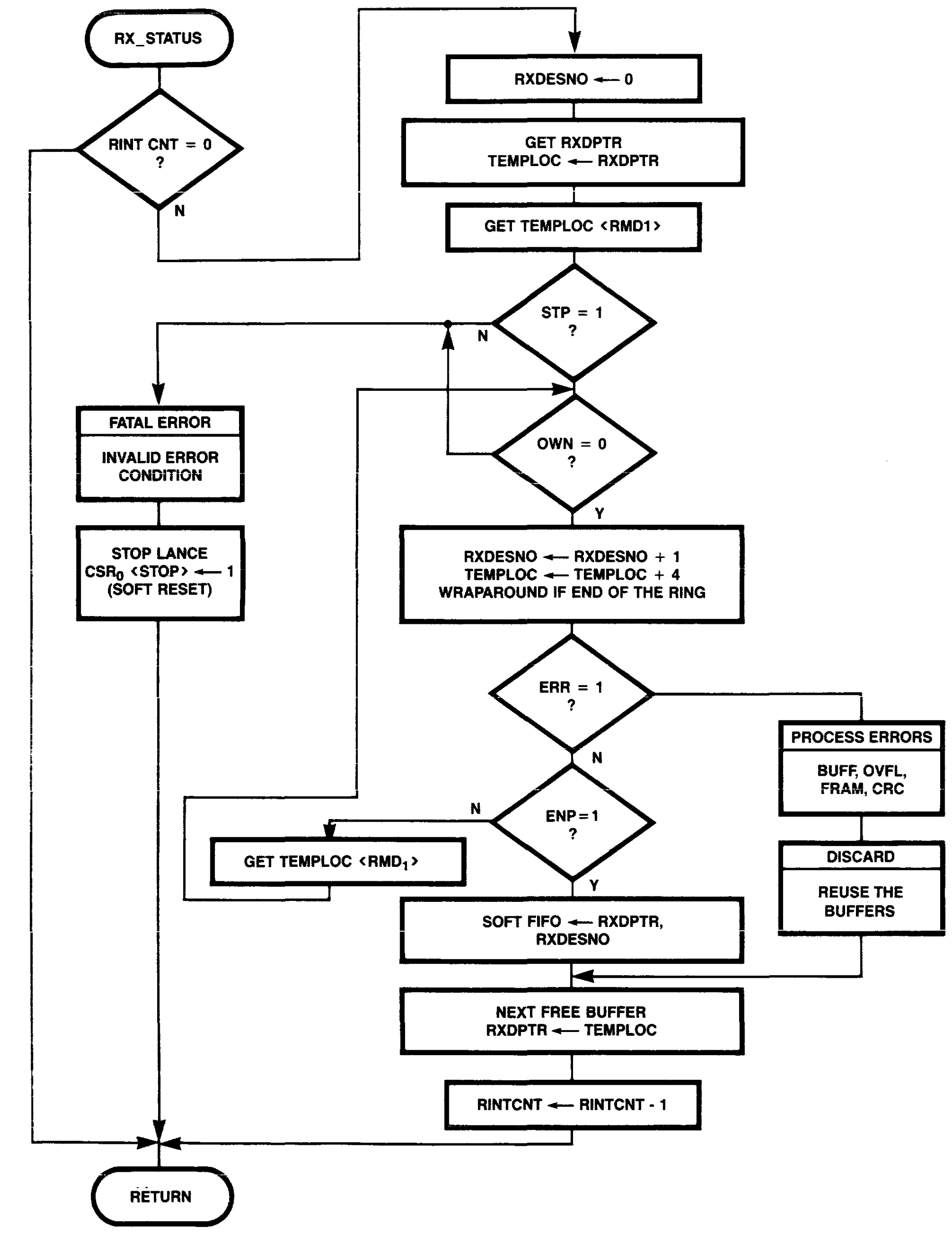
When the LANCE interrupts the CPU due to RINT, packet reception, the user should clear the source of interrupt (here RINT bit) so that it can distinguish the arrival of the next packet from the current one when RINT bit is set again.

The minimum time (worst case) for the RINT bit to be set again is as follows: At the end of a packet reception from the cable, the Silo has 48 bytes for the LANCE to transfer into a buffer before issuing a RINT. This transfer requires three DMA sequences of 4.8 us each plus two bus dwell times of .7 us. each (bus request response time, $\overline{HOLD}$ to $\overline{HLDA}$, is ignored) for a total of 15.8 us. However, the transfer does not start until after the interpacket gap time of 9.6 us followed by 6.4 us of preamble time. The transfer starts just as the data from the next packet starts to enter the Silo. In this case, the RINT bit is set 15.8 us after the next packet starts to fill the Silo. If the worst case happens and the Silo overflows, another RINT interrupt is issued for the error condition. The time required for the second RINT to occur is 22.6 us ($[48 \times 0.8] - 15.8$). This is the interrupt latency of the LANCE for RINT bit to get set again.

This worst case interrupt latency only applies if the LANCE is not assured of prompt response to DMA bus requests. If the LANCE DMA bus latency is eight us, and the LANCE requests the bus as soon as eight words are in the Silo, there will be a maximum of thirteen words in the Silo when the end of the packet arrives. The LANCE takes 4.8 us to transfer the first eight words and 6.5 us ($5 \times (.6 + .7)$) to transfer the remaining five words, one word at a time for a total of 11.3 us before the first RINT is set.

The packet intergap time is 9.6 us and the preamble is 6.4 us for a total of 16 us before data from the next packet starts to fill the empty Silo. If the LANCE now fails to acquire the DMA bus, the Silo will overflow causing a second RINT 43.1 us after the first RINT ($16 - 11.3 + 48 \times .8$). This is the interrupt latency under these more reasonable conditions.

The errors associated with packet reception are BUFF, OVFL, CRC, and FRAM errors. The packets are normally discarded when any of these errors occur. Note that there may be an OVFL error when there is a BUFF error.



RINTCNT =	No. of RINT interrupts
RXDPTR =	Pointer to the first descriptor of the packet received by the LANCE and not processed by the CPU
RXDESNO =	No. of descriptors for the received packet
Soft FIFO =	A software FIFO shared by the host and the node processor. It contains two entries per packet, RXDPTR, RXDESNO has write only by the node processor and read only by the host CPU.

Figure 4-7 Receive Packet Status Processing

06363A-17

4.9 STATUS MONITORING AND ERROR PROCESSING

CSR_0_Status routine shown in Figure 4-8 monitors a software FIFO used by the interrupt handler to save the contents of CSR_0 after each interrupt (Figure 4-3). Each CSR_0 record is checked for IDON, RINT, TINT, and ERR bits. If the ERR bit is set, it calls another routine, ERR_CSR_0 (Figure 4-9), to process the fatal and non-fatal errors.

The IDON bit, when set, indicates that the initialization of LANCE has been done.

The counter variable, RINTCNT is incremented whenever RINT is set. RINTCNT is monitored by the routine, RX-Status (Figure 4-7). RX-Status uses RINTCNT to determine if any receive packets need to be processed. The value of RINTCNT corresponds to the number of packets which have been received by the LANCE but have not been processed by the CPU. Similarly, TX-Status routine uses TINTCNT to be able to process the results of the transmitted packets in the order of transmission.

TINTCNT corresponds to the number of packets to be processed by the CPU which have been transmitted by the LANCE. The counter variable, TINTCNT is incremented whenever the TINT bit is set. TINTCNT is monitored by the routine, TX-Status (Figure 4-6).

ERR_CSR_0 routine shown in Figure 4-9 illustrates the typical actions which users would normally take when processing the errors. The errors, MERR and BABL should be recognized as fatal errors. CERR and MISS are considered non-fatal errors.

MERR error indicates that the LANCE did not receive any acknowledgement from the system ($\overline{\text{READY}}$ High) 25.6 us after the start of the memory cycle. The problem can be the result of the LANCE accessing an invalid address (wrong buffer pointers might have been given to the LANCE in the descriptors) or it may be a memory interface problem between the LANCE, CPU, and memory.

BABL error occurs after the LANCE starts loading the 1519th byte to the SILO (maximum Ethernet packet is 1518 bytes). The LANCE continues to send the remainder of the packet following 1518 bytes. When this error occurs, the user should stop the transmission by setting the Stop bit in CSR_0. It is the user's responsibility not to have the packet more than 1514 bytes (1514 if CRC is generated by the LANCE, DTCR=0; 1518 if CRC is to be generated by the user, DTCR=1). It is worthwhile to mention that the LANCE can send up to 4K bytes (BCNT=12 bits in TMD_2) for each packet where it sets the BABL error following 1518 bytes.

CERR (collision, heartbeat, or SQE Test error) is a transceiver test feature. During this test, CERR is set if the LANCE does not see the collision within 2.0 us after a chip initiated transmission. Most transceivers (IEEE 802.3/Ethernet compatible) have the SQE Test feature built in. Within some delay (less than 2.0 us) following a transmission, they send a 10.0 MHz signal on collision pair. In the LANCE/SIA application, the SIA translates the 10.0 MHz differential signal to a TTL signal on CLSN line to the LANCE. CERR error does not cause an interrupt. The user can normally check CERR when the TINT bit gets set following transmission of the next packet. When the LANCE/SIA is interfaced with a transceiver which does not have the SQE Test function, the user simply ignores the existence of the CERR bit. Typical application where SQE Test may not be desirable is in repeater designs.

4.9.1 MISS ERROR

In the LANCE Rev. B, when a MISS error is set in CSR_0, it must be cleared immediately, or the next packets addressed to this node will not be received, even if some receive buffers have been relinquished to the LANCE. The LANCE Rev. C does not require the MISS bit to be cleared to receive the following packets.

4.9.2 RECEIVER and/or TRANSMITTER TURN OFF DUE to ERROR

There are three types of error which cause the transmitter to be turned off (CSR_0, TXON=0):

(1) Memory error (MERR)
(2) Underflow (UFLO)
(3) Buffer error (BUFF)

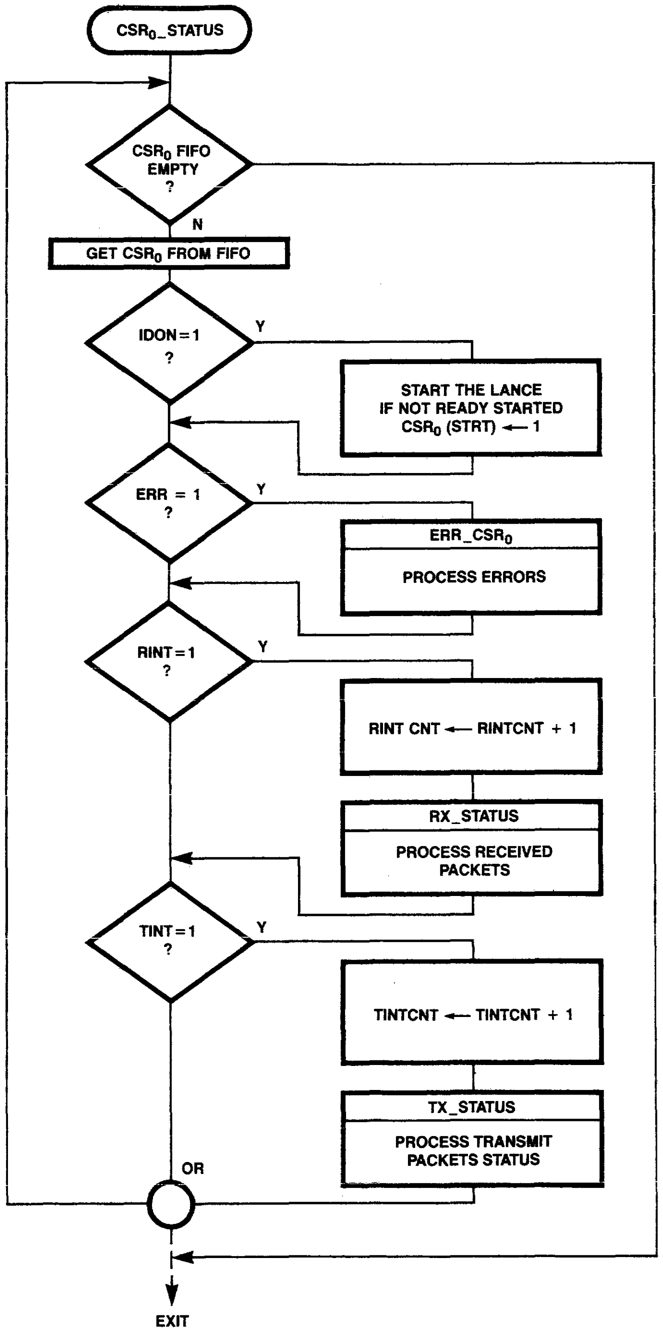
The only type of error which causes the receiver to be turned off (CSR_0, RXON=0) is a memory error (MERR). The chip has to be restarted in order to turn the transceiver and/or receiver back on again. The easiest way to restart the LANCE is by setting the Stop bit and then setting the Start bit in CSR_0. This turns the receiver and transmitter on again because DRX and DTX are still cleared in the Mode Register. However, setting the Stop bit may cause some confusion for the user if the LANCE is in the middle of a transmission or reception or if the buffers are data chained.

It is recommended that the LANCE be re-initialized, rather than setting the Start bit when it stops. Care must be applied when stopping the LANCE because setting the STOP bit clears CSR_3. CSR_3 must be reprogrammed if it contained a non-zero value before the LANCE was stopped. The LANCE can also be re-initialized by setting the INIT bit, but this is not necessary unless a change of operating parameters is desired.

4.10 PROGRAMMER ORGANIZATION OF THE LANCE REGISTER INFORMATION

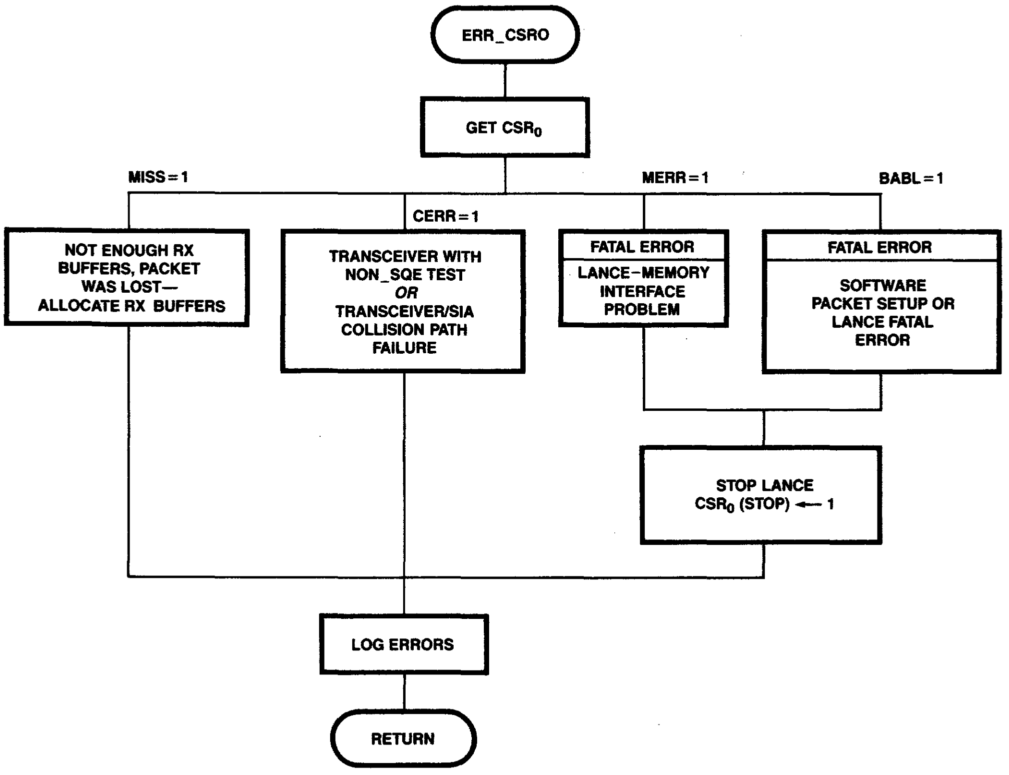
The 8086 System register organization is different than the organization of 68000 registers. Therefore the programmer needs to know the organization of the system to be used with the LANCE so that the Initialization Block information and the Descriptor Ring entries are stored correctly.

In a 16-bit word in the 68000 system, byte 0 consists of bits 8 thru 15 (the most significant bits) and byte 1 consists of bits 0 thru 7. The word, AD 80, in 68000 assembly code is stored in memory as AD 80. This is the correct order of bytes for the LANCE control registers.



06363A-18

Figure 4-8 CSR_0 Status Processing


ERR_CSRO
GET CSR0
MISS=1
CERR=1
MERR=1
BABL=1
NOT ENOUGH RX BUFFERS, PACKET WAS LOST— ALLOCATE RX BUFFERS
TRANSCEIVER WITH NON_SQE TEST OR TRANSCEIVER/SIA COLLISION PATH FAILURE
FATAL ERROR
LANCE-MEMORY INTERFACE PROBLEM
FATAL ERROR
SOFTWARE PACKET SETUP OR LANCE FATAL ERROR
STOP LANCE CSR0 (STOP) ← 1
LOG ERRORS
RETURN

06363A-19

Figure 4-9 CSR$_0$ Error Bit Processor, ERR-CSR0

However, data must be serialized for transmission and serialized data is handled as bytes rather than words. Serial byte data in ascending order is stored in memory as byte 1 (bytes 0–7) followed by byte 0 (bits 8–15). In serializing the data for transmission, the byte in the low-order bit positions is sent first. Therefore, the 68000 data being transmitted or received must be byte-swapped. The LANCE has a byte-swapping mechanism to perform this function. It is a control bit called BSWP in the Control and Status Register 3. When it is set to 1, the LANCE swaps the high and low bytes on DMA data transfers between the Silo and memory.

Figures 4-10, 4-11, and 4-12 show the Initialization Block and Descriptor Ring organization of control information required in a 68000 program for proper functioning in the LANCE. Figure 4-16 shows the relationship between control data in a 68000 register, in memory, and in a LANCE register. Figure 4-17 shows the relationship of serialized data and memory with and without byte swapping.

The 8086 stores sixteen-bit words with the low order byte (byte 0, bits 0–7) on an even byte boundary and byte 1 (the most significant eight bits) on the next higher byte address. For example, a data word stored in an 8086 register or in a program as AD 80, is stored in memory as 80 AD.

The LANCE stores word data in its control registers in the same order as it is stored in memory. It is the programmer's responsibility to store control information in the order needed by the LANCE. Figures 4-13, 4-14, and 4-15 give an 8086 programmer's view of the LANCE Initialization Block and Descriptor Ring information. Figure 4-16 shows the relationship between control data in an 8086 register, in memory, and in a LANCE register.

Because of the order that the 8086 stores bytes in memory during word transfers, no byte swapping of transmitted or received data is required when an 8086 system is used with the LANCE. The BSWP control bit is set to zero.

Word Address	Byte 0								Byte 1								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bits
IADR + 0	PROM	Reserved								INTL	DRTY	COLL	DTCR	LOOP	DTX	DRX	Mode Word
IADR + 2	PADR (bits 15:8)								PADR (bits 7:0)								Physical Address Register
IADR + 4	PADR (bits 31:24)								PADR (bits 23:16)								
IADR + 6	PADR (bits 47:40)								PADR (bits 39:32)								
IADR + 8	LADRF (bits 15:8)								LADRF (bits 7:0)								Logical Address Filter
IADR + 10	LADRF (bits 31:24)								LADRF (bits 23:16)								
IADR + 12	LADRF (bits 47:40)								LADRF (bits 39:32)								
IADR + 14	LADRF (bits 63:56)								LADRF (bits 55:48)								
IADR + 16	RDRA (bits 15:8)								RDRA (bits 7:0)								Receive Descriptor Ring Pointer
IADR + 18	RLEN (31:29)			Reserved					RDRA (bits 23:16)								
IADR + 20	TDRA (bits 15:8)								TDRA (bits 7:0)								Transmit Descriptor Ring Pointer
IADR + 22	TLEN (31:29)			Reserved					TDRA (bits 23:16)								

06363A-20

Figure 4-10. Initialization Block Organization in 68000 Assembly Code

Word Address	Byte 0								Byte 1	
	15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0	Bits
RDRA + 0	RBADR (bits 15:8)								RBADR (bits 7:0)	
RDRA + 2	OWN	ERR	FRAM	OFLO	CRC	BUFF	STP	ENP	RBADR (bits 23:16)	
RDRA + 4	1	1	1	1	BCNT (bits 11:8)				BCNT (bits 7:0)	
RDRA + 6	Reserved				MCNT (bits 11:8)				MCNT (bits 7:0)	

06363A-21

Figure 4-11. Receive Descriptor Ring Entry Organization in 68000 Assembly Code

Word Address	Byte 0								Byte 1	
	15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0	Bits
TDRA + 0	TBADR (bits 15:8)								TBADR (bits 7:0)	
TDRA + 2	OWN	ERR	Res	MORE	ONE	DEF	STP	ENP	TBADR (bits 23:16)	
TDRA + 4	1	1	1	1	BCNT (bits 11:8)				BCNT (bits 7:0)	
TDRA + 6	BUFF	UFLO	RES	LCOL	LCAR	RTRY	TDR	(9:8)	TDR (bits 7:0)	

06363A-22

Figure 4-12. Transmit Descriptor Ring Entry Organization in 68000 Assembly Code

Word Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bits
	Byte 1								Byte 0								
IADR + 0		INTL	DRTY	COLL	DTCR	LOOP	DTX	DRX	PROM	Reserved							Mode Word
IADR + 2	PADR (bits 7:0)								PADR (bits 15:8)								Physical Address Register
IADR + 4	PADR (bits 23:16)								PADR (bits 31:24)								
IADR + 6	PADR (bits 39:32)								PADR (bits 47:40)								
IADR + 8	LADRF (bits 7:0)								LADRF (bits 15:8)								Logical Address Filter
IADR + 10	LADRF (bits 23:16)								LADRF (bits 31:24)								
IADR + 12	LADRF (bits 39:32)								LADRF (bits 47:40)								
IADR + 14	LADRF (bits 55:48)								LADRF (bits 63:56)								
IADR + 16	RDRA (bits 7:0)								RDRA (bits 15:8)								Receive Descriptor Ring Pointer
IADR + 18	RDRA (bits 23:16)								RLEN (31:29)			Reserved					
IADR + 20	TDRA (bits 7:0)								TDRA (bits 15:8)								Transmit Descriptor Ring Pointer
IADR + 22	TDRA (bits 23:16)								TLEN (31:29)			Reserved					

06363A-23

Figure 4-13. Initialization Block Organization in 8086 Assembly Code

Word Address	Byte 1: 15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0	Bits
		Byte 0								
RDRA + 0	RBADR (bits 7:0)	RBADR (bits 15:8)								
RDRA + 2	RBADR (bits 23:16)	OWN	ERR	FRAM	OFLO	CRC	BUFF	STP	ENP	
RDRA + 4	BCNT (bits 7:0)	1	1	1	1	BCNT (bits 11:8)				
RDRA + 6	MCNT (bits 7:0)	Reserved				MCNT (bits 11:8)				

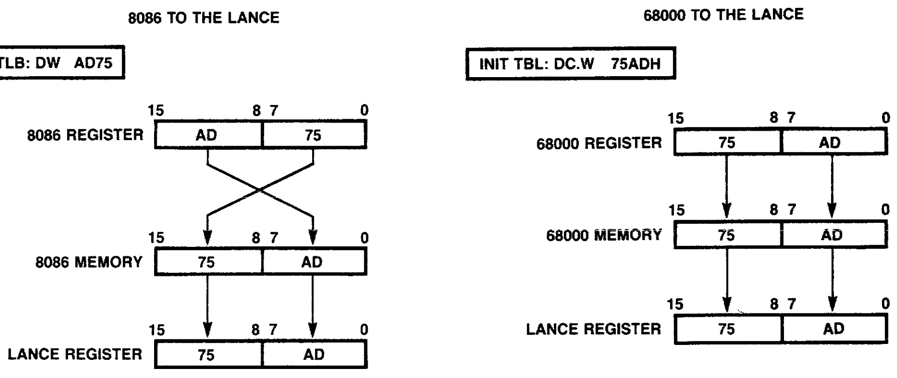
06363A-24

Figure 4-14. Receive Descriptor Ring Entry Organization in 8086 Assembly Code

Word Address	Byte 1: 15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0	Bits
		Byte 0								
TDRA + 0	TBADR (bits 7:0)	TBADR (bits 15:8)								
TDRA + 2	TBADR (bits 23:16)	OWN	ERR	RES	MORE	ONE	DEF	STP	ENP	
TDRA + 4	BCNT (bits 7:0)	1	1	1	1	BCNT (bits 11:8)				
TDRA + 6	TDR (bits 7:0)	BUFF	UFLO	RES	LCOL	LCAR	RTRY	TDR (9:8)		

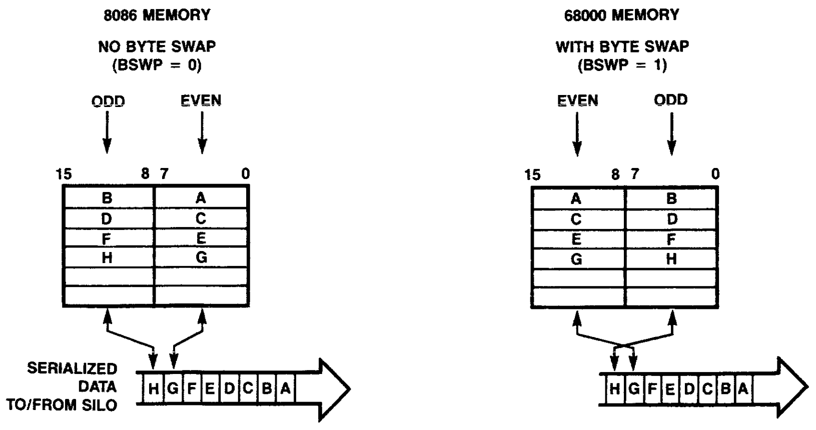
06363A-25

Figure 4-15. Transmit Descriptor Ring Entry Organization in 8086 Assembly Code



06363A-26

Figure 4-16 Control Data Transfers Between the LANCE and CPU



06363A-27

Figure 4-17 Serial Data Storage in Memory

APPENDIX A
GLOSSARY

ACON	ALE Control bit in CSR$_3$
ADR	Register Data/Address Port Select
ALE	Address Latch Enable output pin on the LANCE
AS	Address Strobe output for use by slave device
BABL	Error flag in CSR$_0$ (more than 1518 bytes in one packet)
BCNT	Buffer Byte Count entries in RMD$_2$ and TMD$_2$
BCON	Byte Control bit in CSR$_3$ for Byte Mask and Hold I/O pins
BMo/BYTE	Input/output select
BSWP	Byte Swap bit in CSR$_3$
BUFF	Buffer Error flag in RMD$_1$ and TMD$_3$
BUSAKO	Bus request output in daisy chain configuration of the LANCE. If the LANCE is requesting the bus when it receives $\overline{\text{HLDA}}$, $\overline{\text{BUSAKO}}$ remains high. Otherwise, $\overline{\text{BUSAKO}}$ is set low when $\overline{\text{HLDA}}$ arrives.
BUSRQ	Bus Request output pin on the LANCE used in daisy chain configuration (CSR$_3$ BCON = 1).
Bus Bandwidth	Percentage of time a device holds and controls the DMA bus.
Bus Latency	Allowable bus latency is the time a device can wait for the DMA bus after requesting it without the loss of data or other functions.
Bus Master	When a device has access to a DMA bus, it is a bus master.
BM	Byte Mask bits on the LANCE indicating which byte(s) on the DAL lines are to be read or written
CERR	Collision Error flag in CSR$_0$
CLSN	Collision logical input to the LANCE indicating the presence of a 10.0 MHz differential signal in the collision pairs (collision ±) at the transceiver interface cable
COLL	Force Collision bit in the Mode Register in the LANCE
CRC	Cyclic Redundancy Check flag in RMD$_1$
CS	Chip Select input when asserted puts the LANCE into bus slave mode
CSR	Control and Status Registers in the LANCE
DAL	Data/Address Line pins on the LANCE
DALI	Data/Address Lines input control
DALO	Data/Address Lines output control
DAS	Data Strobe is an input/output pin on the LANCE used to distinguish the data portion of a bus transfer from the address portion.
DEF	Deferred flag in TMD$_1$. Transmission deferred
DMA	Direct Memory Access
DMA Cycle	The time required for the LANCE to transfer one word over the DMA bus (600 ns plus any wait states). In the burst mode, the LANCE transfers eight words in 4.8 us each time it acquires the DMA bus.
DRTY	Disable Retry flag in Moter (allows only one transmission attempt)
DRX	Disable the Receiver bit in Mode Register
DS	Data Strobe on the Z8001
DTCR	Disable Transmit CRC bit in Mode Register
DTX	Disable Transmitter bit in Mode Register
Daisy-chain	A method of connecting bus master devices to control interrupt priorities.
Dwell time	The time interval between the LANCE's release of the bus to the next bus request. It is the time interval that the $\overline{\text{HOLD}}$ line remains inactive.
ENP	End of Packet flag in RMD$_1$ and TMD$_1$
ERR	Error flag in descriptor rings, RMD$_1$ and TMD$_1$ and in Am7990 Control and Status Register CSR$_0$ ("or" of BABL, CERR, MISS, and MERR)
FIFO	First In First Out data buffer (Silo)
FRAM	Framing Error flag in RMD$_1$
HADR	High Order 8 Address Bits of buffer described in this entry of RMD$_1$ or TMD$_1$
HLDA	Hold Acknowledge input to the LANCE makes it the bus master.
HOLD	Bus Hold Request output asserted by the LANCE when it requires the bus.
IADR	Low order 16 bits of address of Initialization Block in CSR$_1$, high order 8 bits in CSR$_2$.
IDON	Status bit in CSR$_0$ indicates initialization has been done
INEA	Interrupt Enable bit in CSR$_0$ enables the LANCE INTR output.
INIT	Initialize flag in CSR$_0$ in the LANCE
INT	Interrupt
INTL	Internal Loopback flag in the Mode Register
INTR	Interrupt flag in CSR$_0$. Active when one or more of the CSR$_0$ status flags, BABL, IDON, MERR, MISS, RINT, or TINT, are set.
ISO	International Standards Organization
LADR	Low Order 16 Address Bits of buffer of this descriptor in RMD$_0$ and TMD$_0$
LADRF	Logical Address Filter in Initialization Block. 64-bit mask for logical addresses
LANCE	Local Area Network Controller for Ethernet (Am7990)

LCAR	Loss of Carrier flag in TMD_3
LCOL	Late Collision flag in TMD_3
Latency	See Bus Latency
LOOP	Mode Register bit to control loopback for test
MCNT	Message Byte Count entry in RMD_3
MERR	Memory error flag in CSR_0 set when the LANCE as a bus master has not received READY within 25.6 us after asserting the address on the DAL lines.
MISS	Missed Packet flag in CSR_0 set when the receiver loses a packet because no buffers are available.
MORE	More than one entry needed (to transmit packet) flag in TMD_1
OFLO	Overflow Error flag in RMD_1. Received data lost.
ONE	Exactly One Entry Needed (to transmit packet) flag in TMD_1
OWN	Descriptor entry owner (host/chip) flag in RMD_1 and TMD_1
PADR	48-bit Physical Address assigned to chip (entry in Initialization Block)
PAL	Programmable Array Logic
PROM	Promiscuous mode (accepts all incoming packets) flag in Mode Register
RAM	Random Access Memory
RAP	Register Address Port (bus addressable register) in LANCE
RAS	Register Address Select
RDP	Data Port (bus addressable register) in the LANCE
RDRA	Receive Descriptor Ring base address entry in Initialization Block
READ	Input/output pin on the LANCE. High in Bus Master mode is input to read DAL lines, low means the LANCE has placed data on the DAL lines. The meaning is reversed in the Bus Slave mode.
RENA	Receive Enable input to the LANCE. Indicates presence of carrier on the channel.
RES	Reserved
RESET	Reset (Am7990 input) stops the LANCE operation, clears its internal logic (CSR_0 and CSR_3), and sets it into idle state with STOP bit set in CSR_0.
RINT	Receiver Interrupt flag in CSR_0 set after the LANCE has received a packet and updated the Receive Descriptor Ring. Also set if the reception is stopped due to a failure.
RLEN	Receive Ring Length
RQ/GT	Request/Grant bus exchange handshake
RTRY	Retry Error flag in TMD_3 (transmitter failed in 16 attempts)
RW	Read/Write Select
RX	Receive Input
RXON	Receiver On flag in CSR_0
Runt	A packet that is less than 64 bytes long
RX	Receive input bit stream to the LANCE
SIA	Serial Interface Adapter
STP	Start of Packet flag in TMD_1 and RMD_1
STRT	Start bit in CSR_0
Silo	A 48-byte FIFO memory in the LANCE used to transfer data between the local memory and the serial interface adapter
Slave Mode	The slave mode of the LANCE (CS asserted) is used to initialize the LANCE.
STOP	Control bit in CSR_0 when set stops all external activity of the LANCE
T-state	Timing cycle (100 ns) in the LANCE. One LANCE DMA cycle consists of six T-states plus any required wait states.
TDMD	Transmit Demand for DMA bus. Control bit in CSR_0 to bypass DMA wait time
TDR	Time Domain Reflectometer flag in TMD_3 in the LANCE
TDRA	Transmit Descriptor Ring base address in Initialization Block
TENA	Transmit Enable output on the LANCE used to enable external transmit logic
TINT	Transmitter Interrupt flag in CSR_0 set after the LANCE has completed sending a packet and has updated the Transmit Descriptor Ring. Also set if the transmission is stopped due to a failure.
TLEN	Transmit Ring Length
TMD	Transmit Message Descriptor entry in Transmit Descriptor Ring
TX	Transmit (output)
TXON	Transmitter On flag in CSR_0
TX	Transmit output bit stream on the LANCE
UFLO	Underflow Error flag in TMD_3

APPENDIX B

8086 program for multicast addressing and hash filter.

```
                     ;          SUBROUTINE TO SET A BIT IN THE HASH FILTER FROM A
                     ;          GIVEN ETHERNET LOGICAL ADDRESS
                     ;          ON ENTRY SI POINTS TO THE LOGICAL ADDRESS WITH LSB FIRST
                     ;                   DI POINTS TO THE HASH FILTER WITH LSB FIRST
                     ;          ON RETURN SI POINTS TO THE BYTE AFTER THE LOGICAL ADDRESS
                     ;                   ALL OTHER REGISTERS ARE UNMODIFIED
                     ;
                                PUBLIC  SETHASH
                                ASSUME  CS:CSE61
                     ;
 = 1DB6              POLYL      EQU     1DB6H       ;CRC POLYNOMINAL TERMS
 = 04C1              POLYH      EQU     04C1H
                     ;
0000                 CSE61      SEGMENT PUBLIC 'CODE'
                     ;
0000                 SETHASH    PROC    NEAR
0000 50                         PUSH    AX          ;SAVE ALL REGISTERS
0001 53                         PUSH    BX
0002 51                         PUSH    CX
0003 52                         PUSH    DX
0004 55                         PUSH    BP
                     ;
0005 B8 FFFF                    MOV     AX,0FFFFH   ;AX,DX = CRC ACCUMULATOR
0008 BA FFFF                    MOV     DX,0FFFFH   ;PRESET CRC ACCUMULATOR TO ALL 1'S
000B B5 03                      MOV     CH,3        ;CH = WORD COUNTER
                     ;
000D 8B 2C           SETH10:    MOV     BP,[S1]     ;GET A WORD OF ADDRESS
000F 83 C6 02                   ADD     SI,2        ;POINT TO NEXT ADDRESS
0012 B1 10                      MOV     CL,16       ;CL = BIT COUNTER
                     ;
0014 8B DA           SETH20:    MOV     BX,DX       ;GET HIGH WORD OF CRC
0016 D1 C3                      ROL     BX,1        ;PUT CRC31 TO LSB
0018 33 DD                      XOR     BX,BP       ;COMBINE CRC31 WITH INCOMING BIT
001A D1 E0                      SAL     AX,1        ;LEFT SHIFT CRC ACCUMULATOR
001C D1 D2                      RCL     DX,1
001E 81 E3 0001                 AND     BX,0001H    ;BX = CONTROL BIT
0022 74 07                      JZ      SETH30      ;DO NOT XOR IF CONTROL BIT = 0
                     ;
                     ;          PERFORM XOR OPERATION WHEN CONTROL BIT = 1
                     ;
0024 35 1D86                    XOR     AX,POLYL
0027 81 F2 04C1                 XOR     DX,POLYH
                     ;
002B 0B C3           SETH30:    OR      AX,BX       ;PUT CONTROL BIT IN CRC0
002D D1 CD                      ROR     BP,1        ;ROTATE ADDRESS WORD
002F FE C9                      DEC     CL          ;DECREMENT BIT COUNTER
0031 75 E1                      JNZ     SETH20
0033 FE CD                      DEC     CH          ;DECREMENT WORD COUNTER
0035 75 D6                      JNZ     SETH10
                     ;
                     ;          FORMATION OF CRC COMPLETE, AL CONTAINS THE REVERSED HASH
                                CODE
                     ;
0037 B9 000A                    MOV     CX,10
003A D0 E0           SETH40:    SAL     AL,1        ;REVERSE THE ORDER OF BITS IN AL
003C D0 DC                      RCR     AH,1        ;AND PUT IT IN AH
003E E2 FA                      LOOP    SETH40
```

```
                              ;
                              ;         AH NOW CONTAINS THE HASH CODE
                              ;
0040 8A DC                              MOV     BL,AH          ;BL = HASH CODE, BH IS ALREADY ZERO
0042 B1 03                              MOV     CL,3           ;DIVIDE HASH CODE BY 8
0044 D2 EB                              SHR     BL,CL          ;TO GET TO THE CORRECT BYTE
0046 B0 01                              MOV     AL,01H         ;PRESET FILTER BIT
0048 80 E45 07                          AND     AH,7H          ;EXTRACT BIT COUNT
004B 8A CC                              MOV     CL,AH
004D D2 E0                              SHL     AL,CL          ;SHIFT BIT TO CORRECT POSITION
004F 08 01                              OR      [DI + BX],AL   ;SET IN HASH FILTER
0051 5D                                 POP     BP
0052 5A                                 POP     DX
0053 59                                 POP     CX
0054 5B                                 POP     BX
0055 58                                 POP     AX
0056 C3                                 RET
                              ;
0057                          SETHASH   ENDP
                              ;
0057                          CSEG1     ENDS
                              ;
                                        END
```

Basic computer program example to generate the hash filter, for multicast addressing, in the LANCE.

```
100  REM
110  REM   PROGRAM TO GENERATE A HASH NUMBER GIVEN AN ETHERNET ADDRESS
120  REM
130  DEFINT A-Z
140  DIM A(47) : REM ETHERNET ADDRESS = 48 BITS
150  DIM C(32) : REM CRC REGISTER = 32 BITS
160  PRINT "ENTER STARTING ADDRESS'; : INPUT A$
170  IF LEN (A$) < > 12 THEN 160 : REM THE INPUT ADDRESS STARTING MUST BE 12 CHARS
180  REM
190  REM   UNPACK STARTING ADDRESS INTO ADDRESS ARRAY
200  REM
210  M = 0
220  FOR I = 0 TO 47 : A(I) = 0 : NEXT I
230  FOR N = 12 TO 1 STEP -1
240  Y$ = MID$ (A$,N,1)
250  IF Y$ = "0" THEN 420
260  IF Y$ = "1" THEN A(M) = 1 : GOTO 420
270  IF Y$ = "2" THEN A(M + 1) = 1 : GOTO 420
280  IF Y$ = "3" THEN A(M + 1) = 1 : A(M) = 1 : GOTO 420
290  IF Y$ = "4" THEN A(M + 2) = 1 : GOTO 420
300  IF Y$ = "5" THEN A(M + 2) = 1 : A(M) = 1 : GOTO 420
310  IF Y$ = "6" THEN A(M + 2) = 1 : A(M + 1) = 1 : GOTO 420
320  IF Y$ = "7" THEN A(M + 2) = 1 : A(M + 1) = 1 : A(M) = 1 : GOTO 420
330  A(M + 3) = 1
340  IF Y$ = "8" THEN 420
350  IF Y$ = "9" THEN A(M) = 1 : GOTO 420
360  IF Y$ = "A" THEN A(M + 1) = 1 : GOTO 420
370  IF Y$ = "B" THEN A(M + 1) = 1 : A(M) = 1 : GOTO 420
380  IF Y$ = "C" THEN A(M + 2) = 1 : GOTO 420
390  IF Y$ = "D" THEN A(M + 2) = 1 : A(M) = 1 : GOTO 420
400  IF Y$ = "E" THEN A(M + 2) = 1 : A(M + 1) = 1 : GOTO 420
410  IF Y$ = "F" THEN A(M + 2) = 1 : A(M + 1) = 1 : A(M) = 1
420  M = M + 4
430  NEXT N
```

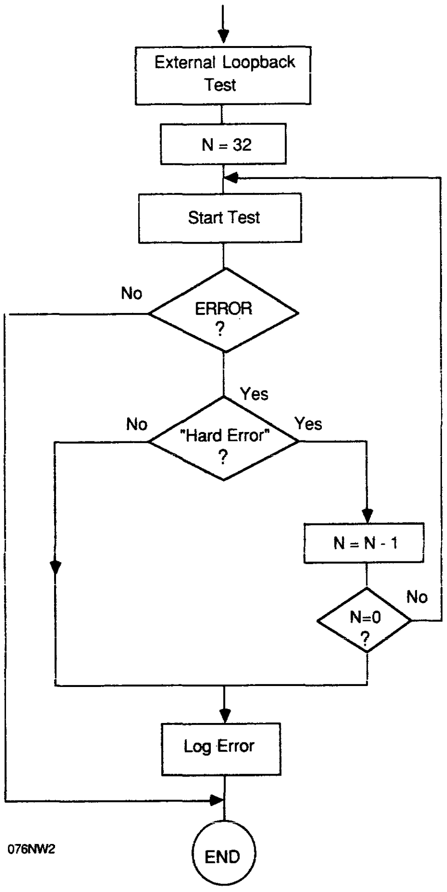
```
440  REM
450  REM   PERFORM CRC ALGORITHM ON ARRAY A(0-47)
460  REM
470  FOR I = 0 TO 31 : C(I) = 1 : NEXT I
480  FOR N = 0 TO 47
490  REM LEFT CRC REGISTER BY 1
500  FOR I = 32 TO 1 STEP -1 : C(I) = C(I - 1) : NEXT I
510  C(0) = 0
520  T = C(32) XOR A(N) : REM T = CONTROL BIT
530  IF T < > THEN 600 : REM JUMP IF CONTROL BIT = 0
540  C(1) = C(1) XOR 1 : C(2) = C(2) XOR 1 : C(4) = C(4) XOR 1
550  C(5) = C(5) XOR 1 : C(7) = C(7) XOR 1 : C(8) = C(8) XOR 1
560  C(10) = C(10) XOR 1 : C(11) = C(11) XOR 1 : C(12) = C(12) XOR 1
570  C(16) = C(16) XOR 1 : C(22) = C(22) XOR 1 : C(23) = C(23) XOR 1
580  C(26) = C(26) XOR 1
590  C(0) = 1
600  NEXT N
610  REM
620  REM   CRC COMPUTATION COMPLETE, EXTRACT HASH NUMBER FROM C(0) TO C(5)
630  REM
640  HH = 32*C(0) + 16*C(1) + 8*C(2) + 4*C(3) + 2*C(4) + C(5)
650  PRINT "THE HASH NUMBER FOR ";A$;" IS ";HH
660  GOTO 160
```

MAPPING OF LOGICAL ADDRESS TO FILTER MASK

LAF Reg Bits Set	LAF Loc	Destination Address Accepted	LAF Reg Bits Set	LAF Loc	Destination Address Accepted
	Dec	(Hex)		Dec	(Hex)
0	0	0000 0000 0085	0	32	0000 0000 0021
	1	0000 0000 00A5		33	0000 0000 0001
L	2	0000 0000 00E5	L	34	0000 0000 0041
A	3	0000 0000 00C5	A	35	0000 0000 0071
F	4	0000 0000 0045	F	36	0000 0000 00E1
	5	0000 0000 0065		37	0000 0000 00C1
0	6	0000 0000 0025	2	38	0000 0000 0081
	7	0000 0000 0005		39	0000 0000 00A1
	8	0000 0000 002B		40	0000 0000 008F
	9	0000 0000 000B		41	0000 0000 00BF
	10	0000 0000 004B		42	0000 0000 00EF
	11	0000 0000 006B		43	0000 0000 00CF
	12	0000 0000 00EB		44	0000 0000 004F
	13	0000 0000 00CB		45	0000 0000 006F
	14	0000 0000 008B		46	0000 0000 002F
15	15	0000 0000 00BB	15	47	0000 0000 000F
0	16	0000 0000 00C7	0	48	0000 0000 0063
	17	0000 0000 00E7		49	0000 0000 0043
	18	0000 0000 00A7		50	0000 0000 0003
	19	0000 0000 0087		51	0000 0000 0023
L	20	0000 0000 0007	L	52	0000 0000 00A3
A	21	0000 0000 0027	A	53	0000 0000 0083
F	22	0000 0000 0067	F	54	0000 0000 00C3
	23	0000 0000 0047		55	0000 0000 00E3
1	24	0000 0000 0069	3	56	0000 0000 00CD
	25	0000 0000 0049		57	0000 0000 00ED
	26	0000 0000 0009		58	0000 0000 00AD
	27	0000 0000 0029		59	0000 0000 008D
	28	0000 0000 00A9		60	0000 0000 000D
	29	0000 0000 0089		61	0000 0000 002D
	30	0000 0000 00C9		62	0000 0000 006D
15	31	0000 0000 00E9	15	63	0000 0000 004D

Appendix C

External Loopback Test Flow Chart



N = Max. number of times to repeat the test.

External Loopback Test Procedure

Due to the problem of Silo Pointer Mis-alignment in the LANCE's External Loopback, the following gives the terminology used and procedures recommended for performing the External Loopback Test.

Terminology:

LANCE Hard Errors:

These can be caused by External Loopback Silo Pointer Mis-alignment (false Hard Error), or they can be real Hard Errors in the network that the software can correct. Examples of real Hard Errors are: LCAR, RTRY, CRC, FRAM, BABL, MISS,OFLO, BUFF.

LANCE Soft Error:

This is a real error. It is not a result of External Loopback Silo Pointer Mis-alignment, and the software must correct. The Soft Error is: CERR.

System Hard Errors:

These errors signal a hardware failure in the system. Examples for this type of error are: MERR, UFLO. These are not caused by External Loopback Silo Pointer Mis-alignment.

Testing Procedure:

When a LANCE Hard Error occurs and the source cannot be determined, repeat the External Loopback Test until it passes; or until a real Hard Error, a Soft Error, or a System Hard Error is found; or until it has continuously failed for a predetermined number of times (N). The error in the last attempt is then logged. If a Soft error, or System Hard Error occurs, an error handling routine will take the proper action and the error is logged.

ADVANCED MICRO DEVICES, INC.
901 Thompson Place
P.O. Box 3453
Sunnyvale,
California 94088-3453
(408) 732-2400
TWX: 910-339-9280
TELEX: 34-6306
TOLL-FREE
(800) 538-8450

APPLICATIONS HOTLINE
(800) 222-9323
(408) 749-5703

Printed in USA

06363C