PERKIN-ELMER

OS/32 NETWORK DRIVERS

Programming Reference Manual

48-079 F00 R00

The information in this document is subject to change without notice and should not be construed as a commitment by The Perkin-Elmer Corporation. The Perkin-Elmer Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license, and it can be used or copied only in a manner permitted by that license. Any copy of the described software must include the Perkin-Elmer copyright notice. Title to and ownership of the described software and any copies thereof shall remain in The Perkin-Elmer Corporation.

The Perkin-Elmer Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Perkin-Elmer.

The Perkin-Elmer Corporation, Data Systems Group, 2 Crescent Place, Oceanport, New Jersey 07757

© 1984 by The Perkin-Elmer Corporation

Printed in the United States of America

TABLE OF CONTENTS

CHAPTERS (Continued)

CHAPTERS (Continued)

APPENDIX

FIGURES

TABLES

PREFACE

This manual is specifically designed for assembly and system level programmers. Its purpose is to familiarize the user with the Ethernet and zero-bit insertion/deletion (ZBID) line drivers.

Chapter 1 is an overview of the ITAM/32 system with references to the Ethernet and ZBID line drivers, which are accessed by the supervisor call 15 (SVC15) instruction and provides the reader with general background information. Chapter 2 consists of general terms and concepts of the zero-bit insertion/deletion data link control (ZDLC), which centralizes on interpretation of the ZDLC frame, control modes and various ZDLC configurations. Chapter 3 contains information dealing with the ZBID line driver. This driver controls line synchronization, data transparency and data blocking for medium- to high-speed communication lines. In addition to a description of its features and various commands, this chapter familiarizes the user with the quad-synchronous adapter (QSA). Chapter 4 focuses on Ethernet, a communications system developed to carry data among locally distributed computer systems. This chapter contains information based on the Ethernet device controller discussing such topics as frames and command operations. Chapter 5 focuses on SVC15 programming of the Ethernet line driver and various command and register accesses. Appendix A is geared toward the user who is familiar with the ZBID line driver. This appendix summarizes the changes made to the ZBID line driver interface and the commands that are either supported or not supported by the Ethernet interface line driver.

For information on the contents of all Perkin-Elmer 32-bit manuals, see the 32-Bit Systems User Documentation Summary.

CHAPTER 1
ACCESSING A LINE DRIVER

1.1 INTRODUCTION

The OS/32 Telecommunications Access Method (ITAM/32) is an optional subsystem of OS/32 that provides a software interface between user programs and data communications networks. ITAM/32 enables the user to access a remote terminal or computer via communications facilities, such as telephone lines and modems, as easily as a locally attached peripheral. There is no difference to the user between local Teletype (TTY) and remote TTY access.

ITAM/32 line drivers provide the more primitive functions that allow the user to tailor a communications system to a particular need. The line drivers interface between the communications adapter and the user program, allowing the user to easily specify the control sequences and data necessary to complete a transmission over a communications line.

The ITAM/32 line driver supervisor call 15 (SVC15) access allows the user program to specify a sequence of control commands and the data required by the control sequence. ITAM/32 line driver access allows the user to support a communications network with maximum efficiency and throughput by providing the capability for the user program to specify buffering techniques, monitor the progress of a control sequence and alter control sequences and their data while in progress.

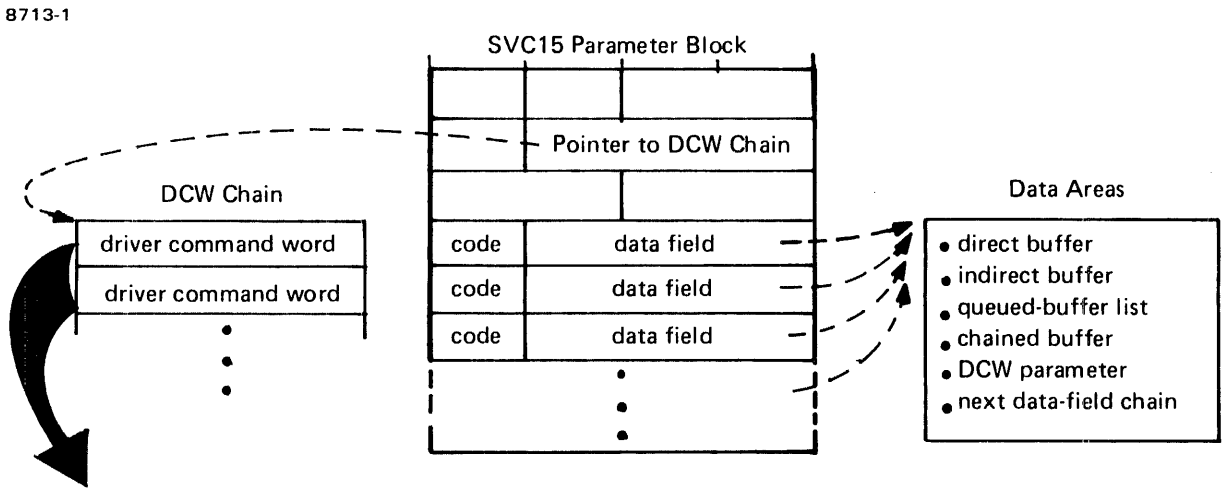
Access to a remote terminal connected to an ITAM/32-supported communications line is through a logical unit (lu). The general procedure for SVC15 line level access is as follows:

1. Assign the communications line to the desired program lu.
2. Issue SVC15 to specify the initial control sequence and data.
3. Use a trap handling routine to monitor the progress of the control sequence.
4. Modify the control sequence and/or data, or issue a subsequent SVC15 specifying a new control sequence and/or data to continue communications as desired.
5. Upon completion, close the lu assigned to the communications line.

Two line drivers, Ethernet and zero-bit insertion/deletion (ZBID), are accessed by the SVC15 instruction. Ethernet provides a simple SVC15 access for normal data transfer and allows a user task (u-task) to access all the features of the Perkin-Elmer Series 3200 Ethernet interface on an individual basis. The ZBID line driver uses quad-synchronous adapter (QSA) device commands to initialize the QSA-to-ZBID mode, performs reads and writes, disables previously issued reads and writes and handles switched lines; it can also use a single synchronous adapter (SSA). For a list of basic differences between the Ethernet and ZBID line drivers, see Appendix A.

1.2 SUPERVISOR CALL 15 (SVC15) ACCESS TO A LINE DRIVER

The standard SVC15 call directly accesses any ITAM/32 line driver. As shown in Figure 1-1, this call is based on the standard SVC15 parameter block, the driver command word (DCW) chain(s) and the data areas associated with the individual DCWs.



Driver Command Word

$\overline{2^{15}}$	$\overline{2^{14}}$	$\overline{2^{13}}$	$\overline{2^{12}}$	$\overline{2^{11}}$ 0	$\overline{2^{10}}$ 0	$\overline{2^{9}}$ 0	$\overline{2^{8}}$ 0	$\overline{2^{7}}$	$\overline{2^{6}}$	$\overline{2^{5}}$	$\overline{2^{4}}$	$\overline{2^{3}}$	$\overline{2^{2}}$	$\overline{2^{1}}$	$\overline{2^{0}}$

BIT	DESCRIPTOR	
2^{15}	Chain Command	Flag bits specifying options in effect for the DCW.
2^{14}	Command Enable	
2^{13}	Buffer Enable	
2^{12}	Timeout	
2^{11}-2^{8}	(reserved)	Line driver commands
2^{7}-2^{3}	Command Modifier	
2^{2}-2^{0}	Command Type	

(See ITAM/32 Reference Manual for detailed descriptions.)

Figure 1-1 U-task Structures for SVC15 Access to a Line Driver

By issuing the SVC15 call, the user activates a line driver. The driver then fetches the DCWs from the DCW chain, along with any associated data area, and executes them. To monitor the progress of the SVC15 call and to provide facilities for dynamic buffer management, the u-task can have a trap generated whenever the line driver:

- starts to execute any DCW,
- starts to use any data area and/or
- terminates the SVC15 call.

These traps enable a task to synchronize its execution with that of the line driver.

In summary, the SVC15 call directly accesses an ITAM/32 line driver for a communications u-task. With this call, the user can specify input/output (I/O) buffering techniques and, concurrently with driver execution, monitor or alter DCWs and their data areas.

Figure 1-2 and its description blocks show the format of the SVC15 parameter block. See the OS/32 Basic Data Communications Reference Manual for detailed information.

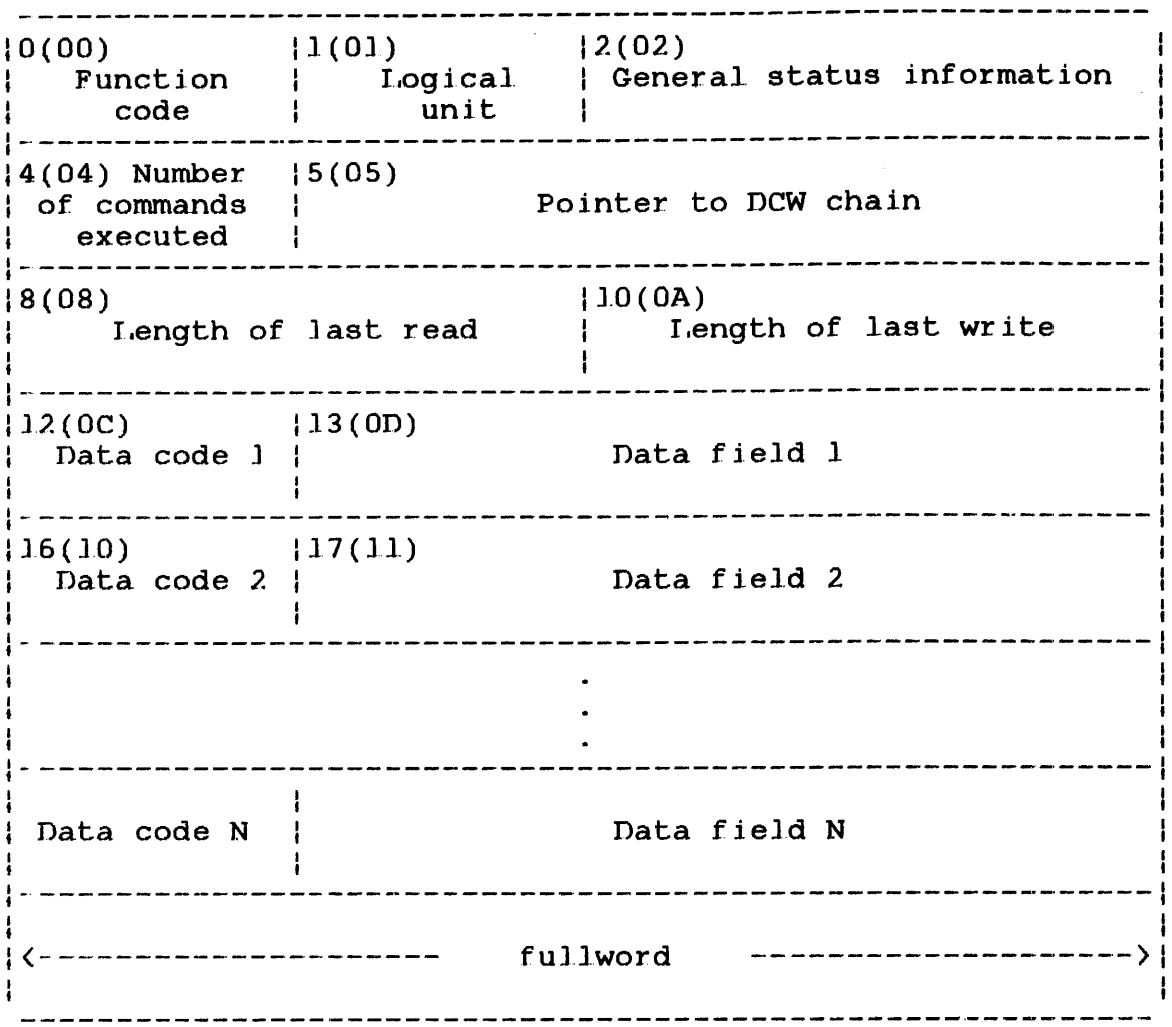


Figure 1-2 SVC15 Parameter Block

Fields:

Function code — specifies general request functions for the SVC15 call. As shown in Table 1-1, function code bit settings allow the user to:

- Halt a current command request.

- Generate a trap when the line driver fetches a new DCW from the DCW chain. OS/32 trap generation is ideally suited for application programs that control communication lines with SVC15 calls. For application details, see the OS/32 System Level Programmer Reference Manual and the OS/32 Basic Data Communications Reference Manual.

- Generate a trap when the SVC15 DCW chain ends.

- Request continuous error processing (CEP). Sections 3.2.2 and 5.2.4 discuss this capability.

Logical unit specifies the logical unit number of the communications line used for the I/O.

General status information holds the SVC15 termination status returned by the operating system. Table 1-2 shows the format for this halfword; Table 3-2 lists the termination codes possible with the status halfword. Most of the encoded termination codes listed in Table 3-2 correspond to those described in the OS/32 Basic Data Communications Reference Manual for all ITAM/32 line drivers. Certain codes, however, are unique to or take specific meaning with the ZBID line driver, especially X'17' (bad sequence), X'20' (idle line) and X'21' (frame abort). Similarly, certain codes take on specific meaning with the Ethernet line driver. See Table 5-2 for more details.

Number of commands executed maintains a 1-byte count for the number of DCWs fetched and executed. When the SVC15 call terminates, this number is returned here.

Pointer to DCW chain contains an address pointing to the first DCW in the DCW chain (see Figure 1-1).

Length of last read is when the SVC15 call terminates and the line driver updates this halfword to reflect the number of bytes transferred during the last read.

Length of last write is when the SVC15 call terminates and the line driver updates this halfword to reflect the number of bytes transferred during the last write.

Data codes holds hexadecimal code identifying the type of data area associated with a DCW (see Figure 1-1).

Data fields holds addresses pointing to the previously mentioned data areas.

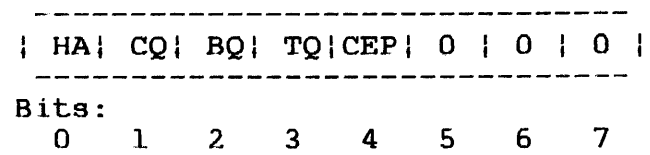


TABLE 1-1 SVC15 FUNCTION CODE BYTE

BIT	HEX MASK	DESCRIPTION
0	80	Halt I/O - If the line driver is connected to the task, this bit halts task I/O. Otherwise, a condition code of 1 (CC=1) is returned in the program status word (PSW) indicating that the line driver was not connected.
1	40	Command Queue Entry Enable - When this bit is set, along with the command enable bit of the DCW, a command queue entry is added to the task queue before the DCW is executed.
2	20	Buffer Queue Entry Enable - When this bit is set, along with the buffer enable bit of the DCW, a buffer queue entry is added to the task queue when the first character of any buffer used by the DCW is processed.
3	10	Termination Queue Entry Enable - This bit causes a termination queue entry to be added to the task queue when the line driver terminates. If the enable SVC15 queue entry bit of the task status word (TSW) is also set, a trap is taken at driver termination.
4	08	CEP - This bit prevents certain error codes from terminating a line driver READ command. Such prevention permits continuous reading of sequential input frames, regardless of any error condition affecting only a single frame.
5-7	07	Reserved; reset to zeros.

ER	BU	TNB	TO		DC		ETC

Bits:

0	1	2	3	4	5	6	9	10	15

TABLE 1-2 GENERAL STATUS INFORMATION HALFWORD

BIT	HEX MASK	DESCRIPTION
0	8000	Error - Set for all error conditions; i.e., for bits 3 and 5 and for most termination codes.
1	4000	Busy - Set whenever the line driver is still busy with the SVC15 call. The SVC15 call can be aborted with the halt I/O bit in the function code.
2	2000	Transfer Not Begun - Set when the line driver is waiting for a starting flag sequence; i.e., the input transfer (read) has not yet begun. This bit is reset once transfer begins. When the ending flag is received, the bit is reset.
3	1000	Time-out - Set if an I/O time-out occurs; i.e., if the error timer expires before a DCW is completely executed.
4	----	N/A
5	0400	Data Check - Set whenever a fixed control store error is detected.
6-9	----	N/A
10-15	003F	Encoded Termination Code - These six bits hold one of the hexadecimal termination codes. The interpretation of these bits depends on the individual driver. See Table 3-2 for the ZBID line driver and Table 5-2 for the Ethernet line driver termination.

CHAPTER 2
ZERO-BIT INSERTION/DELETION DATA LINK CONTROL (ZDLC) PROTOCOL SUPPORT

2.1 INTRODUCTION TO ZERO-BIT INSERTION/DELETION (ZBID)

ZDLC protocol is Perkin-Elmer's implementation of the three bit-synchronous protocols: synchronous data link control (SDLC), high-level data link control (HDLC) and advanced data communication control procedures (ADCCP). The channel terminal manager (CTM) product provides support for the ZDLC protocol at the supervisor call 1 (SVC1) level. See the OS/32 Bit-Synchronous Communications Reference Manual for more information on CTM. The CTM uses the ZBID driver for support of the quad-synchronous adapter/single synchronous adapter (QSA/SSA) hardware. In addition, the user can access the ZBID driver at the SVC15 level that is described later in this chapter.

2.2 INTRODUCTION TO FRAMES

All information transferred between stations in the ZDLC protocol are binary codes conveyed within a formatted ZDLC frame. This frame is the vehicle for all communications on the data line (see Figure 2-1).

8667

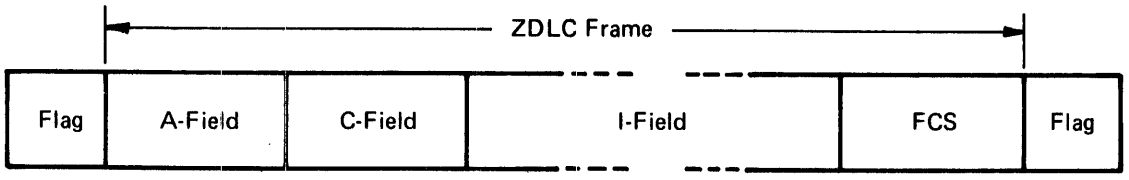


Figure 2-1 Basic ZDLC Frame

The ZDLC frame consists of three required fields and one optional field. The sequence of these fields is fixed and invariable. The address field (A-field), control field (C-field) and frame check sequence (FCS) field are required. The information field (I-field) is an optional, variable-length field. A leading and trailing flag marks the start and end of a ZDLC frame.

The ZBID driver is concerned only with the A-field, and then only when performing a 'selective read'. However, the ZBID driver can be used to transmit data in any format, requiring only the beginning and terminating flag.

2.2.1 Zero-Bit Insertion/Deletion (ZBID)

A ZBID flag is the unique binary sequence, '01111110'. As shown in Figure 2-1, flags mark the start and end of a ZDLC frame. When encountering a ZBID flag, a receiving station interprets subsequent bits as the start of a new ZDLC frame. But what if the '01111110' sequence appears within the I-field, or possibly within the A-, C- or FCS field? Without the ZBID technique, the receiving station would erroneously interpret subsequent bits as the start of a new ZDLC frame.

With the ZBID technique, however, binary coding within a ZDLC frame can be any combination of 0-bits and 1-bits; i.e., this technique ensures data transparency within the frame. Basically, it involves selective insertion and deletion of zeros so that the ZBID flag sequence never occurs within a ZDLC frame.

As shown in Figure 2-2, a station sending a frame performs zero-bit insertion. The station checks all binary bits being sent. Whenever it encounters a '011111' sequence (0-bit followed by five 1-bits), the station inserts a 0-bit. In Figure 2-2, inserted 0-bits are shaded.

A station receiving a ZDLC frame performs zero-bit deletion; i.e., deletion of the zeros inserted by the sending station. The station monitors all bits being received. Whenever the station encounters the '011111' sequence, it inspects the next (seventh) bit. If it is a 0-bit, that 0-bit is deleted and the '011111' sequence is accepted as data. But if the seventh bit is a 1-bit, the next (eighth) bit is immediately inspected. If it is a 0-bit, the '01111110' sequence is accepted as a ZBID flag. If the eighth bit is a 1-bit, the sequence can be "abort" or "idle". These control sequences are discussed in subsequent sections.

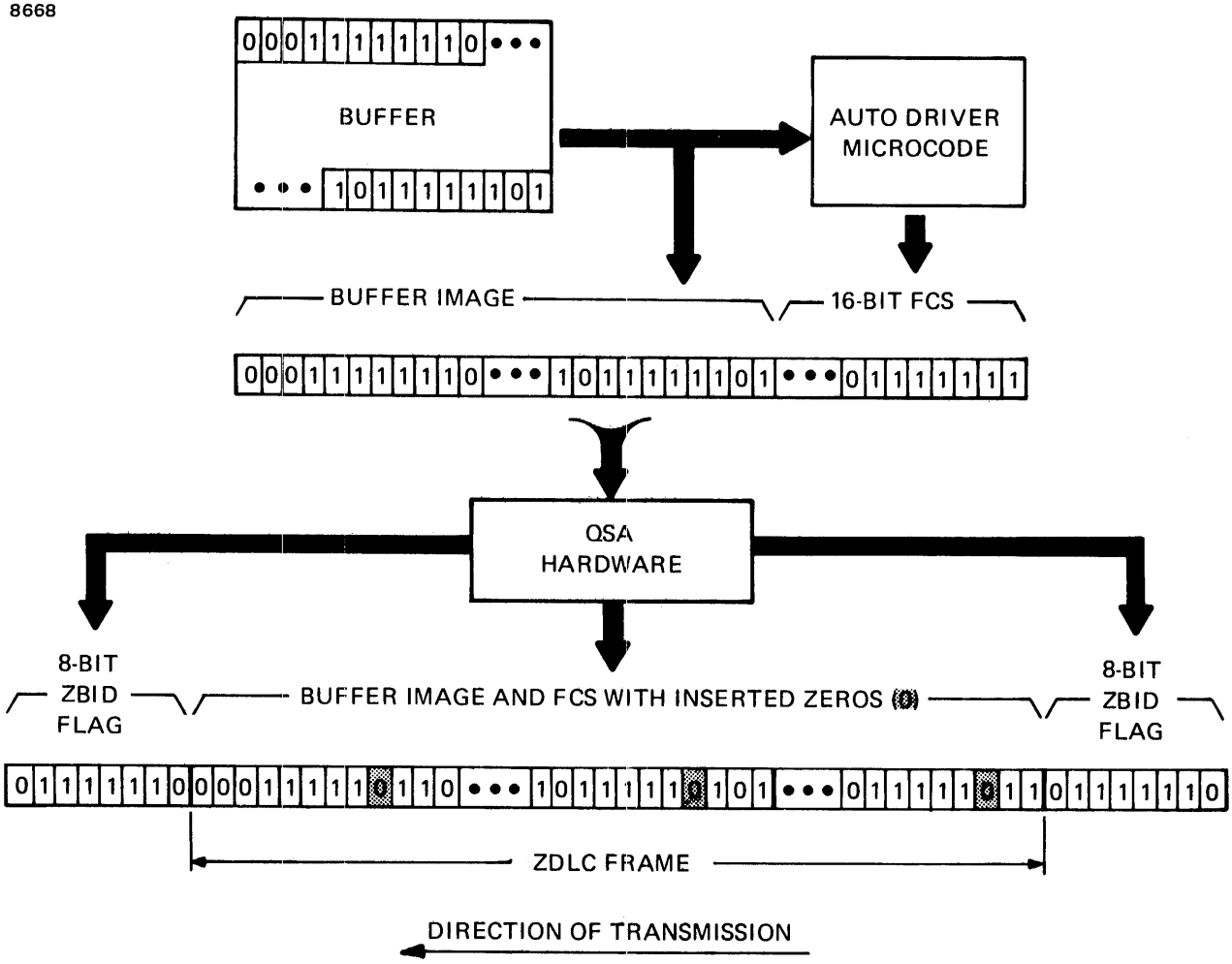


Figure 2-2 Zero-Bit Insertion at Sending Station

2.2.2 Sequence of Bit Transmission for Zero-Bit Insertion/ Deletion Data Link Control (ZDLC) Frames

An octet, the telecommunications equivalent to a character, is a set of eight bits. As shown in Figure 2-3, an octet includes a least significant bit (LSB), with a place value of 2^0, and a most significant bit (MSB), with a place value of 2^7.

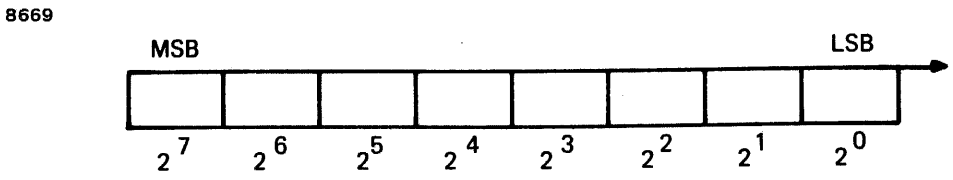


Figure 2-3 The Octet

With Perkin-Elmer's hardware/software support, bits of a frame are transmitted as a continuous succession of octets starting with the A-field and ending with the FCS field, with bit 2^7 of each octet being transmitted first.

Figure 2-4 illustrates this method of transmission.

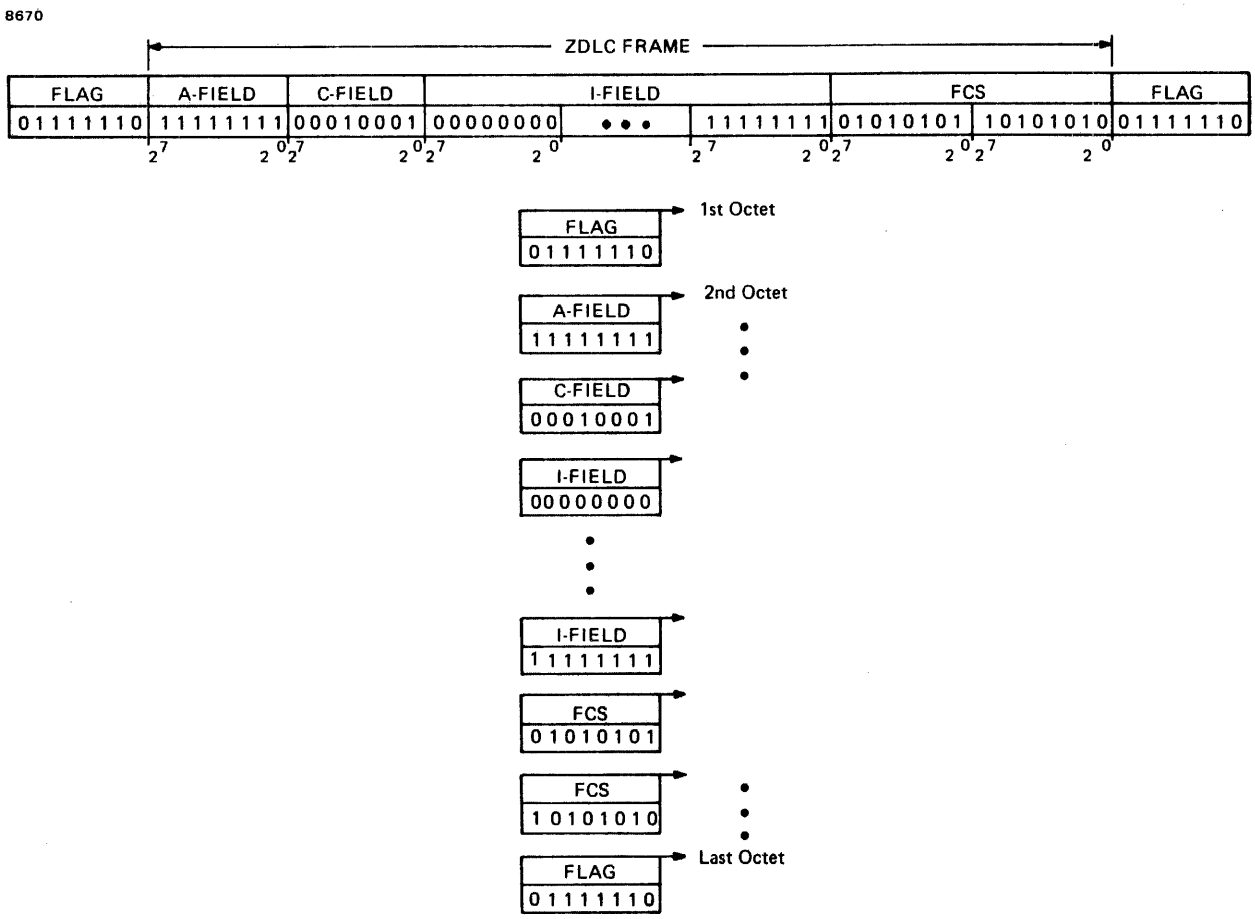


Figure 2-4 ZDLC Frame Transmission by Octets

2.2.3 Zero-Bit Insertion/Deletion Data Link Control (ZDLC) Sequences

There are three bit sequences for controlling the active and inactive transmission states on the data line. These ZDLC control sequences, along with their binary codes, are listed below.

1. ZBID Flag - '01111110' (a 0-bit, six consecutive 1-bits and a trailing 0-bit)

   Flags, by enclosing frames, are involved with all active communications. A primary or secondary also transmits ZBID flags for interframe timefill. Interframe timefill maintains line synchronization and keeps the line in the active state.

2. Abort - '011111110' through '01111111111111110' (a 0-bit, at least 7 but not more than 14 consecutive 1-bits, and a trailing 0-bit)

   A primary or secondary transmits an abort sequence to cancel and nullify a frame being transmitted. Any station receiving an aborted frame (i.e., a frame terminated with abort sequence), ignores it.

3. Idle Line - '01111111111111111...' (at least 15 consecutive 1-bits)

   A primary or secondary having a 2-wire, two-way alternate configuration uses the idle sequence during line turnaround. At line turnaround, no frames are being transferred and the data line is in an idle state.

2.3 INTERPRETATION OF THE ZERO-BIT INSERTION/DELETION DATA LINK CONTROL (ZDLC) FRAME

The following sections discuss the components of the ZDLC frame that are applicable to the ZBID line driver.

2.3.1 Zero-Bit Insertion/Deletion (ZBID) Flag

ZBID flags always enclose the ZDLC frame. When the leading flag is encountered, transmission error checking (i.e., FCS computation) and ZBID begin. When the trailing flag is encountered, transmission error checking and ZBID end. In some applications, the trailing flag also marks the beginning of the next frame.

Flags have the additional function of synchronizing the data line by being sent between frames as interframe timefill.

The binary value of the ZBID flag octet is always '01111110', hexadecimal value, X'7E'. All stations attached to the line continually hunt, on a bit-by-bit basis, for this binary sequence. To achieve data transparency, the ZBID technique prohibits the flag sequence from occurring anywhere within the ZDLC frame. Therefore, a receiving station will always recognize this sequence as a ZBID flag. Following any ZBID flag is either:

- a ZDLC frame,
- another ZBID flag (for interframe timefill),
- an idle control sequence or
- an abort control sequence.

An idle or abort sequence, however, does not have to follow a ZBID flag.

2.3.2 Address Field (A-Field)

The A-field is the first field of a ZDLC frame. It holds a secondary station address, which is a unique line-level address of the secondary sending or receiving the ZDLC frame. The A-field never contains the address of a primary station.

There are two mutually exclusive A-field formats:

- Unextended, used with unextended ZDLC frame formatting.
- Extended, used with extended ZDLC frame formatting.

NOTE

The 'selective read' feature of the ZBID driver provides support of the unextended format only.

Unextended addressing, also known as single addressing, is a standard method for addressing one secondary station at a time. The unextended A-field format consists of one octet, as shown in Figure 2-5, enabling the primary to address any one of a maximum of 254 secondaries.

8672

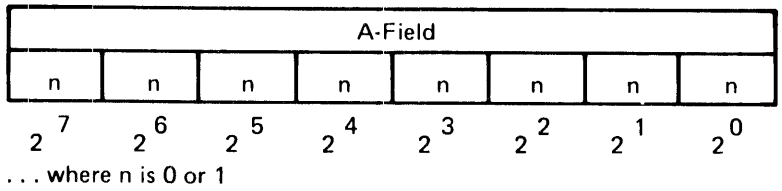


Figure 2-5 Unextended A-Field Format

The unextended addressing format is the only addressing scheme available within SDLC. In HDLC and ADCCP, it may be specified optionally in lieu of the extended format.

Extended addressing is an enhanced method for addressing one secondary at a time. The extended A-field format consists of one or more octets, enabling the primary to address more than 254 secondaries with no maximum (see Figure 2-6).

8673

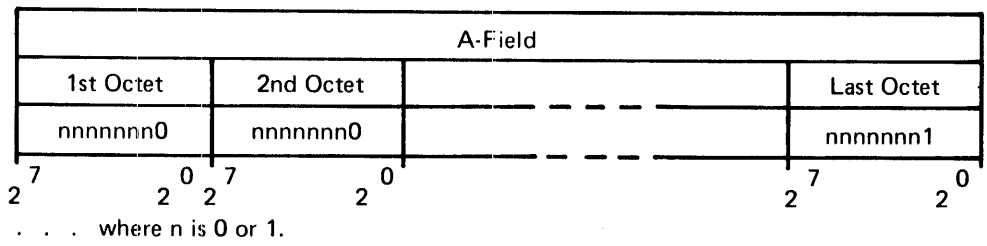


Figure 2-6 Extended A-Field Format

Extended addressing is effected by resetting bit 2^0 of all addressing octets except the last; bit 2^0 of the last addressing octet is set to one. At least one addressing octet is required in the A-field. If one address is to be specified, bit 2^0 of its octet must be set. When more than one addressing octet B is used to address the secondary, bit 2^0 of each addressing octet, except the last, must be reset to signify that additional addressing octets follow. The last octet must be set, marking it as the final addressing octet.

An addressing technique known as compromised single addressing is also available with ZDLC. Compromised single addressing can be used by a station setup for unextended addressing when that station wants to communicate with a station having extended addressing. The compromise is accomplished by setting bit 2^0 of the unextended addressing octet, thereby making it appear as the last octet of an extended address chain.

Two special addresses, global and null, are defined for A-field use, but are not supported as such by the ZBID line driver.

An address of '11111111' (X'FF') is reserved for the global address. It is used by a primary when the address of a specific secondary is unknown or unnecessary; e.g., switched connections and broadcast transmissions.

An address of '00000000' (X'00') is reserved for the null (no station) address. It is used by a primary when no response is required for transmitted frame(s); e.g., station transmission test. When the first octet of any A-field has a null address, other stations ignore the associated frame.

2.3.3 Frame Check Sequence (FCS) Field

The FCS field is generated by cyclic redundancy checking (CRC) and is used for transmission error checking. By definition, the FCS is the remainder of a CRC modulo-2 division, using the generator polynomial, $X^{16}+ X^{12}+ X^{5}+1$, as a divisor. Excluding zeros inserted for transparency, the CRC computation includes the contents of the A-, C- and I-fields. For more information on CRC, see the appropriate Processor User's Manual or Instruction Set Reference Manual.

The ZBID driver supports an image write that allows the user to calculate on FCS other than the standard CRC-16 or to specify no FCS transmission at all.

2.4 ZERO-BIT INSERTION/DELETION DATA LINE CONTROL (ZDLC) CONFIGURATIONS

ZDLC permits two-way alternate (half-duplex) and two-way simultaneous (TWS) (full-duplex) communications between stations on a data line.

With two-way alternate communication, only one station at a time can transmit frames over the data line. For example, a primary transmits one or more frames, a line turnaround occurs and the secondary transmits one or more frames in the opposite direction. A two-way alternate data line is either a switched (telephone) line or a nonswitched (private or leased) line. The nonswitched data line is either 2- or 4-wire; 2-wire lines are exclusively for two-way alternate, while 4-wire lines are for two-way alternate or TWS.

With TWS communication, two stations can simultaneously transmit frames over the data line and transmission requires no line turnaround. A TWS line is a 4-wire, nonswitched line. As previously mentioned, stations may also use the 4-wire line for two-way alternate communication (see Figure 2-7).

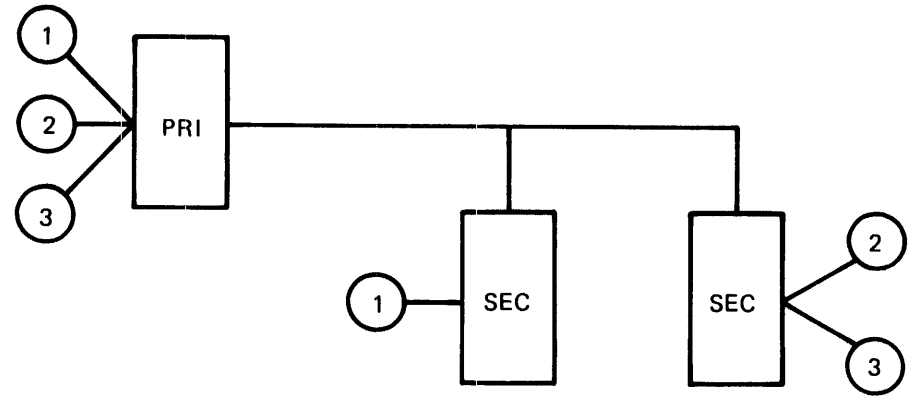


Figure 2-7 Polling Configuration for Multiplexing Data Streams over a Single Line

CHAPTER 3
GENERAL DESCRIPTION OF THE ITAM/32 ZERO-BIT INSERTION/DELETION (ZBID) LINE DRIVER

3.1 INTRODUCTION

The ITAM/32 ZBID line driver is part of the ITAM/32 software package. Using the ZBID technique described in Chapter 2, this line driver controls line synchronization, data transparency and data blocking for medium- to high-speed (1,200 to 19,200 baud) communication lines. Even though the line driver does not support any specific protocol, its flexible design is ideally suited to support line access for the bit-synchronous protocols; namely, synchronous data link control (SDLC), high-level data link control (HDLC) and advanced data communication control procedures (ADCCP). Features of the ZBID line driver include:

- One level of line access (see Figure 3-1) - A communications user task (u-task) can access the bit-synchronous communications line by issuing a supervisor call (SVC). This call is based on the standard ITAM/32 SVC15 parameter block. SVC15 access is a nonbuffered access where the user must handle the protocol support. It has the advantages of flexible buffer control and a powerful repertoire of driver commands. Section 3.2 discusses SVC15 access.

- Continuous error control - The line driver has expanded capabilities for continuous error control of the bit-synchronous protocol. Section 3.2.2 discusses this capability.

- Configuration options - As specified DEVICE statements at system generation (sysgen) time, the ZBID line can be configured as 2- or 4-wire, simplex, half- or full-duplex, switched or leased.

- Line driver modification - With the SVC15 MODE command, a u-task can dynamically modify certain line driver control fields.

- Protocol control - The line driver automatically generates, recognizes and validates ZBID flags, abort requests and polynomial cyclic redundancy check (CRC) requests.

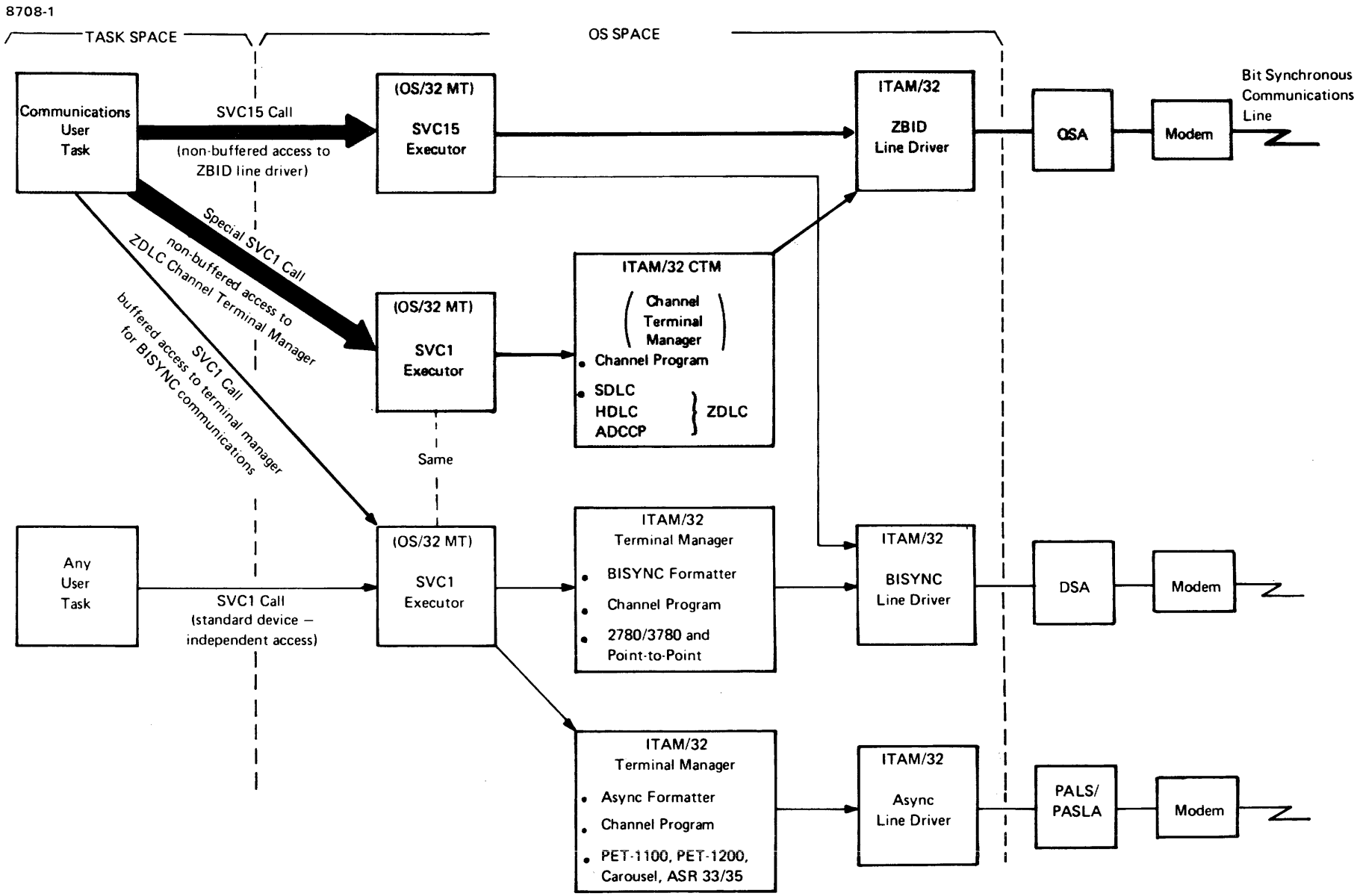


Figure 3-1 ITAM/32 Terminal Manager/Line Driver Relationships

3.2 SUPERVISOR CALL 15 (SVC15) ACCESS TO THE ZERO-BIT INSERTION/ DELETION (ZBID) LINE DRIVER

The following sections describe SVC15 access to the ZBID line driver.

3.2.1 Read and Write

The read and write method of access is the standard method of data transfer. All standard ITAM/32 buffer formats are supported. For more information on buffer formats, see the OS/32 Basic Data Communications Manual.

3.2.2 Continuous Error Processing (CEP)

A u-task requests the capability for CEP by setting the CEP bit (2^3) in the SVC15 function code byte. This capability is meaningful only for a u-task reading a stream of contiguous frames into queued or chained buffers during bit-synchronous communications.

CEP permits a u-task to identify three of the encoded SVC15 termination codes (listed in Table 3-2) as nonterminating error conditions. These codes are exclusively associated with the line driver READ command. They are:

- X'05' - Data check frame check sequence (FCS)
- X'0E' - Overflow
- X'21' - Frame abort
- X'20' - Idle line

If CEP has not been requested and one of these errors is encountered, the following occurs:

- The SVC15 request terminates because of the error.
- Control returns to the u-task.
- The u-task immediately issues another READ in time to read the next frame. If the two-way simultaneous (TWS) bit-synchronous communications is over a high-speed line, the next frame will have already been sent and lost before the u-task is ready to receive it. Hence, two frames would be lost because of a single error.
- The task selectively rejects the errored frame.

CEP alleviates the problem of losing any frame(s) following a bad frame by preventing the line driver from being terminated whenever one of the three termination codes listed above is encountered. When the line driver encounters one of these nonterminating error conditions, it:

- places the error status halfword into the "bytes used" halfword of the queued or chained buffer being used,
- returns the buffer to the task, as it would for a correctly received frame, and
- reads the next frame into the next available user buffer.

With these operations performed, the u-task can process the bad frame and still be ready to receive the next one.

After each input/output (I/O), the u-task is responsible for checking the "bytes used" halfword of the queued/chained buffers. If any halfword is negative (i.e., if the most significant bit (MSB) is set), the halfword contains status information for an errored input frame. If the halfword is positive, the "bytes used" halfword contains the total number of bytes read, including the 2-byte FCS. Also, because the MSB determines the meaning for this halfword, it cannot be used for the byte count; this limits the maximum buffer size to 32,767 bytes.

3.3 COMMAND REPERTOIRE FOR THE ZERO-BIT INSERTION/DELETION (ZBID) LINE DRIVER

Table 3-1 lists the ZBID line driver commands that may be used in a driver command word (DCW) chain. With the exception of the four commands marked with asterisks (*), these commands are the same as those described in the OS/32 Basic Data Communications Reference Manual for all ITAM/32 line drivers. The asterisk marked commands are unique to the ZBID line driver because they are executed by special routines within the driver, not by common ITAM/32 subroutines. These four commands are discussed in the following sections.

TABLE 3-1 ZBID LINE DRIVER COMMANDS

COMMAND FAMILY	COMMAND	HEX	DESCRIPTION
NULL	NOP	0000	No operation
	WAIT	0008	Time interval wait
	XFER	0010	Transfer command chain address
	CXFER	0018	Conditional transfer
CONTROL	EXAMINE	0001	Examine device status
	RING WAIT	0009	Wait for the phone to ring
	ANSWER	0011	Answer the phone
	DISCONNECT	0019	Hang up the phone
READ	READ	0002	Read with ITAM/32 buffer management
	SELECTIVE READ*	001A	Read with address check
WRITE	WRITE	0004	Write with ITAM/32 buffer management
	WRITE IMAGE*	001C	Write without FCS
	WRITE IDLELINE*	0024	Write idle line
MODE	TOUT	0006	Change time-out constants
	CMD2	000E	Change programmable adapter command
	RCMD	0016	Change READ command
	WCMD	001E	Change WRITE command
	RDIS	0026	Change READ-DISABLE command
	WDIS	002E	Change WRITE-DISABLE command
	DISC	0036	Change DISCONNECT command
	ADDR*	005E	Change selective address

* Executed by special routines of the ZBID line driver

The following sections describe the specific commands within each command family. The binary setting and hexadecimal mask, for the low-order byte of the DCW is given for each command.

For all commands described, normal completion causes the next DCW to be fetched; if none exists, a successful completion status is returned to the status halfword of the SVC15 parameter block. Nonerror status conditions are noted by setting the appropriate bits in the status halfword; upon termination of the SVC15, the value of the status halfword may reflect the accumulation of several such conditions.

Invalid flag bits cause the SVC15 to terminate with the appropriate error status.

3.3.1 NULL-Type Commands

Chain command and command trap flags are valid. There are four NULL-type commands, none of which issue I/O instructions to the adapter.

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
NOP	0000 0000	00	1
WAIT	0000 1000	08	1
XFER	0001 0000	10	1
CXFER	0001 1000	18	2

3.3.1.1 NOP Command

The NOP command performs no operation; one data field is fetched but not used. (It must, however, specify a valid program address.) If chain command (bit 0) is set, the next command is fetched.

3.3.1.2 WAIT Command

The WAIT command is used to suspend driver execution for a specified interval. One data field is fetched; it must point to a halfword containing the value of a time interval in multiples of 100ms. The driver waits until the specified interval expires, and then continues execution (if chain command is set) or terminates (if chain command is reset).

3.3.1.3 Transfer In (XFER) Command

The XFER command is used to specify the next DCW in a chain that does not exist in contiguous memory. One data field is fetched; if chain command is set, the next command to be executed is fetched from the address contained in the data field. Thus, a branch to a DCW is performed.

3.3.1.4 Conditional Transfer (CXFER) Command

The CXFER command is used to test the internally maintained status of the SVC15 call. Two data fields are fetched. The first data field points to two halfwords, the first of which is logically ANDed with the current state of the ITAM/32 logical status halfword. The result is compared to the second halfword of the pair, and if equal, the next command to be executed is specified by the second data field. Otherwise, command execution continues with the next command in the current chain. The CXFER command can be used to test for specific conditions, as indicated by the ITAM/32 logical status; the first halfword of the first data area contains a mask with a ONE in each bit portion to be tested; the second halfword of the first data area contains the value to be tested against; e.g., if one or more special characters were detected after a read operation, a different command sequence might be desired.

3.3.2 CONTROL-Type Commands

There are four CONTROL-type commands. They are:

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
EXAMINE	0000 0001	01	1
RING WAIT	0000 1001	09	0
ANSWER	0001 0001	11	0
DISCONNECT	0001 1001	19	0

3.3.2.1 EXAMINE Command

The EXAMINE command is used to obtain the device status of the specified adapter. This command obtains one data field. The value obtained specifies the address of a writable byte into which the status of the device is stored. The last known physical device status is fetched from a byte in memory that is maintained by the driver during I/O activity. When the byte is nonzero, its contents are returned to the user and the byte is reset to zero. When the byte is zero, a sense status is performed on the device and its present status is returned to the user.

3.3.2.2 RING WAIT Command

The RING WAIT command is used to suspend fetching of DCWs until a ring signal is received from the adapter. This command fetches no data fields and interrupts from the adapter are enabled. The data terminal ready (DTR) lead to the modem is not enabled. The command terminates when a ring signal is received from the adapter. When chain command is set, execution continues with the next command; otherwise, the driver terminates. When time-out is set, the command waits only as long as the value specified in the write timer halfword; when this interval expires, time-out error status is set. If time-out is not set, the command waits indefinitely for a ring signal.

3.3.2.3 ANSWER Command

For nonswitched lines and switched lines that are already connected, the ANSWER command terminates immediately. For dial-in lines not connected, the DTR lead to the modem is enabled, causing the modem to answer the incoming call. The command terminates when the data set indicates it is ready for I/O. Time-out and chain commands are handled as described in the RING WAIT command.

3.3.2.4 DISCONNECT Command

The DISCONNECT command is used to disconnect from a switched line. The command resets the DTR lead to the mode, suspends DCW fetching for one second and either continues to the next command (if chain command is set) or terminates (if reset).

3.3.3 READ-Type Commands

There are two READ-type commands. They are:

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
READ BUFFER	0000 0010	02	1 or 2
SELECTIVE READ	0001 1010	1A	1 or 2

3.3.3.1 READ BUFFER Command

The READ BUFFER command is used to read data into the specified buffers. One or two data fields are fetched, depending on the buffer type; i.e., chained, direct or indirect. The data field specifies a buffer or buffer chain into which data is read. All buffers must be in the same logical segment of the task as the address contained in the first data field of the parameter block. The first byte of the data field fetched specifies whether the buffer type is direct text, indirect text or chained buffer. If the buffer type is direct text, a second data field is fetched.

3.3.3.2 SELECTIVE READ Command

The ZBID line driver performs a selective read the same way the ITAM/32 subroutine performs a standard read. But if the first byte of an input frame does not match the address byte previously specified with the MODE ADDR command, the line driver aborts the reading of that frame and attempts to read the next input frame. The selective read feature of the ZBID driver only provides support of the unextended format.

3.3.4 WRITE-Type Commands

Chain command, command trap and time-out are valid flag bits. There are three WRITE-type commands.

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
WRITE BUFFER	0000 0100	04	1 or 2
WRITE IMAGE	0001 1100	1C	1 or 2
WRITE IDLELINE	0010 0100	24	0

3.3.4.1 WRITE BUFFER Command

The WRITE BUFFER command is used to write data from the specified buffers. One or two data fields are fetched, depending on the buffer type. The data field specifies a buffer or buffer chain from which data is written to the line. All buffers must be in the same logical segment as the address in the first data field of the parameter block. The left-most byte of the first data field fetched specifies whether the buffer type is direct text, indirect text or chained buffers. If the buffer type is direct text, a second data field is fetched.

3.3.4.2 WRITE IMAGE Command

The ZBID line driver performs a write image the same way the ITAM/32 subroutine performs a standard write, but the entire frame is sent without FCS computation.

3.3.4.3 WRITE IDLELINE Command

The WRITE IDLELINE command transmits a ZBID idle sequence that consists of a 0-bit followed by 15 or more consecutive 1-bits and puts the ZBID line into an inactive state.

3.3.5 MODE-Type Commands

Chain command and command trap are valid flag bits. If the default value specified in the individual driver descriptions are acceptable, no MODE commands need to be executed. Once a value is changed by a MODE command, the only means of restoring the default condition is by a MODE command specifying the correct value. Modifications should be coordinated if access is being shared by more than one program. There are eight defined MODE-type commands.

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
MODE TOUT	0000 0110	06	1
MODE CMD2	0000 1110	0E	1
MODE RCMD	0001 0110	16	1
MODE WCMD	0001 1110	1E	1
MODE RDIS	0010 0110	26	1
MODE WDIS	0010 1110	2E	1
MODE DISC	0011 0110	36	1
MODE ADDR	0101 1110	5E	1

3.3.5.1 MODE TOUT Command

The MODE TOUT command is used to set the error time intervals for commands that enable the time-out flag. One data field is fetched containing the address of the first of two halfwords; the first contains an error time-out interval for read-type operations and the second contains an interval for write-type operations (both in units of one second).

The timer is strictly for error detection; when it expires, the command is terminated in error. Interval timing is performed via the WAIT command, which uses a separate (100ms resolution) clock.

3.3.5.2 MODE CMD2 Command

The MODE CMD2 command is used to specify the device-dependent command used to set adapter options. One data field is fetched with the address of a byte containing the device command to be output to specify programmable adapter options such as parity, number of data bits, etc.

3.3.5.3 MODE RCMD and MODE WCMD Commands

The MODE RCMD and MODE WCMD commands specify the device-dependent commands to set read or write modes in the adapter. One data field is fetched for these commands with the address of a byte containing the device command to be output for read or write data transfers, respectively.

3.3.5.4 MODE RDIS and MODE WDIS Commands

The MODE RDIS and MODE WDIS commands specify the device-dependent commands used to set the read or write side of the adapter quiescent. One data field is fetched with the address of a byte containing the read or write side of the line.

3.3.5.5 MODE DISC Command

The MODE DISC command specifies the device-dependent command used to disconnect the adapter from the line. One data field is fetched with the address of a byte containing the device command to be output to disconnect the communications line (reset data terminal ready).

3.3.5.6 MODE ADDR Command

The MODE ADDR command specifies the contents of the selective read address byte used by the selective read. The driver fetches one data field containing the frame address to be checked by the SELECTIVE READ command.

Although the ZBID line driver has no specific command for outputting an abort sequence, the user can abort a frame being written by issuing an SVC15 call; the halt I/O bit 0 in the SVC15 function code byte must be set. The call causes an abort sequence to be transmitted, terminating the write. (The abort sequence consists of a 0-bit and at least seven, but not more than 14, consecutive 1-bits, followed by a trailing 0-bit.)

3.4 SUPERVISOR CALL 15 (SVC15) ERROR HANDLING

Table 3-2 shows the encoded SVC15 termination codes.

TABLE 3-2 ENCODED SVC15 TERMINATION CODES FOR ZBID LINE DRIVER

TERMINATION CODE	CAUSE
00	No errors - Successful SVC15 call with no errors.
05*	Data check - Termination by data error, FCS; also, bit 2 in the SVC15 status halfword is set.
06	Buffer Limit - Buffer limits reached; no proper ending sequence.
0A	Loss of carrier - Lost carrier on reads.
08	CL2S error - Lost clear to send on write.
0C	Data set not ready.
0D	Device unavailable - Adapter not present.
0E*	Overflow - Character overflow.
0F	Ring - Phone is ringing.
10	Buffer overrun-1 - Busy and/or done bits in chain buffers bad.
11	Number of commands executed (NCE) overflow - More than 255 commands executed.
12	Task trap error - Task trap queue was full, invalid or nonexistent during trap attempt.
13	Buffer overrun-2 - Event service routine (ESR) does not execute in time; may indicate that priority is too low.
14	Time-out.
15	Halt I/O - Request aborted I/O.
17	Bad Sequence.
18	Illegal command - Invalid command or modifier.
19	Memory fault-1 - Memory fault when referencing data. (See Section 3.5 for possible causes.)

TABLE 3-2 ENCODED SVC15 TERMINATION CODES FOR ZBID LINE DRIVER (Continued)

TERMINATION CODE	CAUSE
1A	Memory fault-2 - Memory fault when referencing data. (See Section 3.5 for possible causes.)
1B	Illegal logical unit (lu) - lu was unassigned invalid; might be hardware problem.
1C	Illogical status - Device status invalid; might be hardware problem.
1D	Power fail - Power failure has occurred.
1E	Illegal action - Illegal software condition.
1F	Illegal translation table - translation table invalid.
20*	Idle line - Line was set inactive by receipt of an idle line sequence.
21*	Frame abort - Frame was terminated by the receipt of an abort sequence.
22	Invalid frame - Received frame does not have enough bytes or has a bit length that is not an even multiple of eight. (Any frame must have at least 32 bits: 8-bit A-field, 8-bit C-field and 16-bit FCS field; the line driver mandates that all frames consist of an integral number of octets.)
23	Queue empty - A from queue list was found to be empty prior to transfer completion.
24	Queue overflow - The receive queue list for a queued I/O has overflowed.

* These commands are flagged for CEP

3.5 CHAINED/QUEUED BUFFER LINK WORD FLAG BYTE

The OS/32 Basic Data Communications Reference Manual details the buffer formats supported by ITAM/32 and the ZBID driver. For chained and queued buffers, the flag byte (bits 0 to 7) of the buffer link word is used by the ZBID driver as follows:

BIT NUMBER	0	1	2	3	4 5 6 7
MEANING	Busy	Done	Frame End	Frame Begin	User must clear to zero

Where:

Busy/Done — is standard ITAM usage; see the OS/32 Basic Data Communications Reference Manual.

Frame Begin — is set on the first or only buffer in a chain; this allows many buffers chained together to be written as one frame.

Frame End — is set on the last or only buffer in a chain.

NOTE

These bits must be set properly or the SVC15 call will cause termination codes with X'19' or X'1A'.

3.6 QUAD-SYNCHRONOUS ADAPTER (QSA) PROGRAMMING

The ZBID line driver uses QSA device commands to:

- initialize the QSA to ZBID mode,
- perform reads and writes,
- disable previously issued reads and writes and
- disconnect switched lines.

ZBID default values for the QSA device commands, as assembled within the device control block (DCB) and issued by the ZBID line driver, are listed in Table 3-3.

TABLE 3-3 ZBID DEFAULT VALUES FOR QSA DEVICE COMMANDS

QSA DEVICE COMMAND	DEFAULT
Read Time-out	10 seconds
Write Time-out	10 seconds
Device-Read	X'59'
Device-Write	X'5B'
Disable-Read	X'D9'
Disable-Write	X'DB'
Disconnect	X'C1'
Set-Programmable-Option	X'02'

Normally, the QSA device commands are of no concern to the ITAM/32 SVC15 user because they are issued by the driver, transparent to the user. But for the rare occasion when a user wants to modify a QSA device command, the SVC15 MODE commands are available. (For information on using the SVC15 MODE commands, see the Quad-Synchronous Adapter (QSA) Programming Manual.) Tables 3-4 and 3-5 show the QSA device command byte. Table 3-6 shows the control mode command byte.

Likewise, the QSA device status is of no concern to most ITAM/32 SVC15 users. When a user wants to interrogate the actual QSA device status, the EXAMINE command is available. Tables 3-7 and 3-8 show the QSA device status byte that the QSA hardware supplies to the ZBID line driver.

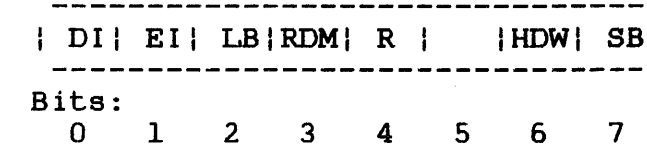


TABLE 3-4 QSA DEVICE COMMAND BYTE (TRANSMIT)

BIT	DESCRIPTION
0	Disable interrupts.
1	Enable interrupts.
2	Local loop-back mode - Causes the outgoing data to be diverted from the data set to the receive side of another line pair.
3	Reset data mode - Causes the communications line to transmit continuously the last contents of the transmit data holding register; this transmission has no zero-bit insertion and it starts at the completion of the current character being shifted out.
4	Ready DTR - Causes the ready control line (to the data set) to go to the ON level.
5	Unused by ZBID.
6	Request to send (RTS) half-duplex write in 2-wire mode - Causes this bit to ready the line for transmitting; when reset, it causes the line to go into read mode. For both 2- and 4-wire lines, this bit activates the RTS control line going to the data set; when reset, it deactivates this control line.
7	Steering bit - Causes command instructions to go into the I/O mode command registers.

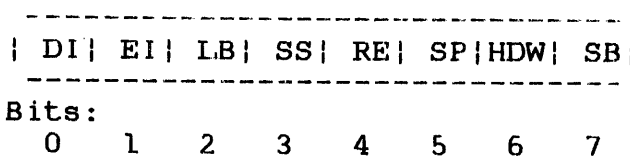


TABLE 3-5 QSA DEVICE COMMAND BYTE (RECEIVE)

BIT	DESCRIPTION
0	Disable interrupts.
1	Enable interrupts.
2	Local loop-back mode
3	Sync search - Causes the receive line to go to the FLAG search mode.
4	Ready
5	Special
6	RTS half-duplex write
7	Steering bit - Causes command instructions to go into the I/O mode command register.

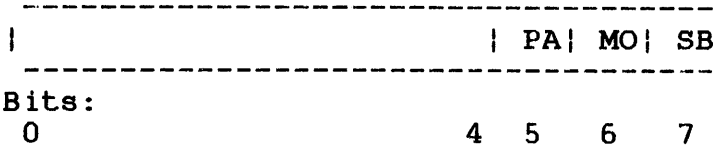


TABLE 3-6 CONTROL MODE COMMAND BYTE

BIT	DESCRIPTION
0-4	Unused by ZBID.
5	Parity - Reset for no parity.
6	Mode - Set to specify ZBID.
7	Steering bit - Causes reset for storing command data into the control-mode command register.

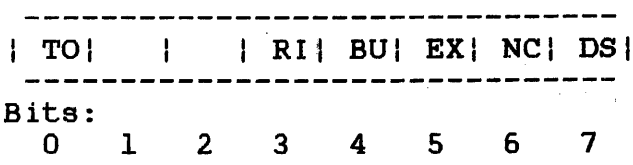


TABLE 3-7 QSA DEVICE STATUS BYTE (TRANSMIT)

BIT	DESCRIPTION
0	Transmit overflow - Data not sent to QSA in proper time.
1	Not used with ZBID.
2	Not used with ZBID.
3	Ring - Ring-control line from the data set is on.
4	Busy - Transmit line is not ready to handle data.
5	Examine - Transmit line has bit 0 set.
6	Not clear to send - Clear to send (CL2S) control line from the data set is at OFF level.
7	Data set not ready - The data set ready (DSR) control line from the data set is at OFF level.

RO	TE		RI	BU	EX	EO	DS

Bits:
0 1 2 3 4 5 6 7

TABLE 3-8 QSA DEVICE STATUS BYTE (RECEIVE)

BIT	DESCRIPTION
0	Receive overflow - Data not read from QSA in proper time.
1	Termination - Received ZBID flags or abort sequence.
2	Not used with ZBID.
3	Ring - Ring control line from the data set is on.
4	Busy - Receive line is not ready to handle data.
5	Examine - Receive line had either bit 0 or 1 set.
6	Carrier off - Carrier control line from the data set is at OFF level.
7	Data set not ready - The DSR control line from the data set is at OFF level.

CHAPTER 4
THE ETHERNET LINE DRIVER

4.1 INTRODUCTION

Ethernet is a multi-access, packet-switched communications system for carrying data among locally distributed computing systems. The Ethernet data link controller (EDLC) provides processor-to-processor serial communication, clocked at 10Mbits over a common coaxial cable segment, using a peer-to-peer protocol known as carrier sense multiple access with collision detection (CSMA/CD). The Ethernet specification will support over 100 processor interfaces in a variety of configurations:

- Coaxial cable segments up to 500 meters (546.8 yards)
- Up to 100 transceiver tap connections (stations) per coaxial cable segment
- Up to 50 meters (54.7 yards) of transceiver cable between an EDLC and its transceiver tap on a coaxial cable segment
- Up to 2,500 meters (2,734.7 yards) maximum station separation
- One point-to-point link up to 1,000 meters (1,093 yards) in length between any two coaxial cable segments
- Up to 1,024 stations per network

The EDLC is connected in parallel to the common coaxial cable bus through an Ethernet transceiver that contains the necessary tap hardware and buffer electronics to transmit and receive over the 50 ohm coaxial cable. Communication sequences over the half-duplex 10Mbit channel are divided into packets with frame sizes ranging from 64 to 1,518 bytes, complete with two 48-bit address fields in the header and a 32-bit cyclic redundancy checking (CRC) frame check sequence (FCS) for error detection.

4.2 NETWORK SOFTWARE AND DRIVER

The function of the programming interface to the 35-863 Ethernet protocol module (EPM) is to provide the means of sending and receiving Ethernet frames to another processor or Ethernet station sharing the same network at a remote site. The transmit data (T_DATA) and receive data (R_DATA) registers are single, 8-bit buffers that operate as the data interface between the software and the EPM first-in/first-out (FIFO) buffers.

There is a 2kb transmit FIFO buffer that is written from the T_DATA register and there is a 13.5kb receive FIFO that writes to the R_DATA register. The command (C_REG) will return a reply in the S_REG on a one-on-one basis; i.e., the next command code cannot be executed until the previous command replay has been taken from the S_REG. The command codes are used to activate optional EPM characteristics, to activate a report of operating statistics or to run loopback tests or diagnostics. The control (H_REG) register is used to read the state of five hardware control lines that provide the software with a snapshot of EPM machine states. The EDLC board (32-217) provides private multiplexor (PMUX) bus access to a subset of the hardware control lines reported by the H_REG.

An OUTPUT command is used to address the 35-863 EPM registers. The bits CMD13:15 in the OUTPUT command must contain the address of the desired register. The contents of the 35-863 EPM register are delivered to the user process by a READ DATA command, which starts a data-request cycle on the PMUX bus. A WRITE DATA command delivers one byte from the user process to the selected register by executing a data-available cycle on the MUX bus. The selector channel (SELCH) will read and write the EPM registers through a sequence of alternating data-available/data-request and status-request operations on the PMUX bus.

4.3 INTRODUCTION TO FRAMES

All commands, responses and information transferred between tranceiver stations are binary codes conveyed within a formatted Ethernet frame. This frame is the vehicle for all communications on the coaxial cable.

The EPM formats part of the frames and performs the CSMA/CD transmit link management functions required to successfully deliver frames onto the network. Figure 4-1 illustrates the Ethernet frame format. Only frames with a destination address matching the station address are accepted by the EPM for transfer to the host system. The module performs physical, multicast group and broadcast address recognition. Figures 4-2 and 4-3 show the Ethernet frame as it appears in the transmit and receive FIFO of the EPM.

8747

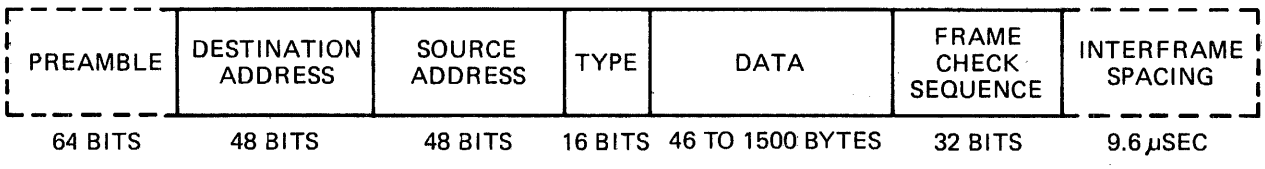


Figure 4-1 Ethernet Frame Format

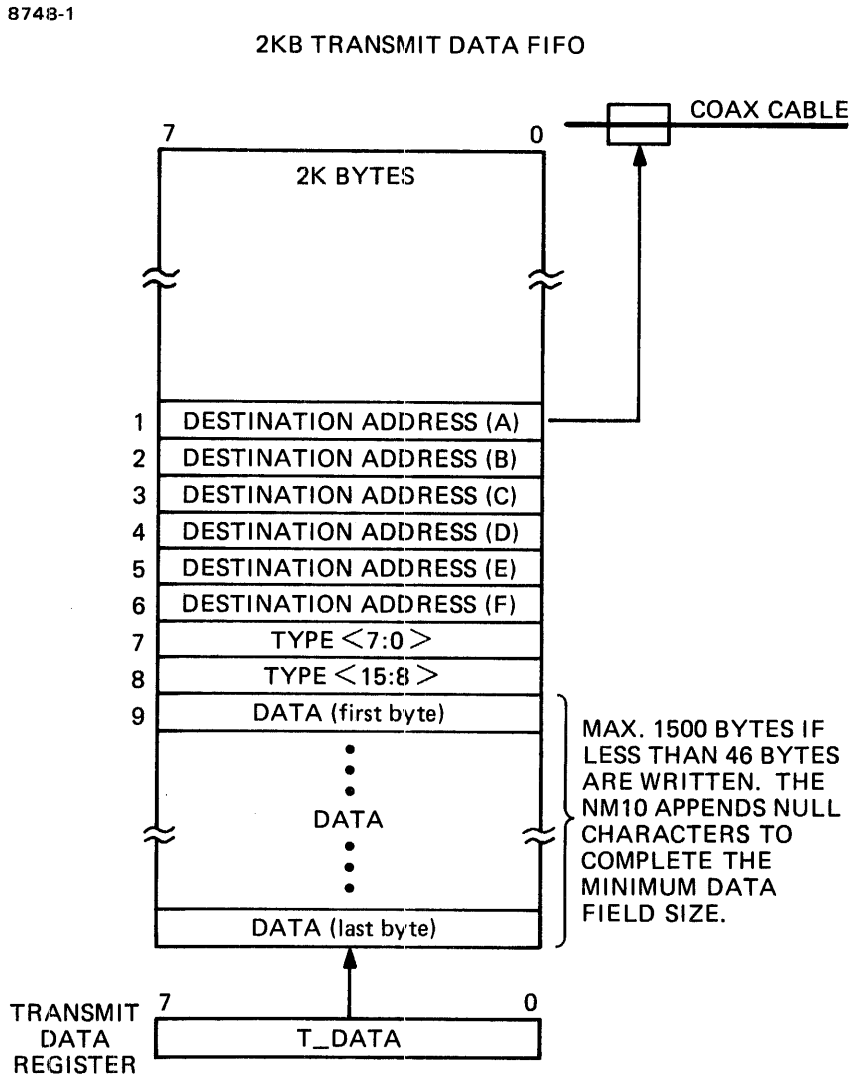
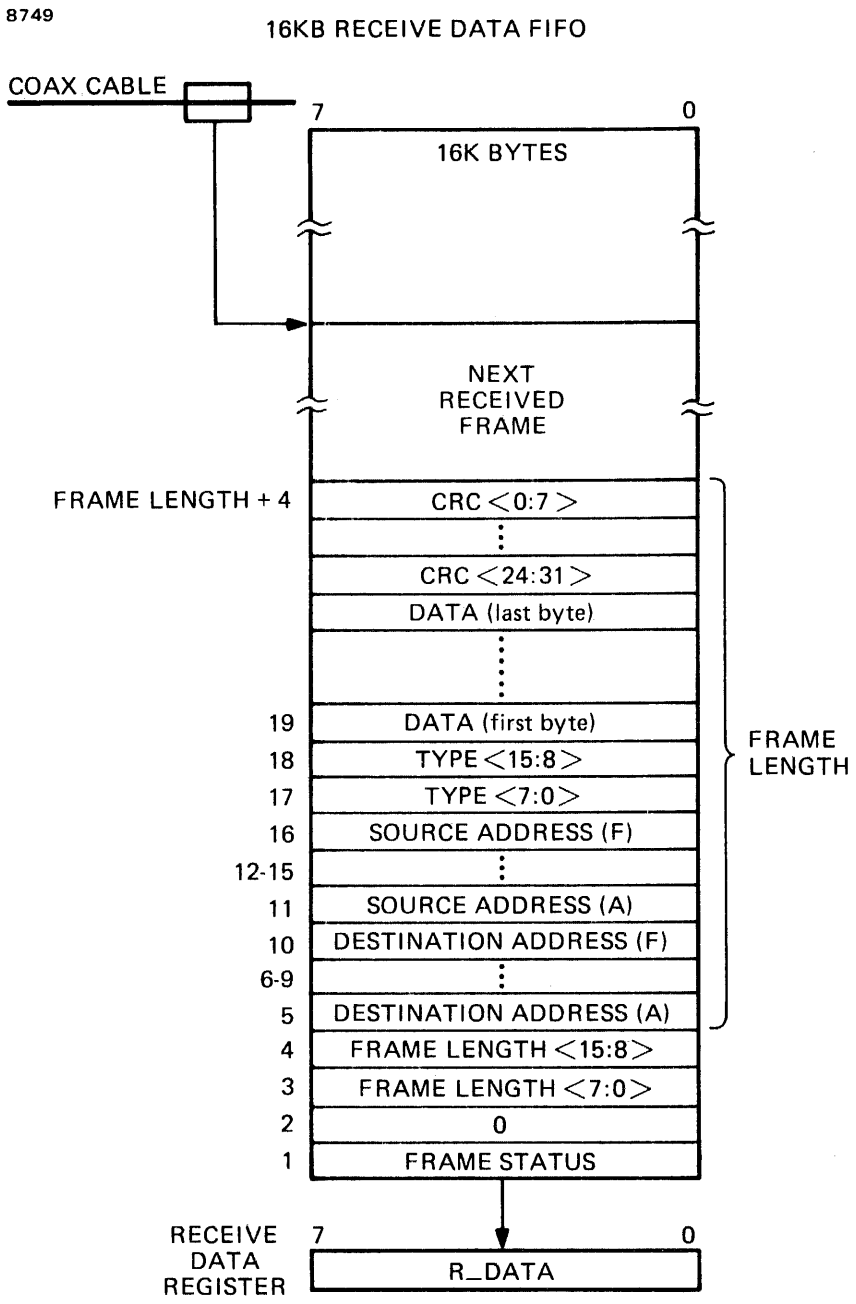


Figure 4-2 Transmit FIFO Buffer Data Organization



A byte of data is removed from the FIFO on each read cycle on R_DATA.

Figure 4-3 Receive FIFO Buffer Data Organization

4.3.1 Ethernet Protocol Module (EPM) Ethernet Address

Each EPM has been assigned to a unique 48-bit physical address. The address assigned to the board provides the station with a distinct physical address that is unique to that station on any Ethernet. Although this physical address has been programmed into nonvolatile memory, the copy of the address in volatile memory can be changed by issuing the LOAD PHYSICAL ADDRESS command.

4.3.2 Multicast Addresses

A provision is made for recognition of multiple destination addresses by one station. The multiple destination address recognition capability consists of the two following types:

- Multicast group - An address associated with a group of logically related stations. The first bit of the destination field is set=1, followed by the multicast address.
- Broadcast - An address associated with all stations. All 48 bits of the destination field set=1.

TABLE 4-1 EPM COMMAND REGISTER CODES

COMMAND CODE (HEX)	COMMAND FUNCTION
00	Reserved
01	Set module interface loopback mode
02	Set internal loopback mode
03	Clear loopback mode
04	Set promiscuous receive mode
05	Clear promiscuous receive mode
06	Set receive-on-error mode
07	Clear receive-on-error mode
08	Go off-line
09	Go on-line
0A	Run on-board diagnostics

TABLE 4-1 EPM COMMAND REGISTER CODES (Continued)

COMMAND CODE (HEX)	COMMAND FUNCTION
0B-0C	Reserved
0D	Set insert source address mode
0E	Clear insert source address mode
0F	Set physical address to default
10	Set receive all multicast packets
11	Clear receive all multicast packets
12-17	Reserved
18	Report and reset statistics
19	Report collision delay times
1A	Reserved (maintenance)
1B-27	Reserved
28	Load transmit data
29	Load transmit data and send
2A	Load group address(es)
2B	Delete group address(es)
2C	Load physical address
2D-3E	Reserved
3F	Reset

TABLE 4-2 EPM STATUS REGISTER CODES

COMMAND CODE (HEX)	COMMAND STATUS
00	Success
01	Success with retries
02	Illegal command
03	Inappropriate command
04	Failure
05	Buffer size exceeded
06	Frame too small
07	Reserved
08	Excessive collisions
09	Reserved
0A	Buffer alignment error
0B-FF	Reserved

TABLE 4-3 EPM DIAGNOSTIC STATUS CODES

COMMAND CODE (HEX)	COMMAND STATUS
00	Success
01	Read-only memory (ROM) checksum error
02	ROM memory error
03	Address error
04	Loopback failure
05	Carrier sense failure

TABLE 4-4 EPM H_REG CONTENTS

BIT	STATUS
0-2	Not used
3	Transmit data empty (TDE) not present
4	Receive block available (RBA) not present
5	Receive data fullword (RDF) present
6	Status block available (SBA) not present
7	Status register full (SRF) present

4.4 RECEIVE REGISTER/FRAME STATUS

To keep the transmit and receive status reports separate, the receive register (R_DAT) is used to convey received-frame status. At the head of every received frame is a 1-byte status r ; that conveys the following information:

Bits 0:4	Not used
Bit 5	Lost frame
Bit 6	Alignment error
Bit 7	CRC error (least significant bit)

CHAPTER 5
PROGRAMMING THE ETHERNET LINE DRIVER

5.1 GENERAL DESCRIPTION OF THE ETHERNET LINE DRIVER

Ethernet is a carrier sense multiple access with collision detection (CSMA/CD) communications protocol designed for carrying data among locally distributed computing systems. Because there is no central control, access to the channels by stations wishing to transmit, is coordinated by the stations themselves. To acquire the channel, stations check whether the network is busy and wait until the Ethernet medium is idle before they transmit. When an idle period is detected, the deferring station begins to transmit.

The Perkin-Elmer Series 3200 Ethernet interface board has been developed to provide a 10Mb per second, processor-to-processor interface. The board is configured as a direct memory access (DMA) device and is supported by the Perkin-Elmer Series 3200 Ethernet line driver.

The design goals of this driver are two-fold: first, to provide a simple supervisor call 15 (SVC15) access for normal data transfer, and second, to allow a user task (u-task) to access all the features of the Ethernet Protocol Module (EPM) on an individual basis.

Features of the Ethernet line driver include the following:

- A communications u-task can access the bit-synchronous communications line by issuing an SVC15 call. This call is based on the standard ITAM/32 SVC15 parameter block. SVC15 access is a nonbuffered access where the user must handle the protocol support. It has the advantages of flexible buffer control and a powerful repertoire of driver commands.

- The Perkin-Elmer Series 3200 Ethernet Data Link Controller (EDLC) uses one device address for both read and write operations, with both types of interrupts enabled, disabled or disarmed together (i.e., it is not possible to enable read interrupts without write interrupts and vice versa). To support simultaneous reads and writes, two device control blocks (DCBs), with two device codes, are provided for the u-task. These are linked at by system generation using the System Generation/32 (Sysgen/32) Utility, thus allowing each process to discover the state of the other. The interface runs under a selector channel (SELCH) that can be shared with other devices.

5.2 SUPERVISOR CALL 15 (SVC15) ACCESS TO THE ETHERNET LINE DRIVER

The following sections discuss SVC15 access to the Ethernet line driver.

5.2.1 Read and Write

The read and write method of access is expected to be the normal method of data transfer. Chained and queued buffers in the standard ITAM/32 format are supplied by the u-task to the driver. For more information on buffer formats, see the OS/32 Basic Data Communications Reference Manual. For more information on read/write, see Chapter 4.

5.2.2 Ethernet Protocol Module (EPM) Command Access

There are three types of commands implemented by the EPM. They are referred to as Types I, II and III, each defining a different method for controlling the EPM and transferring data.

In a TYPE I command, a command code is written to the command register (C_REG) and the result is read from the status register (S_REG). This type of command is used for controlling the EPM.

TYPE II commands constitute writing one command byte into the command register and then reading several bytes from the status register. This type of command is used for statistics gathering.

In a Type III command, multibyte transfers to the transmit data register and a single byte returned through the status register are involved. The EPM buffers the data written to the transmit data register (T_DATA) in a 1,508 byte buffer located on the module.

5.2.3 Ethernet Protocol Module (EPM) Register Access

By using the MODE RDIS or MODE WDIS command as defined in Section 3.5.4, the u-task can address one of the EPM registers. Any READ1 command will be directed to the register addressed by the current read DCB.DOCR value; similarly, any WRITE1 command will be directed to the register addressed by the current write DCB.DOCW value.

It is the responsibility of the u-task to properly address the EPM registers. If the u-task issues a READ1 command, the SVC15 call is terminated with an illegal action status. If the u-task issues a WRITE1 command, the data field byte will be directed to the register addressed by the current value in write DCB.DOCW, which can be any EPM register, depending on previous commands.

To change the example, suppose the u-task issues a MODE WDIS command that modifies the value of write DCB.DOCW to address C_REG. A subsequent WRITE1 will properly be directed to the C_REG. A subsequent READ1, however, will be directed to the EPM register addressed by the current value in read DCB.DOCR that could be any EPM register, depending on previous commands.

It is the responsibility of the u-task to keep track of interrupts that are being enabled or disabled by the modified read DCB.DOCR or write DCB.DOCW values. For example, if a u-task issues a MODE WDIS, which addresses the C_REG and disables interrupts, it then issues a WRITE1 command, specifying a TYPE I EPM command (i.e., single byte status return). The EPM command status will not cause an interrupt, which means that the u-task must issue a MODE RDIS to address the S_REG (EPM status register), followed by a READ1 to read the status byte.

The EPM interface is accessible under program control by means of five addressable 8-bit registers. These registers are listed below.

REGISTER	DESCRIPTOR
C_REG	Command register (write-only)
S_REG	Status register (read-only)
T_DATA	Transmit data register (write-only)
R_DATA	Receive data register (read-only)
H_REG	Handshake control register (read-only)

These registers can be used as the program interface to the EPM.

5.2.4 Continuous Error Processing (CEP)

A u-task requests CEP by setting bit 4 in the SVC15 function code. For programming the Ethernet line driver, this should be considered as a continuous read request (continuous write is not supported). Queued buffers must be used.

Whereas the zero-bit insertion/deletion (ZBID) line driver returns certain error conditions as status halfwords in the bytes-used field of the user buffer (see Section 3.2.2), there is no corresponding use of the bytes-used field of the user buffer for the Ethernet line driver. Therefore, the bytes-used field will contain either a zero if the buffer was not used or a nonzero value indicating the number of bytes used in the user buffer.

In reading frames from the Ethernet interface, the line driver may encounter one or more of the following status conditions:

- Lost frame
- Cyclic redundancy check (CRC) error
- Alignment error

Lost frame status is an indication that one or more frames has been lost "behind" the frame presently being read. This status is treated as a normal status, with the assumption that the lost frames will be detected by higher levels of software. The count of frames lost can be obtained via the REPORT AND RESET STATISTICS command (see Table 4-1).

CRC and alignment errors, whether they occur separately or together, are processed by the driver without using any user buffers or notifying the u-task in any way. Again, the REPORT AND RESET STATISTICS command can be used to determine the occurrence of these status conditions.

Continuous read instructs the driver to obtain more user buffers from the read available list without terminating the original SVC15 READ BUFFER command. In this way, a single READ BUFFER command can initiate the servicing of a continuous stream of incoming frames. It is the u-task's responsibility to remove buffers from the read done list as they are returned by the driver and restore these buffers (after processing) to the read available list, which the driver will access after reading each frame.

If there is no user buffer or insufficient user buffers available when a frame is available to be read from the Ethernet interface, the driver terminates the SVC15 READ BUFFER command with a buffer overrun 1 (hex 8010) error status and returns all buffers to the read done list. The u-task then restores all buffers to the read available list and reissues the READ BUFFER command.

5.3 COMMAND REPERTOIRE FOR THE ETHERNET LINE DRIVER

Table 5-1 lists the Ethernet line driver commands. The asterisk (*) marked commands are unique to the Ethernet line driver because they are executed by special routines within the driver, not by common ITAM/32 subroutines.

TABLE 5-1 ETHERNET LINE DRIVER COMMANDS

COMMAND FAMILY	COMMAND	HEX	DESCRIPTOR
NULL	NOP	0000	No operation
	WAIT	0008	Time-interval wait
	XFER	0010	Transfer command chain
	CXFER	0018	Conditional transfer
CONTROL	EXAMINE	0001	Examine device status
	EXAMINE* READ STATUS	0021	Examine received-frame
	EXAMINE* S_REG	0029	Examine S_REG
READ	READ	0002	Read buffer
	READ1*	000A	Read byte from selected EPM register
WRITE	WRITE	0004	Write buffer
	WRITE IMAGE*	001C	Same as write
	WRITE1*	000C	Write byte to selected EPM register
	WRITE IDLELINE	0024	Write idle line
MODE	TOUT	0006	Change time-out constant
	CMD2	000E	CHANGE PROGRAMMABLE ADAPTER command
	RCMD	0016	CHANGE READ command
	WCMD	001E	CHANGE WRITE command
	RDIS	0026	CHANGE READ-DISABLE command
	WDIS	002E	CHANGE WRITE-DISABLE command
	DISC	0036	CHANGE DISCONNECT command
TEST	TEST I*	000F	TYPE I EPM commands
	TEST II*	0017	TYPE II EPM commands
	TEST III*	001F	TYPE III EPM commands

* Executed by special routines of the Ethernet line driver

NOTE

Any command not listed in this table will receive invalid command status (8018).

5.3.1 NULL-Type Commands

Chain command and command trap flags are valid. There are four NULL-type commands, none of which issue input/output (I/O) instructions to the adapter. They are:

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
NOP	0000 0000	00	1
WAIT	0000 1000	08	1
XFER	0001 0000	10	1
CXFER	0001 1000	18	2

5.3.1.1 NOP Command

The NOP command performs no operation; one data field is fetched but not used. (It must, however, specify a valid program address.) If the chain command (bit 0) is set, the next command is fetched.

5.3.1.2 WAIT Command

The WAIT command is used to suspend driver execution for a specified interval. One data field is fetched; it must point to a halfword containing the value of a time interval, in multiples of 100ms. The driver waits until the specified interval expires and then continues execution (if chain command is set) or terminates (if chain command is reset).

5.3.1.3 Transfer In (XFER) Command

The XFER command is used to specify the next driver command word (DCW) in a chain that does not exist in contiguous memory. One data field is fetched; if a chain command is set, the next command to be executed is fetched from the address contained in the data field. Thus, a branch to a DCW is performed.

5.3.1.4 Conditional Transfer (CXFER) Command

The CXFER command is used to test the internally maintained status of the SVC15 call. Two data fields are fetched. The first data field points to two halfwords, the first of which is logically ANDed with the current state of the ITAM/32 logical status halfword. The result is compared to the second halfword of the pair and if equal, the next command to be executed is specified by the second data field. Otherwise, command execution continues with the next command in the current chain. The CXFER command can be used to test for specific conditions, as indicated by the ITAM/32 logical status. The first halfword of the first data area contains a mask with a 1 in each bit portion to be tested; the second halfword of the first data area contains the value to be tested against. For example, if one or more special characters are detected after a read operation, a different command sequence may be desired.

5.3.2 CONTROL-Type Commands

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
EXAMINE	0000 0001	01	1
EXAMINE READ STATUS	0010 0001	21	1
EXAMINE S_REG	0010 1001	29	1

5.3.2.1 EXAMINE Command

Standard ITAM/32 support will be provided for the EXAMINE command. Note that the device status returned is based on the value stored in the read DCB.DVST (the write DCB.DVST value is undefined at all times).

5.3.2.2 EXAMINE READ STATUS Command

The EXAMINE READ STATUS command returns the value of the latest received frame status byte stored in the read DCB. Note that if CEP is in effect, this command must be issued to the write DCB. However, the command will return the value from the read DCB, regardless of which DCB is accessed.

5.3.2.3 EXAMINE S_REG Command

The EXAMINE S_REG command returns the latest value of S_REG that has been stored in the write DCB; it does not affect the value stored in the write DCB. Whenever the driver reads a value from S_REG, it stores the value read into the write DCB and in certain cases, such as write buffer, it translates the value into a standard ITAM/32 termination code. This command, therefore, allows the u-task to access the untranslated value.

Note that command returns the value from the write DCB, regardless of which DCB is accessed.

5.3.3 READ-Type Commands

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
READ BUFFER	0000 0010	02	1 or 2
READ1	0000 1010	0A	1

5.3.3.1 READ BUFFER Command

The READ BUFFER command supports chained and queued ITAM/32 buffer types. The status field of the SVC15 parameter block contains a standard ITAM/32 SVC15 status code. The actual received frame status byte is stored in the read DCB. Access to this value is directly provided via the EXAMINE READ STATUS command.

Note that read after write lookahead is not supported in this driver.

An outline of frame reception is listed below.

- The u-task issues the READ BUFFER command, supplying a read pool of buffers, typically queued or chained.
- The driver gets the user buffer, enables interrupts on the interface, sets the read pending flag and releases the selector channel (SELCH).
- The incoming frame is buffered on the interface and then interrupts the driver.
- The driver obtains the SELCH and sets the read in progress flag.
- The driver reads the first byte of the read buffer, which is the received frame status byte, and stores it in the read DCB. If it is nonzero, the driver translates it into a standard SVC15 termination code. Note that the EXAMINE READ STATUS command can be used to access the true received frame status byte.

The translation table for received frame status is listed below.

RECEIVED FRAME STATUS	SVC15 TERMINATION CODE
Success	No errors
CRC error	Data check
Alignment error	Invalid frame
Lost frame	Overflow

- The driver reads the next byte, which is a filler byte with a value of zero, and discards it.
- The driver reads the next two bytes which indicate the length of the received frame.
- The driver compares frame length to user buffer length and instructs the SELCH to read, based on the smaller value.
- If the read times-out, the u-task is given a time-out status and the SELCH is stopped.
- When the SELCH interrupts, it is checked for good status and proper ending address for the data transfer.
- The frame length is updated based on the data transfer just completed. If it is zero, the driver waits for the end of medium (EOM) status to interrupt. When it does, a check is made for CEP.
- If the updated frame length is not zero, the driver obtains another read buffer and restarts the SELCH.
- If CEP is specified, the driver places any nonzero error status in the bytes-used field of the user buffer. The driver then obtains another read buffer, releases the SELCH and waits for the next interrupt from the interface, unless another received frame is available. In such a case, it is serviced immediately. The read pending flag is left set, while the read in progress flag is reset.
- If CEP is not specified, the driver schedules termination of the SVC15 READ BUFFER command, releasing the SELCH. The read pending and read in progress flags are reset. The driver then checks the status of the write in progress flag; if set, interrupts are not disarmed on the interface; if not set, interrupts are disarmed on the interface.

5.3.3.2 READ1 Command

The READ1 command returns the current contents of the EPM register addressed by the current value of read DCB.DOCR, with interrupts disarmed; i.e., either the default or the last value placed there directly by the MODE RDIS command or indirectly by a READ BUFFER command.

5.3.4 WRITE-Type Commands

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
WRITE BUFFER	0000 0100	04	1 or 2
WRITE IMAGE	0001 1100	1C	1 or 2
WRITE1	0000 1100	0C	1
WRITE IDLELINE	0010 0100	24	0

5.3.4.1 WRITE BUFFER Command

The WRITE BUFFER command will support chained and queued ITAM/32 buffer types. The status field of the SVC15 parameter field contains a standard ITAM/32 SVC15 status code, translated from the value of S_REG. The actual S_REG contents will be stored in the write DCB. Access to this value is provided directly via the EXAMINE S_REG command.

The following is an outline of frame transmission:

- The u-task issues an SVC15 WRITE BUFFER command, supplying either a chained or queued buffer.
- The driver gets the user buffer, obtains the SELCH, sets the write in progress flag and instructs the SELCH to write the user buffer to the interface.
- The SELCH interrupts the driver when the transfer is complete.
- If there are more write buffers, the driver issues a load transmit data command to the EPM, which causes the u-task data to be buffered on the EPM. The driver then obtains the next write buffer and repeats the step of instructing the SELCH to transfer data to the EPM. If the maximum Ethernet frame size (1,508 bytes) is exceeded, an error status is returned by the EPM.
- If there are no more write buffers, the driver issues a load transmit data and send command to the EPM, which causes the entire frame to be transmitted.

- The interface interrupts the driver when the LOAD TRANSMIT DATA AND SEND command has been completed, with or without success. Note that this interrupt may be preceded by a received frame interrupt from the interface, which will be serviced first. Due to the fact that the frame to be transmitted is separately buffered on the EPM it will only be transmitted when the Ethernet line is available.

- Status available from the S_REG of the EPM, is translated into a standard ITAM/32 termination code for the SVC15 call and stored in the write DCB. Note that the EXAMINE S_REG command can be used to access the untranslated S_REG value.

  The following is the translation table for LOAD TRANSMIT DATA AND SEND:

LOAD TRANSMIT DATA AND SEND STATUS	SVC15 TERMINATION CODE
Success	No errors
Success with retries	No errors
Inappropriate command	Illegal action
Transmitter failure	Illogical status
Buffer size exceeded	Illegal action
Frame too small	Illegal action
Excessive collisions	Time-out

- The driver schedules termination of the SVC15 WRITE BUFFER command, releasing the SELCH, and resets the write in progress flag.

- The driver examines both read in progress and read pending flags. If either flag is set, interrupts are left enabled on the interface (this is the normal operation). If neither flag is set, interrupts are disabled.

5.3.4.2 WRITE IMAGE Command

The WRITE IMAGE command operates identically to the WRITE BUFFER command, since all CRC calculation is done automatically on the EPM.

5.3.4.3 WRITE1 Command

The WRITE1 command transmits the byte specified in the data field to the EPM register addressed by the current value of write DCB.DOCW with interrupts disarmed; i.e., either the default or the last value placed there directly by the MODE WDIS command or indirectly by the WRITE BUFFER command.

5.3.4.4 WRITE IDLELINE Command

The WRITE IDLELINE command is immediately terminated with zero status.

5.3.5 MODE-Type Commands

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
MODE TOUT	0000 0110	06	1
MODE CMD2	0000 1110	0E	1
MODE RCMD	0001 0110	16	1
MODE WCMD	0001 1110	1E	1
MODE RDIS	0010 0110	26	1
MODE WDIS	0010 1110	2E	1
MODE DISC	0011 0110	36	1

5.3.5.1 MODE TOUT Command

The MODE TOUT command is used to set the error time intervals for commands that enable the time-out flag. One data field is fetched containing the address of the first of two halfwords; the first contains an error time-out interval for read-type operations and the second contains an interval for write-type operations (both in units of one second).

The timer is strictly for error detection; when it expires, the command is terminated in error. Interval timing is performed via the WAIT command, which uses a separate (100ms resolution) clock.

NOTE

The resolution of the time is accurate from +0 to -1 second. Timeout values should be set to the desired number of seconds plus 1. To request a time-out interval of 10 seconds, a time-out value of 11 seconds should be specified.

5.3.5.2 MODE CMD2 Command

The MODE CMD2 command is used to specify the device-dependent commands used to set adapter options. One data field is fetched and has the address of a byte containing the device command to be output to specify programmable adapter options such as parity, number of data bits, etc.

5.3.5.3 MODE RCMD and MODE WCMD Commands

The MODE RCMD and MODE WCMD commands are used to specify the device-dependent commands used to set the read or write mode in the adapter. One data field is fetched for these commands, with the address of a byte containing the device command to be output for read or write data transfers, respectively.

5.3.5.4 MODE RDIS and MODE WDIS Commands

The MODE RDIS and MODE WDIS commands are used to specify the device-dependent commands used to set the read or write side of the adapter quiescent. One data field is fetched with the address of a byte containing the device command to be output to set the quiescent state of the read or write side of the line.

5.3.5.5 MODE DISC Command

The MODE DISC command specifies the device-dependent command used to disconnect the adapter from the line. One data field is fetched, with the address of a byte containing the device command to be output to disconnect the communications line (reset data terminal ready (DTR).

5.3.6 TEST-Type Commands

In the Ethernet interface line driver, TEST-type commands are defined to allow the u-task access to all three EPM command types. Note that the modifier field is used to select the EPM command type. Any other modifier value will cause the SVC15 to be terminated with illegal command status.

COMMAND	BINARY	HEX	NUMBER OF DATA FIELDS
TYPE I	0000 1111	0F	1
TYPE II	0001 0111	17	1
TYPE III	0001 1111	1F	1

5.3.6.1 TYPE I Commands

The command modifier should be set to 1. The data field should specify one direct buffer of length one which includes the actual EPM command code. Status will be returned to the SVC15 status field as the untranslated EPM command or diagnostic status. These commands must be issued to the write device.

5.3.6.2 TYPE II Commands

The command modifier should be set to 2. The data field should specify two chained buffers. The first, of length one, specifies the actual EPM command code; the second, which should specify a buffer of sufficient size, will receive the response from the EPM command. If the second buffer is too small or not present, the SVC15 is terminated with buffer limit status. If the command issued generates a zero length status buffer (e.g., report collision delay times issued when no collisions have occurred), a different buffer limit error status is returned. If the second buffer is properly provided, the SVC15 is terminated with the untranslated EPM status code.

5.3.6.3 TYPE III Commands

The command modifier should be set to 3. The data field must indicate two chained buffers. The first, of length one, specifies the actual EPM command code; the second, of appropriate length, supplies the data for the EPM command. Status will be returned to the SVC15 status field as the untranslated EPM command status.

5.3.6.4 Reserved Commands

The reserved commands are terminated with illegal command status which is the same as the standard ZBID line driver for all TEST-type commands.

5.4 CHAINED/QUEUED BUFFER LINK WORD FLAG BYTE

The OS/32 Basic Data Communications Reference Manual details the buffer formats supported by ITAM and the ZBID driver. For chained and queued buffers, the flag byte (bits 0-7) of the buffer link word is used by the Ethernet driver as follows:

BIT NUMBER	0	1	2	3	4 5 6 7
MEANING	Busy	Done	Frame End	Frame Begin	User must clear to zero.

Where:

Busy/Done	is standard ITAM usage; see the OS/32 Basic Data Communications Reference Manual
Frame End	is set on the last or only buffer in a chain.
Frame Begin	is set on the first or only buffer in a chain. This allows many buffers chained together to be written as one frame.

5.5 ERROR HANDLING

Error codes are placed into the status field of the SVC15 parameter block at termination or into the bytes-used field of the user buffer for CEP. Table 5-2 is a list of the standard ITAM/32 termination codes encountered during an Ethernet operation.

TABLE 5-2 ENCODED SVC15 TERMINATION CODES FOR ETHERNET LINE DRIVER

TERMINATION CODE	CAUSE
8002-800C	Invalid for normal Ethernet operation.
800D	Device unavailable or false sync from Ethernet interface or SELCH. Error in system initialization (this error should be retried at least once).
800E,800F	Invalid for normal Ethernet operation.
8010	Buffer overrun 1 - Possible causes are: - Not enough user buffers for read buffer operation - Second user buffer too small for TEST TYPE II command
8011	Number of SVC15 commands executed > 255.
8012	Task queue error - u-task queue is full, non-existent or invalid.

TABLE 5-2 ENCODED SVC15 TERMINATION CODES FOR ETHERNET LINE DRIVER (Continued)

TERMINATION CODE	CAUSE
8013	Buffer overrun 2 - Possible causes are: - Zero-length status buffer available for TEST-TYPE II commands (e.g., report collision delay times) - Event service routine (ESR) doubly scheduled
8014	Time-out on excessive collisions.
8015	Halt I/O - I/O was aborted by a halt I/O.
8016,8017	Invalid for normal Ethernet operation.
8018	Invalid command - Possible causes are: - Examine issued with no read or write device number - Read request to write device code - Write request to read device code - TEST command to read device code - TEST command invalid for EPM
8019	Possible causes are: - SELCH memory malfunction or memory parity fail - SELCH ending address error
801A	Memory fault referencing buffer - Possible causes are: - Invalid buffer type (only chained/queued allowed) - Invalid buffer address or size field - DCB.IFLG corrupted for queued buffer request (IFL.QBFB reset means no queued buffer support) - Invalid buffer address being returned to u-task - Invalid queued buffer list address
801B	Illegal logical unit (lu) - Was not assigned for SVC15 access or unassigned.

TABLE 5-2 ENCODED SVC15 TERMINATION CODES FOR ETHERNET LINE DRIVER (Continued)

TERMINATION CODE	CAUSE
801C	Illogical device status - Possible causes are: - Received frame status byte unrecognizable - EPM command status unrecognizable - EPM command status is unsupported status - EPM command status is transmitter failure
801D	Invalid for normal Ethernet operation
801E	Illegal software condition - Possible causes are: - READ 1 command to EPM write-only register - WRITE 1 command to EPM read-only register - Done bit set twice in user buffer - EPM command status is inappropriate command - EPM command status is buffer size exceeded - EPM command status is frame too small - No user buffer provided on write buffer request - User buffer already marked busy when obtained by driver
801F-8021	Invalid for normal Ethernet operation
8022	Invalid frame - Possible causes are: - Write buffer issued with no frame termination bit set - Received frame status is alignment error - Received frame status is CRC error
8023	Invalid for normal Ethernet operations.
8024	Queue overflow - u-task "To" list is full.

5.6 ETHERNET INTERFACE PROGRAMMING

The following registers are the program interface to the EPM, which include status, command and address registers.

5.6.1 Primary Registers

Status register (STAT 08:15) Read-Only:

STATUS REGISTER	DESCRIPTION
STAT 08	Not used
STAT 09	Not used
STAT 10	SRF1 (status register full)
STAT 11	RDF1 (receive data full)
STAT 12	BUSY1
STAT 13	Not used
STAT 14	EOM1 (end of medium)
STAT 15	Not used

Command Register (CMD 08:15) Write-Only:

COMMAND REGISTER	DESCRIPTION
CMD 08	DIS1 (disable interrupts)
CMD 09	EN1 (enable interrupts)
CMD 10	HW1 (not used, deactivated)
CMD 11	CMD11 REOM (reset EOM)
CMD 12	CMD12 TSTNP (test new protocol)
CMD 13	A2 (secondary register address)
CMD 14	A1 (secondary register address)
CMD 15	A0 (secondary register address)

Address Register (ADR 08:15) Read-Only:

The dual in-line package (DIP) switch contains the address of the Ethernet device controller.

ADDRESS REGISTER	DESCRIPTION
ADR 08:11	First hexadecimal digit
ADR 12:15	Second hexadecimal digit

5.6.2 Secondary Register Definitions

The Secondary registers are the program interface to the EPM and their use is described in greater detail in Chapter 4. They are addressed via the command register as follows:

CMD 13:15	SECONDARY REGISTERS
000	C_REG
001	S_REG
010	T_DATA
011	R_DATA
101	H_REG

5.6.3 Default Output Commands

Ethernet default values for the EPM device commands issued by the Ethernet line driver are as follows:

COMMAND	VALUE
Read Time-out	6 seconds
Write Time-out	5 seconds
Device Read	X'43'
Device Write	X'42'
Disable Read	X'83'
Disable Write	X'C2'
Disconnect	X'43'
Set Programmable Option Command	X'43'

For a list of command register codes, status register codes, diagnostic status codes and received-frame status, see Chapter 4.

5.7 ETHERNET DEVICE CONTROL BLOCK (DCB) FIELDS

The Ethernet DCB is identical for the read (device code 212) and write (device code 213) devices; however, some fields are used differently by the two devices. The following is a brief explanation of the Ethernet-specific portion of the Ethernet DCB:

DCB.WDCB

The DCB.WDCB fullword field is a pointer to the write DCB. It is defined for both read and write DCBs.

DCB.RDCB

The DCB.RDCB fullword field is a pointer to the read DCB. It is defined for both read and write DCBs.

DCB.BSAD

The DCB.BSAD fullword field records the physical address of the start of the current read or write user buffer (depending on the DCB), or zero if there is no current buffer. This field, when combined with the fields DCB.DSAD and DCBA.EAD, constitutes an address triple corresponding to one user buffer.

DCB.DSAD

The DCB.DSAD fullword field records the physical address of the start of the data field of the current read or write user buffer depending on the DCB, or zero if there is no current buffer.

DCB.EAD

The DCB.EAD fullword field records the physical address of the end of the current read or write user buffer depending on the DCB, or zero if there is no current buffer.

DCB.SEND

The DCB.SEND fullword field is used to store the last end address read from the SELCH after a read or write operation, accordingly.

DCB.SDSA

The DCB.SDSA fullword field is used as a temporary storage area for the user buffer data start address (i.e., copy of DCB.DSAD), for use during runt-frame processing.

DCB.SEAD

The DCB.SEAD fullword field is used as a temporary storage area for the user buffer data end address (i.e., copy of DCB.EAD), for use during runt-frame processing.

DCB.FLST

The DCB.FLST storage area is initialized by the driver as a standard circular list and is used to store buffers obtained from the u-task (size of the list is governed by the constant ETH.LSIZ, which is found in the macro $ETHDCBS; default 15). As buffers are required by the driver, they are removed from DCB.DSAD and DCB.EAD, pushed onto DCB.TLST and finally returned to the u-task.

DCB.TLST

The DCB.TLST storage area is initialized by the driver as a standard circular list and is used to store buffers to return to the u-task. (Size of the list is governed by the constant ETH.LSIZ, which is found in the macro $ETHDCBS; default 15.) As buffers are required by the driver, they are removed from DCB.BSAD, DCB.DSAD and DCB.EAD, pushed onto DCB.TLST and finally returned to the u-task.

DCB.TRAC

The DCB.TRAC storage area is initialized by the driver as a standard circular list; this list is not normally accessed. To enable the logic that will trace the most recent ETH.TSIZ items (where ETH.TSIZ is a constant in $ETHDCBS; default 100), where "items" includes all ESR and interrupt service routine (ISR) entries and exits, two instructions must be no-oped in the driver; namely, the first instructions in the driver routines MAKTRACE and MAKTRACI.

DCB.SLNK

The DCB.SLNK fullword field is used as a temporary save area for the link register.

DCB.EER1

The DCB.EER1 fullword field records the link register of the ESR error exit call (i.e., the BAL to ESR.EXIT); as such, this field is useful for determining exactly where in the driver an error termination originated.

DCB.EER2

The DCB.EER2 halfword field records the link register of the ISR error exit call (i.e., the BAL to ERR.EXIT); as such, this field is useful for determining exactly where in the driver an error termination originated.

DCB.SISR

The DCB.SISR halfword field records the link register for normal ISR exit (i.e., BAL to ISR.EXIT) and as such, is a simple trace mechanism for determining the last ISR to be exited on the respective read and write sides. More extensive trace capability is available using the DCB.TRAC storage area.

DCB.ETB

The DCB.ETB halfword field is used to schedule ESR subroutines via the ESR Executive (ISSEXECE). Definition of the bits in this field is found in the macro $ETHDOBS.

DCB.DLEN

The DCB.DLEN halfword field adds the total data area represented by the buffers stored in DCB.FLST. This value is used particularly by the read side to determine if enough user buffer space is available to read the incoming frame.

DCB.FLEN

In the read DCB, the FLEN halfword field records the number of bytes available to be read in the currently available frame on the EDLC. After a frame is read, this field is zero. In the write DCB, this field undefined.

DCB.RLEN

In the read DCB, the RLEN halfword field records the total size of the currently available frame on the EDLC. During a normal read operation, this field is equal to DCB.FLEN initially, then DCB.FLEN is updated as data is read into user buffers. In the write DCB, this field is undefined.

DCB.ETFL

The DCB.ETFL halfword field is used to coordinate the read and write operations. The definition of the bits in this field is found in the macro $ETHDCBS. This field is only defined in the read DCB.

DCB.ESDN

The DCB.ESDN halfword field is set up by SYSGEN/32 to designate the address of the SELCH used by the EDLC.

DCB.SLEN

In the write DCB, the SLEN halfword field is used to record the number of bytes available for a TYPE II (TEST-type) command; namely, a statistics-gathering command. In the read DCB, this field is undefined.

DCB.NM10

The DCB.NM10 byte field records the last TYPE I, TYPE II or TYPE III command that is issued to the EPM module on the EDLC. This field is only defined for the write DCB.

DCB.SREG

In the write DCB, the SREG byte field records the last S_REG value read from the EDLC. This value is reported to an Examine S_REG command. In the read DCB, this field is undefined.

DCB.RFSB

In the read DCB, the RFSB byte field records the most recent received frame status byte read from the EDLC, indicating status of the incoming Ethernet frame. This value is reported to the EXAMINE READ STATUS command. In the read DCB, this field is undefined.

DCB.HREG

In the read DCB, the HREG byte field records the most recent contents of the EPM H_REG register ("handshaking" register). The definition of the bits in this field are found in the macro $ETHSTCM. In the read DCB, this field is undefined.

APPENDIX A
SUMMARY OF CHANGES TO ZERO-BIT INSERTION/DELETION (ZBID) LINE DRIVER INTERFACE

All commands, whether or not supported by the standard ZBID line driver, are respectively handled by the Ethernet interface line driver, with the following exceptions:

- READ AFTER WRITE is not supported.
- No direct or indirect buffer support is provided.
- EXAMINE READ STATUS is supported.
- EXAMINE S_REG is supported.
- RING-WAIT, ANSWER and DISCONNECT are not supported.
- READ1 is supported in order to allow user task (u-task) access to Ethernet protocol module (EPM) registers.
- SELECTIVE READ is not supported.
- WRITE1 is supported in order to allow u-task access to EPM registers.
- WRITE IMAGE is identical to WRITE BUFFER.
- WRITE IDLELINE is not implemented.
- MODE ADDR is not supported.
- TEST-type commands are supported in order to allow u-task access to EPM commands.

INDEX

Z

PERKIN-ELMER

PUBLICATION COMMENT FORM

We try to make our publications easy to understand and free of errors. Our users are an integral source of information for improving future revisions. Please use this postage paid form to send us comments, corrections, suggestions, *etc.*

1. Publication number _____

2. Title of publication _____

3. Describe, providing page numbers, any technical errors you found. Attach additional sheet if neccessary.

   _____

   _____

   _____

4. Was the publication easy to understand? If no, why not?

   _____

5. Were illustrations adequate? _____

   _____

   _____

6. What additions or deletions would you suggest? _____

   _____

   _____

7. Other comments: _____

   _____

   _____

From _____ Date _____

Position/Title _____

Company _____

Address _____

_____

_____

STAPLE STAPLE

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 22 OCEANPORT, N.J.

POSTAGE WILL BE PAID BY ADDRESSEE

PERKIN-ELMER

Data Systems Group
106 Apple Street
Tinton Falls, NJ 07724

ATTN:
TECHNICAL SYSTEMS PUBLICATIONS DEPT.

FOLD FOLD

STAPLE STAPLE

6433-1