KL10 Maintenance Guide

Company Confidential

Volume II

digital

EK-OKL10-MG-003

KL10 Maintenance Guide
Volume II

Prepared by Educational Services
of
Digital Equipment Corporation
Marlborough, MA

1st Edition, February 1979
2nd Edition, December 1983
3rd Edition, April 1985

Printed in U.S.A.

ORDERING PROCEDURES

1. KL10/KS10 Maintenance Guides

To order one or both of these guides (if you have never been on the distribution list), send your order by TWX or memo to:

Publishing & Circulation Services
Order Processing
NR0-2/W3
TWX RCS Code NR12

Specify the following:

Name
Badge Number
Cost Center
Location
Quantity: 1 (only one copy of each guide per name)
EK-OKL10-MG (KL10 Maintenance Guide)
EK-OKS10-MG (KS10 Maintenance Guide)

2. KS10 Maintenance Guide Volume II

If you have Volume I of the KS10 Maintenance Guide, but did not receive Volume II (Software and Diagnostics), distributed June 1980, then your name may not be in the distribution list for that guide. To order Volume II, send a TWX or memo to:

Publishing & Circulation Services
Order Processing
NR0-2/W3
TWX RCS Code NR12

Specify the following:

PLEASE ADD MY NAME TO THE DISTRIBUTION LIST

Name
Badge Number
Cost Center
Location
EK-OKS10-MG (Volume II only)

3. Deletions and Transfers

To be deleted from the distribution list or to transfer your subscription to someone else, send a TWX or memo to:

Publishing & Circulation Services
List Maintenance
NR0-3/M1
TWX RCS Code NR12

Specify the following:

Remove: Name
Badge Number
Cost Center
Location

Add: New Name
Badge Number
Cost Center
Location

State which Maintenance Guide you want to stop receiving or to transfer:

EK-OKL10-MG (KL10 Maintenance Guide)
or
EK-OKS10-MG (KS10 Maintenance Guide)

To the Reader:

IMPORTANT - This guide contains information for internal use only and is intended for use by DIGITAL Field Service engineers only. Refer to the Field Service Methods and Procedures Manual for company policy pertaining to internal information.

OBJECTIVE - The objective of this guide is to organize and present the maintenance information necessary to resolve 80% of all KL10 hardware malfunctions.

To properly maintain and improve this guide in subsequent revisions, we need feedback concerning accuracy and clarity. This communication is very helpful to your fellow engineers. Please forward any corrections, suggestions, and comments that would improve this guide to:

Customer Services, Systems Engineering (CSSE)
RE: KL10 Maintenance Guide
MR01-1/S35

ORGANIZATION - Volume I contains general maintenance information pertaining to the KL10. The volume is divided into sections with separate Table of Contents as follows.

1. GENERAL INFORMATION consists of miscellaneous maintenance information that cannot be classifed and filed in any of the other hardware sections.

2. SWITCHES AND JUMPERS contains information pertaining to hardware switch positions and jumper connections.

3. TABLES AND MAPS describes the process tables and bit maps associated with the KL10 mainframe and peripheral equipment.

4. CHECKS AND ADJUSTMENTS consists of check and adjustment procedures performed during preventive and corrective maintenance.

5. DIAGRAMS AND MULS contains block diagrams, power supply layouts, and module utilization lists associated with KL10-based systems.

6. MULTI-CPU contains maintenance and diagnostic information specific to multiprocessor systems.

7. DECnet-10/20 contains system hardware and software information.

The information in each hardware section is arranged according to unit and subsystem (i.e., CPU, memory, disk, tape, and I/O).

Volume II contains additional hardware and software information related to the KL10. The volume is divided into tabbed sections with separate tables of contents as follows.

1. COMPUTER INTERCONNECT contains descriptions of the card cage, module locations, switch settings, bit/error formats, diagnostics, and label information.

2. NETWORK INTERCONNECT provides descriptions of the card cage, module locations, switch setting, bit/error formats, diagnostics, and label information.

3. HSC SUBSYSTEM consists of RA81, RA60, and HSC50 Error Codes.

4. CLUSTER TROUBLESHOOTING includes procedures for fault isolation on the cluster level.

5. RP07/RP20 presents RP07 registers and RP20 FSC, jumper, routines, error stops, and other information.

6. S/X BUS contains a genral description, installation, and operation information.

7. ARM-10LS provides installation, operation, and memory fault isolation information.

8. MAINTENANCE SOFTWARE consists of information related to the DIACON, KLDCP, KLDCPU, MEMCON, TRACON, DIAMON, DDT, D20MON programs.

9. SYSTEM SOFTWARE includes information on typical operating systems and command formats.

10. RSX-20F presents information on programs SYSLIB-20F and PARSER, in addition to stop/error codes.

11. TOPS-10 supplies information on TOPS-10, DECnet-10, GALAXY-10, and PIP programs.

12. TOPS-20 contains system program, command summary/format, and error message information.

13. NOTES provides blank pages for note taking.

COMPUTER INTERCONNECT

Table of Contents

Page

DOCUMENTATION
Refer to the CI20 Reference Manual (Document No. EK-0CT20-TM) for functional/logic/installation descriptions.

The Print Set can be ordered as follows:

Order No.	Print Set
MP01903	CI20-A (KL10-E)
MP01906-01	CI20-B (KL10-D)
MP01909-01	CI20-C (KL10-R)

CI CARD CAGE

REAR PANEL CONNECTORS

CARD CAGE:

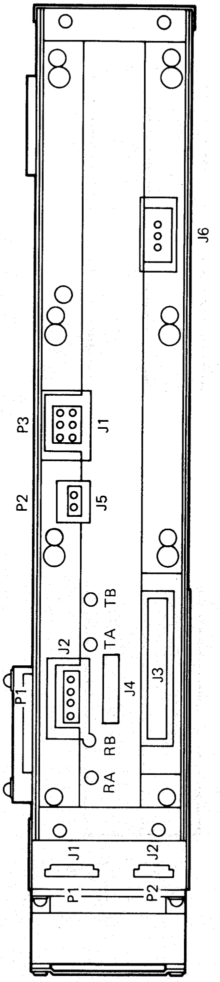
J1	DC POWER +5.0 VOLTS, GND
J2	DC POWER −5.2 VOLTS, GND
J3	PLI
J4	FOR NIA USE ONLY
J5	VOLTAGE MONITOR FOR +5.0 VOLTS
J6	VANE SWITCH
TB	TRANSMIT PATH B
BA	TRANSMIT PATH A
RB	RECEIVE PATH B
RA	RECEIVE PATH B

FAN SUBASSEMBLY:

J1	VANE SWITCH
J2	FAN AC

REAR VIEW

TOP

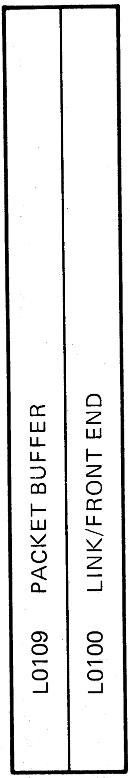


BOTTOM

MR-14240

INTERNAL SLOT ASSIGNMENTS

TOP



CI CARD CAGE
(OPENED FRONT DOOR VIEW)

MUL DECAL
(LOCATED ON SIDE DOOR)

MODULE	LOCATION
L0109	LEFT
L0100	RIGHT

MR-14264

MODULE LOCATIONS
RH/DTE/CI/NIA

The following CI modules are located as follows:

Slot	Module
13	M3001 EBUS INTERFACE/PORT ALU
14	M3002 PORT MICROPROCESSOR CONTROL
15	M3003 CBUS/PLI INTERFACE
16}	
17}	BLANK MODULE ASSEMBLY
18}	

Refer to Volume I DIAGRAMS MULS for all other slot assignments.

MBOX/EBOX

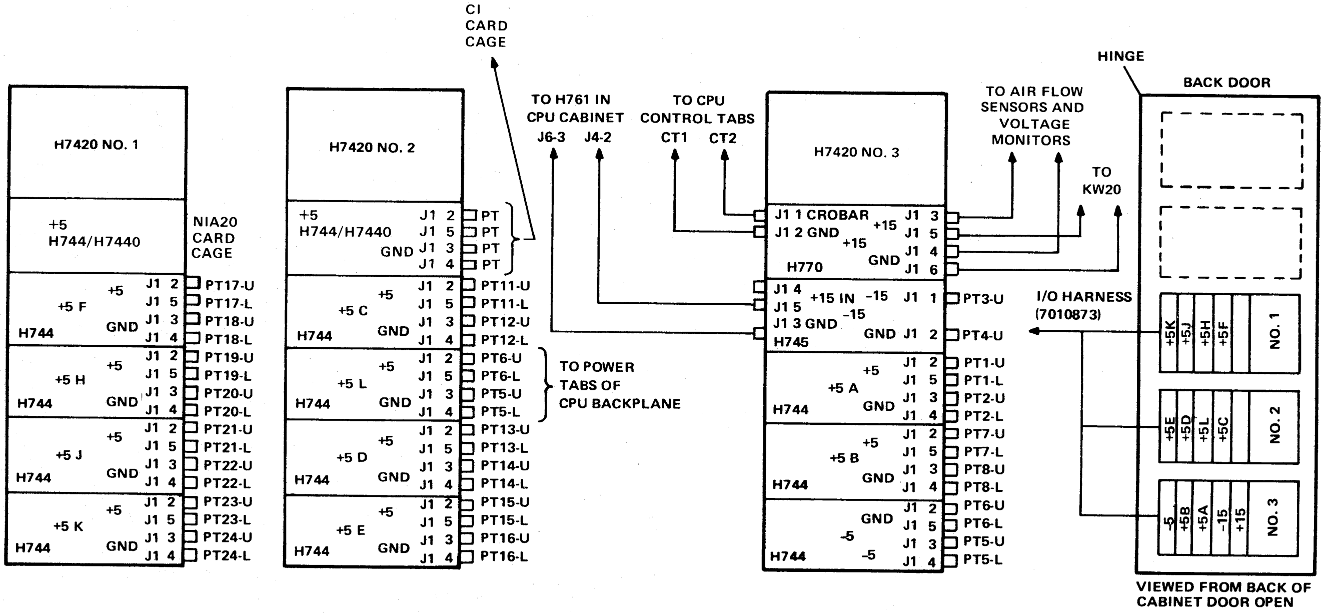
Slot	Module
31	M8532-YA PI BOARD PIC

GENERAL POWER SUPPLY SPECIFICATIONS

The following voltage measurements are to be made from the backplane of the option.

POWER SUPPLY TYPE	OUTPUT	TOLERANCE MAXIMUM	MINIMUM	MAXIMUM RIPPLE IN MILLIVOLTS
702		VARIABLE		600
703	+10	+11.0	+9.4	300
	+1.8	+1.9	+1.7	N/A
	-3.0	-3.15	-2.85	N/A
705	+10	+11.0	+9.4	300
	-15	-16.5	-14.5	700
706	+50	+54.0	+49.0	1500
723	+8	+9.5	+7.8	600
725	-15	-18.0	-14.7	900
728	+10	+11.0	+9.5	700
	-15	-16.0	-14.5	700
732		VARIABLE		600
739	+53	+55	+52	250
	+65	+65	+63	250
742/7420	+25 V	+30	+20	N/A
	-15	-16.5	-13.5	N/A
	+3	+3.5	+2.5	N/A
744/7440	+5	+5.05	+4.95	150
745	-15	-15.05	-14.95	450
754	+20	+20.2	+19.8	450
	-5	-5.05	-4.95	150
761	-2	NONE		
	-5.2	NONE		
770	+15	+15.05	+14.95	450
778	-15	-16.5	-14.5	700
7131/7131A	+5	+5.07	+4.93	50
	+12	+12.18	+11.82	100
	-2	-2.03	-1.97	50
	-5.2	-5.28	-5.12	50

KL10 I/O DC POWER (DISTRIBUTION)



MR-14239

SWITCH SETTINGS
L0100 LINK/FRONT END MODULE

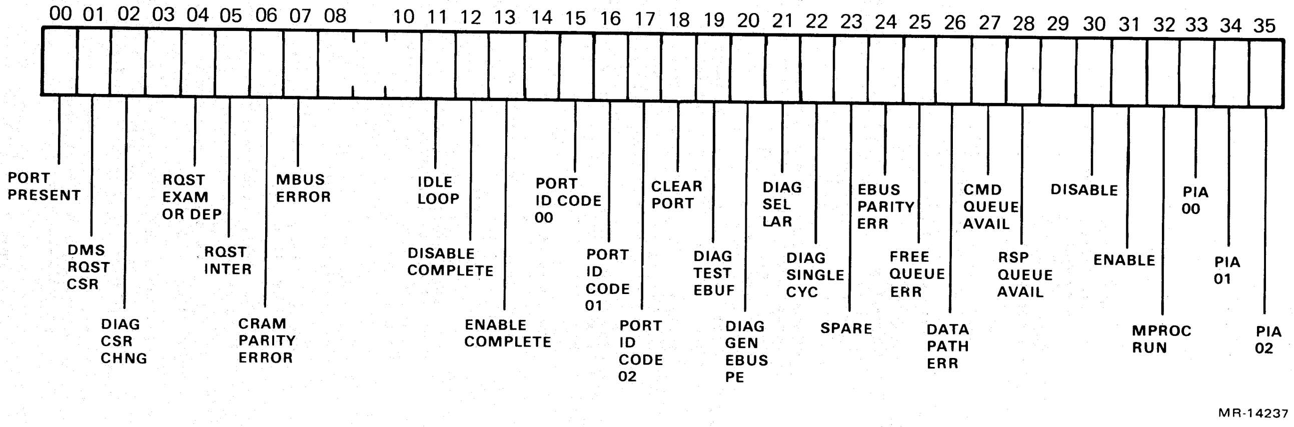
0=OFF (CLOSED)
1=ON (OPEN)

Node	Switch Setting S1	S2	S3	S4	S5	S6	S7	S8
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0
3	1	1	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0
5	1	0	1	0	0	0	0	0
6	0	1	1	0	0	0	0	0
7	1	1	1	0	0	0	0	0
8	0	0	0	1	0	0	0	0
9	1	0	0	1	0	0	0	0
10	0	1	0	1	0	0	0	0
11	1	1	0	1	0	0	0	0
12	0	0	1	1	0	0	0	0
13	1	0	1	1	0	0	0	0
14	0	1	1	1	0	0	0	0
15	1	1	1	1	0	0	0	0

DC VOLTAGE MONITOR BOARD 5414506-01

Switch 1 should be on.
All other switches should be off.

Control and Status Register (CSR) Bit Map



CSR BIT DEFINITIONS

BIT	NAME	DEFINITION
00	PORT PRESENT	Indicates to the KL10 that the port is present (installed and powered-up).
01	DIAG RQST CSR	When set, this diagnostic bit indicates that the port has requested access to the CSR.
02	DIAG CSR CHNG	This diagnostic bit indicates that the contents of the CSR have changed since it was last read by the port micrporocessor.
03	UNUSED	Not used by either the port microprocessor or the KL10.
04	RQST EXAM OR DEP	Used by the port microprocessor to request an EBus interrupt on PI level 00 (Examine or Deposit function). The setting of this bit immediately generates the interrupt request.
05	RQST INTERRUPT	Used by the port microprocessor to request an EBus interrupt on PI levels 01 through 07. The setting of this bit immediately generates the interrupt request.
06	CRAM PAR ERR	Indicates that a control RAM (CRAM) parity error has been detected. If this bit is set, the port microprocessor will be immediately halted and RQST INTERRUPT (CSR bit 05) will be set. A hardware nonvectored (40 + 2n) interrupt will be forced. A CRAM PAR ERR may be forced in order to halt the port microprocessor at a specific location (break point). The port microprocessor cannot be restarted (CSR bit 32 set) until this bit is cleared.
07	MBUS ERR	Indicates that more than one MBUS driver has been turned on at the same time. That is, more than one set of port logic is trying to drive the MBUS at the same time. If this bit is set, the port microprocessor will be immediately halted and RQST INTERRUPT (CSR bit 05) will be set. A hardware nonvectored (40 + 2n) interrupt will be forced. The port microprocessor cannot be restarted (CSR bit 32 set) until this bit is cleared.
08	UNUSED	Neither the port microprocessor or the KL10 use this bit.
09	UNUSED	Neither the port microprocessor or the KL10 use this bit.
10	UNUSED	Neither the port microprocessor or the KL10 use this bit.
11	IDLE LOOP	Indicates that the port microprocessor is in the Idle Loop, and is not "hung" in some other microcode routine.
12	DISABLE COMPLETE	Informs the KL10 that the port microprocessor has placed itself in the DISABLED state.
13	ENABLE COMPLETE	Informs the KL10 that the port microprocessor has placed itself in the ENABLED state.
14	UNUSED	Neither the port microprocessor or the KL10 use this bit.

CSR BIT DEFINITIONS (Cont)

BIT	NAME	DEFINITION
15 16 17	PORT ID CODE 00 PORT ID CODE 01 PORT ID CODE 02 Hardwired so that:	Three-bit PORT IDENT CODE field. Informs software that this is a CI20 port and not an RH20 controller. 00 = 0 01 = 1 02 = 1
18	CLEAR PORT	When set by the KL10, this bit resets the port. The microprocessor is halted and all pertinent registers and control logic are placed in a reset state. The bit clears itself after the reset function is completed.
19	DIAG TEST EBUF	This diagnostic bit enables the KL10 to do an EBus interface loopback function by loading and reading the EBus buffer (EBUF). If the port is not running (CSR bit 32 is reset) and this bit is set, then a KL10: DATAO loads EBus data into the EBUF. DATAI places EBUF data on the EBus.
20	DIAG GEN EBUS PE	This diagnostic bit enables the KL10 to test the EBus parity checker by forcing it to decode an EBus parity error. When this bit is set, EBUS PAR ERR (CSR bit 24) will also be set on the same CONO, assuming there was no real EBus parity error.
21	DIAG SEL LAR	This diagnostic bit enables a KL10 DATAI to read the CRAM address, contained in the Latch Address Register (LAR). If this bit is set and bits 19 and 32 are reset, then the DATAI will cause the LAR contents to be asserted on EBus D01-D12.
22	DIAG SINGLE CYC	This diagnostic bit enables the port microprocessor to be single cycled. If this bit is set and the KL10 sets MPROC RUN (CSR bit 32), the port microprocessor will execute one microcycle and halt. MPROC RUN will be cleared when the microprocessor halts. The current address to be executed is fetched from the RAM Address Register (RAR). The next address to be executed is stored in the LAR at the completion of the microcycle. The KL10 must read the address from the Latch Address Register (LAR) and load it into the RAR before executing the next single cycle.

CSR BIT DEFINITIONS (Cont)

BIT	NAME	DEFINITION
23	SPARE	**Reserved for future software use.**
24	EBUS PARITY ERR	When read by the KL10, this bit indicated that an EBus parity error has been detected. When written as a "1" by the KL10, this bit will clear itself and CRAM PARITY ERR (CSR bit 06).
25	FREE QUEUE ERR	Used by the port to inform the Port Driver that there are no free queue entries available on either the Datagram Free Queue or the Message Free Queue.
26	DATA PATH ERR	Informs the Port Driver that the port microprocessor has detected an error in the DMA data path.
27	CMD QUEUE AVAIL	Used by the Port Driver to inform the port that it has placed a command queue entry on a previously empty command queue.
28	RESP QUEUE AVAIL	Used by the port to inform the Port Driver that it has placed an entry on the previously empty Response Queue.
29	UNUSED	Not used by either the port microprocessor or the KL10.
30	DISABLE	Used by the Port Driver to tell the port to place itself in the DISABLED state (set CSR bit 12).
31	ENABLE	Used by the Port Driver to tell the port to place itself in the ENABLED state (set CSR bit 13).
32	MPROC RUN	When set by the KL10, this bit causes the CRAM Control Register to reset and enables the port microprocessor clocks. The port will start cycling at the address contained in the RAM Address Register (RAR). The next and subsequent addresses will be fetched from the Am2910 sequencer.
33 34	PIA00 PIA01 PIA02	Three-bit KL10 EBus Physical Interrupt Assignment (PIA) field (PI level 01 35 through 07).

PORT CONTROL BLOCK

0	BUFFER DESCRIPTOR TABLE STARTING ADDRESS
1	MESSAGE FREE QUEUE ENTRY LENGTH
2	DATAGRAM FREE QUEUE ENTRY LENGTH
3	RESERVED
4	COMMAND QUEUE 3 INTERLOCK
5	COMMAND QUEUE 3 FLINK
6	COMMAND QUEUE 3 BLINK
7	COMMAND QUEUE 2 INTERLOCK
8	COMMAND QUEUE 2 FLINK
9	COMMAND QUEUE 2 BLINK
10	COMMAND QUEUE 1 INTERLOCK
11	COMMAND QUEUE 1 FLINK
12	COMMAND QUEUE 1 BLINK
13	COMMAND QUEUE 0 INTERLOCK
14	COMMAND QUEUE 0 FLINK
15	COMMAND QUEUE 0 BLINK
16	RESPONSE QUEUE INTERLOCK
17	RESPONSE QUEUE FLINK
18	RESPONSE QUEUE BLINK
19	MESSAGE FREE QUEUE INTERLOCK
20	MESSAGE FREE QUEUE FLINK
21	MESSAGE FREE QUEUE BLINK
22	DATAGRAM FREE QUEUE INTERLOCK
23	DATAGRAM FREE QUEUE FLINK
24	DATAGRAM FREE QUEUE BLINK
25	RESERVED
26	RESERVED
27	RESERVED
28	RESERVED
29	PORT ERROR WORD 0
30	PORT ERROR WORD 1
31	PORT ERROR WORD 2
32	PORT ERROR WORD 3
33	PORT ERROR WORD 4
34	PCB BASE ADDRESS
35	PI LEVEL
36	CHANNEL LOGOUT WORD 1 ADDRESS
37	CHANNEL COMMAND WORD
38	RESERVED TO PORT

MR-14247

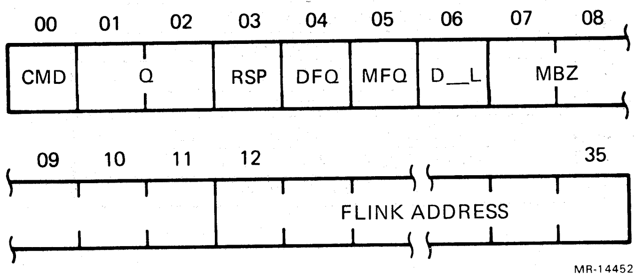
Port Control Block Content

Word	Description
0	Buffer descriptor table (BDT) starting address: The KL10 physical memory address of the first word of the buffer descriptors. Buffer descriptors contain the information needed to tell the port where and how to access a data buffer in KL10 memory.
1	Message free queue entry length: The maximum number of words allowed in an entry on a message free queue. This is a software restriction.
2	Datagram free queue entry length: The maximum number of words allowed in an entry on a datagram free queue.
3	Reserved
4-24	Queue Interlock words, FLINKs, and BLINKs.
25-28	Reserved
29-33	Port error words 0, 1, 2, 3, and 4. The port writes the error words when it encounters a fatal error, writing as much information as possible directly into KL10 memory.
34	PCB base address: The KL10 physical memory address of PCB word 0. The CI20 has no other way to find the PCB.
35	The priority interrupt (PI) level assigned to the CI20
36	Channel logout word 1 address
37	Channel command word (CCW): The port writes a CCW-style word in this location in order to transfer data over the KL10 CBus. The port driver software writes a channel jump word in the EPT location corresponding to the RH20 position that the CI20 occupies.
38	Reserved to the port microcode.

Error Words 0,1 (words 29,30) are written by the port when it encounters fatal errors associated with queue manipulation. This error reporting strategy requires the port to write as much information as possible directly into the host memory. This approach requires the smallest subset of port hardware and microcode to be working to report these errors.

The information in these words provides sufficient data for the port driver to determine the type of error and where the error occurred. When the error is detected, the port writes the contents of the error words in the PCB, enters the disabled state, and generate a host interrupt.

The format of error word 0 is shown in the following figure and described in the following table.

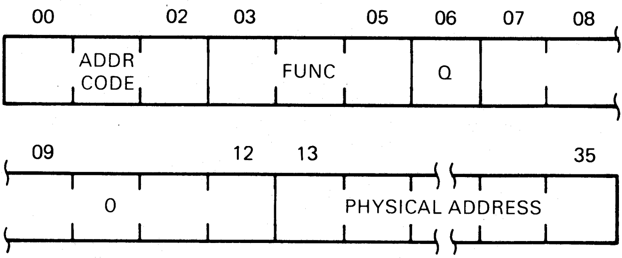


Bit Map, Error Word 0

Error Word 0 Bit Descriptions

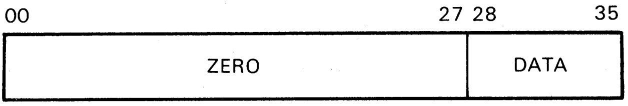
BITS	NAME	DESCRIPTION
0	CMD	Error occurred while touching a command queue entry. The queue with the error is in QUEUE.
1-2	QUEUE	The command queue that had the error. These bits are valid only if the CMD bit is on. 00 = CMD QUEUE 0 01 = CMD QUEUE 1 10 = CMD QUEUE 2 11 = CMD QUEUE 3
3	RSP	Bit is on if error occurred while port was attempting to build a response queue entry.
4	DFQ	Bit is on if error occurred while port was touching a command on the datagram free queue.
5	MFQ	Bit is on if error occurred while port was touching a command on the message free queue.
6	D_L	Bit is on if error occurred while port was linking a command to a queue. Bit is off if error occurred while port was delinking a command from a queue. Bit is valid only with bits 0,4, and 5.
7-11	MBZ	Bits will be zero.
12-35	FLINK ADR	Address of the FLINK word of the queue entry in question.

Error word 1 (word 30) contains the API function word that the port processor used to access memory when the memory error occurred. This word is written here in the same format as it appeared on the EBUS. The format of this word is given in the following figure.



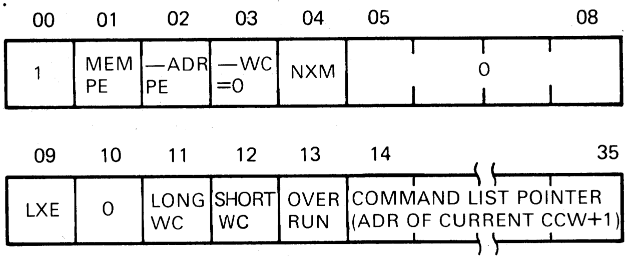
MR-14257

Error word 2 (word 31) contains the register data on transmitter or receiver spurious attention. The format of error word 2 is given in the following figure.



MR-14252

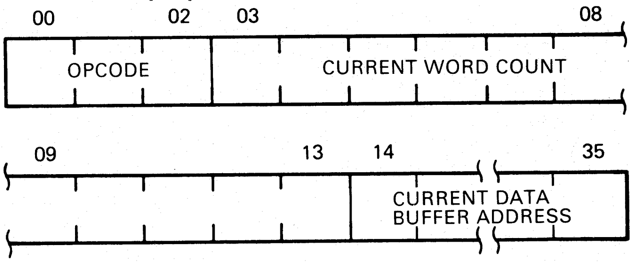
Error word 3 (word 32) contains the channel logout word 1 written by the port on any kind of channel error detected during or immediatly after a DMA transfer. The format of error word 3 is given in the following figure and described in the following table.



MR-14253

Bits	Name	Description
01	MEM PE	Memory parity error
02	-ADR PE	Not address parity error
03	-WC=0	Channel word count did not = 0 when channel did a store to EPT
04	NXM	Channel reference did not exist in memory
09	LXE	Error detected after port term transfer, channel aborts next transfer
11	LONG WC	Port completed transfer, but word count in CCW not reached
12	SHORT WC	Channel transferred data specified by CCW, but port still has data
13	OVER RUN	If device read, port sent data but channel buffers were full. If device write, port req data but channel buffers were empty

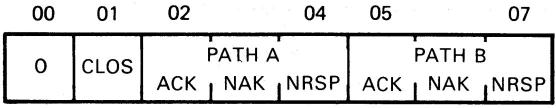
Error word 4 (word 33) contains channel logout word 2 written by the port on any kind of channel error detected during or immediately after DMA transfer. The format of error word 4 is given in the following figure.



MR-14254

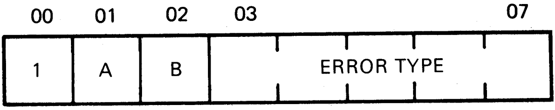
STATUS FIELD

The STATUS field is updated by the port when it builds a response queue entry. The various valid values of the STATUS field are defined below. Note that bit 0 of the STATUS field defines the definition of the remaining bits.



MR-14259

BIT	NAME	DESCRIPTION
1	CLOS	A packet had a retry failure on a path but was transmitted successfully on the other path. The path that failed and the type of failure is indicated in the Path bits. The indicated path is also marked as being bad in the VCDT (Virtual Circuit Descriptor Table).
2	PATH A ACK	The packet was ACKed on this path.
3	PATH A NAK	The packet was NAKed at least once on this path.
4	PATH A NRSP	The packet received No ReSPonse at least once on this path.
5	PATH B ACK	The packet was ACKed on this path.
6	PATH B NAK	The packet was NAKed at least once on this path.
7	PATH B NRSP	The packet received No ReSPonse at least once on this path.



MR-14258

BITS	NAME	DESCRIPTION
1	PTH_A	The error is associated with path A.
2	PTH_B	The error is associated with path B.
3-7	ERROR TYPE	

NO PATH ERRORS

(402)	ERROR = 1 => Access Control violation.
(404)	ERROR = 2 => Invalid Buffer Name.
(406)	ERROR = 3 => Buffer Length violation.
(410)	ERROR = 4 => Packet size violation.
(414)	ERROR = 6 => Local unrecognized command.
(416)	ERROR = 7 => Internal port hardware error.
(420)	ERROR = 10 => Invalid Remote port.
(422)	ERROR = 11 => CRC error reported on received packet.
(424)	ERROR = 12 => No legal path.
(426)	ERROR = 13 => Command not legal in disabled state.
(430)	ERROR = 14 => PLI data PE in SRC byte.
(432)	ERROR = 15 => PLI data PE in OPC byte.
(434)	ERROR = 16 => PLI data PE in body.
(436)	ERROR = 17 => Port disabled during processing.

Path Errors B

(502)	ERROR = 41 => Remote unrecognized command
(504)	ERROR = 42 => Virtual Cicuit closed
(506)	ERROR = 43 => Retries Exhausted (NAK)
(510)	ERROR = 44 => Retries Exhausted (NRSP)
(512)	ERROR = 45 => Transmitter Timeout

PATH A Errors
=============

(602) ERROR = 101 => Remote unrecognized command
(604) ERROR = 102 => Virtual Cicuit closed
(606) ERROR = 103 => Retries Exhausted (NAK)
(610) ERROR = 104 => Retries Exhausted (NRSP)
(612) ERROR = 105 => Transmitter Timeout

PATHS A,B Errors
================

(704) ERROR = 142 => Virtual Cicuit closed
(706) ERROR = 143 => Retries Exhausted (NAK)
(710) ERROR = 144 => Retries Exhausted (NRSP)
(712) ERROR = 145 => Transmitter Timeout

PORT PERFORMANCE MONITORING

The port microcode implements several counters which are under the control of the port driver. The command queue entry Set Counters (SETCNT) allows the port driver to point and/or clear the counters. It also allows the port driver to enable or disable the event counting. There is a mask that is used to control the loading and enabling of the various event counters. For each counter, there are 2 bits in the mask; the first bit enables the counting of the event, and the second bit controls the clearing of the event counter. The port driver may instruct the port to count events for a specified port or a cumulative count for all ports.

The format of the SETCNT command is:

FLINK					0
BLINK					1
RESERVED FOR SOFTWARE					2
00 ↔ 07 STATUS	08 ↔ 15 FLAGS	16 ↔ 23 OPC	24 ↔ 31 MBZ	32 ↔ 35 MBZ	3
00 ↔ 31 XCT_ID				32 ↔ 35 MBZ	4
00 ↔ 31 XCT_ID				32 ↔ 35 MBZ	5
00 ↔ 19 MASK		20 ↔ 31 MBZ		32 ↔ 35 THRESH	6
00 ↔ 23 MBZ			24 ↔ 31 PORT	32 ↔ 35 MBZ	7
RESERVED FOR SOFTWARE					QUEUE LENGTH − 1

MR-14249

WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 201 octal (SETCNT).
6:0-19	MASK	This is the 18 bit mask used to control the enabling and loading of the counters.
6:0	PTH_A ACK	If on, count ACKs received on Path A.
6:1	PTH_A ACKC	If on, clear the counter.
6:2	PTH_A NAK	If on, count NAKs received on Path A.
6:3	PTH_A NAKC	If on, clear the counter.
6:4	PTH_A NRSP	If on, count NO_RSPs received on Path A.
6:5	PTH_A NRSPC	If on, clear the counter.
6:6	PTH_B ACK	If on, count ACKs received on Path B.
6:7	PTH_B ACKC	If on, clear the counter.
6:8	PTH_B NAK	If on, count NAKs received on Path B.
6:9	PTH_B NAKC	If on, clear the counter.
6:10	PTH_B NRSP	If on, count NO_RSPs received on Path B.
6:11	PTH_B NRSPC	If on, clear the counter.
6:12	DG DISCARDED	The count of discarded datagrams because of no DGFree Queue entries.
6:13	DG DISC CLR	If on, clear the counter.
6:14	XMT CNT	Count the packets transmitted to the designated port.
6:15	XMT CLR	If on, clear the counter.
6:16	RCV CNT	Count the packets received from the designated port.
6:17	RCV CLR	If on, clear the counter.

6:18	ERR_CNTR_CLR	If on, clear all error counters (see CNTRD response).
6:19	SET_THRESH	If on, load Port Recoverable Error Threshold value.
6:32-35	THRESH_VAL	Value to load for Port Recoverable Error Threshold.
7:24-31	PORT	This is the designated port for which the above counters will be tracked. If the port value is set to 255, then the counting will be done for all ports.

If the R (response) bit is set in the Set Counters command (SETCNT) it will be placed on the Response Queue instead of the DGFree Queue as a counters Set (CNTSET) command. The format for a Counters Set (CNTSET) command is identical to the Set Counters (SETCNT) command.

Every time the port enters the Enabled state, it will clear all of the counters and set the PORT field to the "all ports" value. The port driver reads these counters, with a Read Counters (RDCNT) command. This command will return the information in the various counters.

The format of a Read Counters (RDCNT) command is:

00–07	08–15	16–23	24–31	32–35	Word
FLINK					0
BLINK					1
RESERVED FOR SOFTWARE					2
00 ↔ 07 STATUS	08 ↔ 15 FLAGS	16 ↔ 23 OPC	24 ↔ 31 MBZ	32 ↔ 35 MBZ	3
00 ↔ 31 XCT_ID				32 ↔ 35 MBZ	4
00 ↔ 31 XCT_ID				32 ↔ 35 MBZ	5
RESERVED FOR SOFTWARE					QUEUE LENGTH − 1

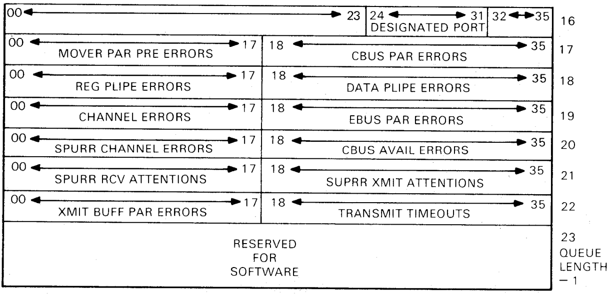
MR-14250

WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 202 octal (RDCNT).

The port will always generate a Counters Read (CNTRD) response to the Read Counters (RDCNT) command.

The format of the Counters Read (CNTRD) response is:

00 ↔ 07	08 ↔ 15	16 ↔ 23	24 ↔ 31	32 ↔ 35	Word
FLINK					0
BLINK					1
RESERVED FOR SOFTWARE					2
STATUS	FLAGS	OPC	MBZ	MBZ	3
XCT_ID (00 ↔ 31)				MBZ	4
XCT_ID (00 ↔ 31)				MBZ	5
MICROCODE VERSION					6
PATH A ACK COUNT					7
PATH A NAK COUNT					8
PATH A NO RESPONSE COUNT					9
PATH B ACK COUNT					10
PATH B NAK COUNT					11
PATH B NO RESPONSE COUNT					12
DATAGRAMS DISCARDED					13
PACKETS TRANSMITTED					14
PACKETS RECEIVED					15



MR-14251

WORD:BITS	NAME	DESCRIPTION
3:15	ERROR	This bit is set if the CNTRD was generated as a result of a Planned CRAM Parity Error (see KLCI Error spec).
3:16-23	OPCODE	OPCODE = 202 octal (CNTRD).

Words 17-22 are called the Port Recoverable Error Counters. The errors have a threshold initially set to 5 by the port during initialization. The threshold can be changed by the port driver with the SETCNT command. The threshold has a value range of 0-17.

DIAGNOSTICS
The following abstracts describe the CI20 diagnostic programs:

- o DFPTA Port Basic Device Diagnostic
- o DFCIA CIA Functional Diagnostic
- o DFSXA KL10 Channel/DTE20-Interaction Test
- o DFCIB CI20 Exerciser
- o DFCIC PDP10 Computer Intercconnect (CI) Responder

DFPTA - PROGRAM ABSTRACT
DFPTA is the basic device diagnostic for the CI20/NI20 controller on the KL10. It is intended to detect and isolate hard or stuck at faults in the device. It is used by Manufacturing to repair CI20/NI20 modules. It is used by Field Service to verify the operation of a port or to isolate a fault to a replaceable module.

DFPTA tests one or two CI20/NI20 controllers. Each controller consists of three port modules residing in RH20 slot #5 or #7.

DFPTA isolates faults to a network of failing chips. Each network is a set of chips, generally functionally related. Typically, several networks are printed, with the first network being the most probable.

DFPTA consists of two major program sections.

- o Test section - Contains all of the hardware tests. This section is used to debug a module or verify a port.

- o Debug section - Allows detailed manipulation or inspection of the port.

DFCIA - PROGRAM ABSTRACT
DFCIA is a functional diagnostic intended to verify the functionality of a CI20 port consisting of a 3 port modules, Packet Buffer module, Link module, CI cables, and Star Coupler. It attempts to isolate faults to one of three items - (1) 3 Port modules, (2) Packet Buffer module, and (3) Link module + cables + Star Coupler.

DFCIA tests the CI port consisting of three port modules residing in RH20 slot #7 and a Packet Buffer module and a Link module residing in a separate card cage.

DFCIA consists of two major program sections.

- o Test section - Contains all of the hardware tests. This section is used to functionally verify the port or to isolate an actual problem.

- o Debug section - Allows detailed manipulation or inspection of the port.

DFSXA - PROGRAM ABSTRACT
"MD-10-DFSXA" is the KL10 Channel/DTE20 interaction test for the KL10 computer system. It provides the service engineer with a software tool that permits interactive testing of all data channels into and out of the KL10's internal memory. The program can test up to eight KL10 I/O Channels (RH20 MASSBUS Controllers or MI20 or CI20 Ports) along with up to four front end PDP-11 Channels (DTE20 KL10 to PDP-11 interfaces).

The RH20 Channels may be tested in one of two modes of operation:

A. In internal loopback mode (deviceless), or

B. By writing/reading data to/from either an RP04 disk pack or a TU16 magtape drive.

The NI20 or CI20 Ports are tested in much the same way as an RH20 Channel in internal loopback mode. Data transfers occur from KL10 memory over the CBUS back to memory over the EBUS interface (called a "write") or in the opposite direction (a "read").

The DTE20 Channels may be tested in one of two possible modes of operation:

A. With only that minimum PDP-11 resident software required to transfer data between "11" and "10" core, or

B. With a preconfigured DEC/X11 systems exerciser load module resident in the PDP-11 front end and exercising the "11" front end devices in parallel with the execution of "DFSXA" on the "10" side.

NOTE
This is only permissible for restricted front ends and does not apply to the master DTE that is running "KLDCP".

The program provides comprehensive error and status reports that permit the service engineer to evaluate system performance and aid in detecting and isolating interactive system problems to the faulty subsystem. This is the only program within the set of DECSYSTEM-20 diagnostics that provides simultaneous testing of both the PDP-11 front-ends and the KL10 I/O channels.

DFCIB - PROGRAM ABSTRACT
The Computer Interconnect Exerciser (DFCIB) is designed to run under the Diagnostic Monitor (KCSUB) in user mode only. It will exercise the entire CI cluster or a desired subset of it. It is designed to guarantee the integrity of the CI and isolate faults to a failing node. The DFCIB will be the driver (controller) in a two process system, the driven process (responder) will be implemented by each node in the CI cluster. The DFCIB resident in the KLIPA (IPA20) will communicate with responders using the Cluster Test Protocol (CTP) and services provided under the Systems Communications Service (SCS). It ensures compliance with the CI Spec. for ID, Message, Datagram, and Data Transfer functions. It will also provide System Performance Data to users.

DFCIC - PROGRAM ABSTRACT
The CI Responder is the slave portion of a master/slave pair of cooperating programs. Its function is to receive and act according to instructions from the master process referred to as a Controller Process. In the CI environment, the Controller Process may be the CI Node Tester (CINT) or a copy of the CI Exerciser Program (CIE) running in some CI Node.

The CI Responder is a user mode only Program that will run under the Tops-20 Operating System (Release 6.0 or newer). The Responder will run during normal timesharing as an Operator Job and does not require exclusive use of any System Resources.

Command and Response Packets are passed between Controller and Responder Processes using the Cluster Test Protocol (CTP). The System Communication Service (SCS) of the Operating System will be used to send CTP messages over the CI.

DESKEWING/ADJUSTMENT PROCEDURE

1. Deskew the port modules by performing the following steps. A Tektronix 475 or equivalent (100 Mhz min) oscilloscope is required.

2. Connect channel 1 of the oscilloscope to MTR MBOX CLK H, 4D33P1, on the CPU backplane. use a ground clip.

3. Set the time base to 20 ns..

4. Set channel 1 vertical gain to 0.5 V/division. SEt the ground reference to 1.3 volts above horizontal center level of oscilloscope. (MTR MBOX CLK H is an ECL signal.)

5. Set the oscilloscope sync to positive external.

6. Connect external sync input to CHT0 H, 4B09K1, on the CPU backplane. Use a ground clip.

7. Connect channel 2 to CDS1, EBUS CLK L, 2A15F1, on the I/O backplane. Set the channel 2 vertical gain to 0.5 V/division. Use ground clip. To measure TTL voltages, set the ground reference to 1.5 volts below horizontal center line of oscilloscope.

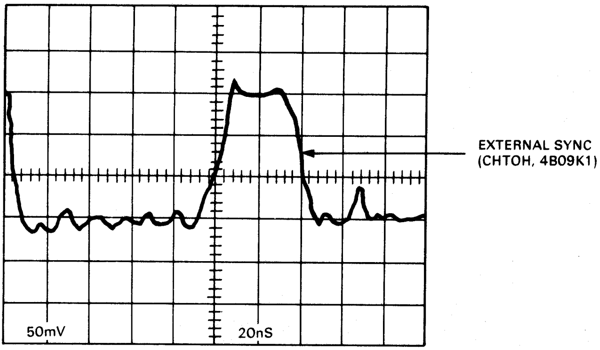
8. Push the Trigger View Switch of the oscilloscope and display the external sync. Adjust the display, so that the rising edge of the external sync aligns with the vertical center line of the oscilloscope.

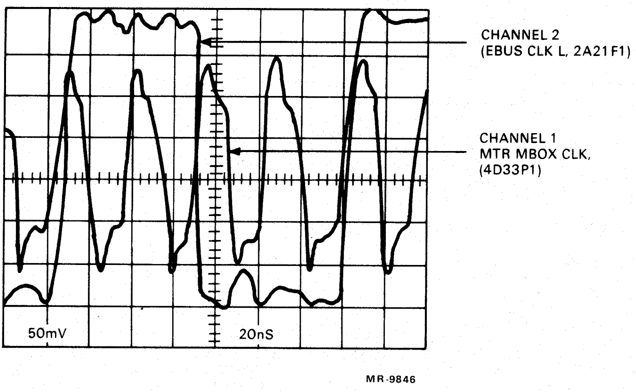
9. Display MBOX CLK H. channel 1. Identify the rising edge of MBOX CLK H that occurs prior to the vertical center line of the oscilloscope. Display channel 1 and channel 2.

10. Put the KL10 in the override fault state. Remove the I/O rear door to access the I/O backplane.

11. In slot 12 of the I/O backplane, locate the bottom potentiometer on the clock distribution module (M8559). Using this potentiometer, adjust the FALLING edge of channel 2,, EBUS CLK L so that it crosses the RISING edge of MBOX CLK H. This crossing occurs on the horizontal center line of the oscilloscope.

12. Disconnect all probes.





Deskew Timing

HARDWARE ERROR CHART

This chart reflects all known CI20 hardware errors. It summarizes the error interface between the Operating System and the Port Hardware. It is recommended that the reader be familiar with the entire contents of this error specification before using this chart.

Event	Monitor Sees	BUGINF	BUGINF Data	Reload Microcode	CI20
single nonplanned CRAM PARITY error 0-7747	CSR bit 6	KLPUCP	CSR, UCODE Adr, UCODE Data	Yes	Yes
EBUS PARITY error	CSR bit 6,24 LAR 7752	KLPEBP	CSR, UCODE ADR, PCB ERROR WORD 0 PAGE FAIL WORD	No	Yes
EBUS PARITY error	CSR bit 6,24 LAR 7753	KLPEBQ	CSR, UCODE ADR, PCB ERROR WORD 0 PCB ERROR WORD 1	No	Yes
PLI Parity error	CSR bit 6 LAR 7754	KLPPPE	" "	No	Yes
DATA PATH error	CSR bit 6,26 LAR 7756	KLPDPP	" "	No	Yes
CBUS Parity error	CSR bit 6 LAR 7755	KLPCBS	" "	No	Yes
CHANNEL ERROR	CSR bit 6 LAR 7763	KLPSCE	" "	No	Yes
SHORT WORD COUNT	CSR bit 6 LAR 7762	KLPSWC	" "	No	Yes
ADDR. PAR ERROR	APR INT. PCB CHAN WORDS 1+2	**STANDARD**	**STANDARD**	No	No
NXM	APR INT. PCB CHAN WORDS 1+2	**STANDARD**	**STANDARD**	No	No
MEM PAR ERROR	APR INT. PCB CHAN WORDS 1+2	**STANDARD**	**STANDARD**	No	No
OVERUN	PCB CHAN WORDS 1+2	**STANDARD**	**STANDARD**	No	No
CBUS AVAIL error	CSR bit 6 LAR 7757	KLPCBN	CSR, UCODE ADR PCB CHAN WORDS 1 + 2	No	Yes
EBUS REQUEST err	CSR bit 6 LAR 7760	KLPERE	CSR, UCODE ADR,	No	Yes
MBUS error	CSR bit 7	KLPMBS	CSR, UCODE ADR, UCODE DATA (TWO WORDS)	No	Yes
GRANT CSR	CSR bit 6 LAR 7761	KLPCSR	" "	No	Yes
TTO	STATUS FIELD	KLPTMO	CSR, UCODE VER, TRANSMIT STATUS REG	No	No
TRANS. BUF. PARITY ERROR	STATUS FIELD	KLPTPE	CSR, UCODE VER, TRANSMIT STATUS REG	No	No
INTERNAL PORT ERR	CSR bit 6 LAR 7750	KLPINP	CSR, UCODE VER, UCODE ADR	No	Yes
FAILED SELF TEST	CSR bit 6 LAR 7751	KLPFST	" "	No	Yes
SPURIOUS TRANS ATTENTION ERR	CSR bit 6 LAR 7764	KLPTAE	CSR, UCODE VER, UCODE ADR, TRANSMIT STATUS REGISTER	No	Yes

HARDWARE ERROR CHART (Cont)

Event	Monitor Sees	BUGINF	BUGINF Data	Reload Microcode	Start CI20
SPURIOUS REC ATTENTION ERR	CSR bit 6 LAR 7765	KLPRAE	CSR, UCODE VER, UCODE ADR, RECEIVE STATUS REGISTER	No	Yes
SPURIOUS CHAN ATTENTION ERR	CSR bit 6 LAR 7763	KLPSCE	CSR, UCODE VER, UCODE ADR, CHAN LOGO WORD 1	No	Yes

STANDARD implies that this error reporting has already been implemented in previous releases of TOPS-20 and has not changed.

Excluding CI20 at start up, perform a switch register load and display the KLI> prompt. Respond to the questions displayed as follows.

```
RSX-20F vb15-20 16:10 24-Aug-84

[SY0: redirected to DB0:]
[DB0: mounted]
KLI -- VERSION VB15-12 RUNNING
KLI -- ENTER DIALOG [NO,YES,EXIT,BOOT]?
KLI>YES
KLI -- KL10 S/N:3543., MODEL B, 60 HERTZ
KLI -- KL10 HARDWARE ENVIRONMENT:
        MCA25 CACHE PAGER
        MOS MASTER OSCILLATOR
        EXTENDED ADDRESSING
        INTERNAL CHANNELS
        CABLE

KLI -- SELECT PAGE TABLE [FILE,BOTH,0,1]?
KLI>BOTH
KLI -- PAGE TABLE SELECTED: BOTH
KLI -- RELOAD MICROCODE [YES,VERIFY,FIX,NO]?
KLI>YES
KLI -- MICROCODE VERSION 352 LOADED
KLI -- RECONFIGURE CACHE [FILE,ALL,YES,NO]?
KLI>ALL
KLI -- ALL CACHES ENABLED
KLI -- CONFIGURE KL MEMORY [FILE,ALL,REVERSE,FORCE,YES,NO]?
KLI>ALL

LOGICAL MEMORY CONFIGURATION
  ADDRESS   SIZE  INT  TYPE CONTROLLER
 00000000 1024K   4    MG20  10
 04000000 1024K   4    MG20  11

KLI -- LOAD KL BOOTSTRAP [FILE,YES,NO,FILENAME]?
KLI>YES
KLI -- WRITE CONFIGRATION FILE [YES,NO]?
KLI>NO
KLI -- BOOTSTRAP LOADED AND STARTED

BOOT V10.0(201)

BOOT>
```

ENTER THE PUBLIC STRUCTURE AND START EDDT AS FOLLOWS:

```
BOOT> PS :/E
[BOOT: [LOADING] [OK]
EDDT
```

ENTER RET COMMAND AT LOCATION KLPINI+1 FOLLOWED BY STARTING LOCATION OF THE MONITOR.

```
KLPINI+1/    CONI 574,T1    RET


147$G
```

CI20 Labels

LABEL FOR
CI CARD
CAGE MUL

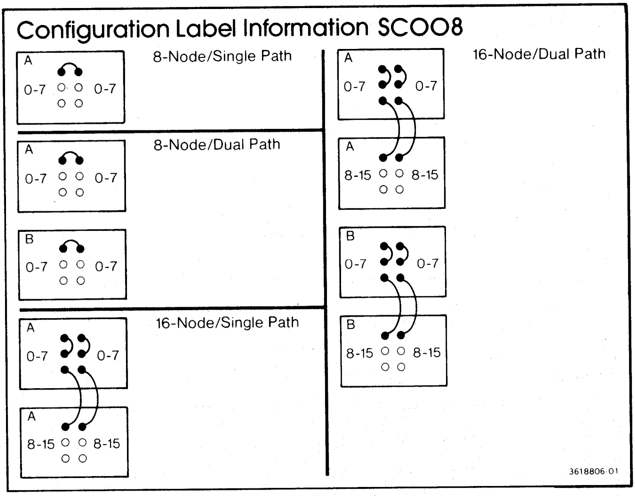
MODULE	LOCATION
L0109	LEFT
L0100	RIGHT

MR-14245

AIR FLOW FAULT LABEL IS PLACED OVER THE EXISTING CPU AIR FAULT MESSAGE DECAL ON THE 863 FAULT SWITCH.

AIR FLOW CPU/CI/NI

MR-14246



TK-9219

Star Coupler SC008 Configuration Path Label

The procedures for READ COUNTERS and SETTING THRESHOLDS will be supplied at a later date.

NETWORK INTERCONNECT ADAPTER

Table of Contents

Page

DOCUMENTATION
Refer to the NIA20 Reference Manual (Document No. EK-NIA20-RM) for detailed functional/logic/installation descriptions. The Print Sets can be ordered as follows:

Print Set	Order No.
NIA20-A	MP-01984
NIA20-B	MP-01907
NIA20-C	MP-01908

NIA20 CARD CAGE

REAR PANEL CONNECTORS

CARD CAGE:

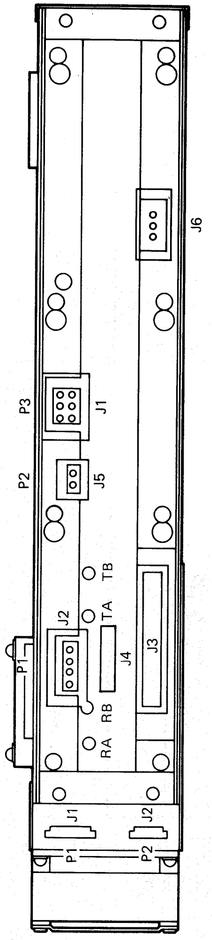
J1	DC POWER +5.0 VOLTS, GND
J2	DC POWER −5.2 VOLTS, GND
J3	PLI
J4	NIA20 INTERNAL TRANSCEIVER CABLE
J5	VOLTAGE MONITOR FOR +5.0 VOLTS
J6	VANE SWITCH
TB	FOR
TA	CI20
RB	USE
RA	ONLY

FAN SUBASSEMBLY :

J1	VAN SWITCH
J2	FAN AC

REAR VIEW

TOP

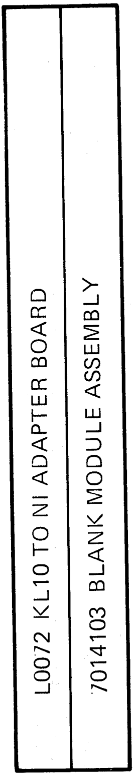


BOTTOM

MR-14241

INTERNAL SLOT ASSIGNMENTS

TOP



NIA20 CARD CAGE
(OPENED FRONT DOOR VIEW)

MUL DECAL
(LOCATED ON SIDE DOOR)

MODULE	LOCATION
7014103	RIGHT
L0072	LEFT

MR-14263

MODULE LOCATIONS
RH/DTE/CI/NIA

The following NIA modules are located as indicated:

Slot	Module
19	M3001 EBUS INTERFACE/PORT ALU
20	M3002 PORT MICROPROCESSOR CONTROL
21	M3003 CBUS/PLI INTERFACE
22}	
23}	BLANK MODULE ASSEMBLY
24}	

Refer to Volume I DIAGRAMS MULS for all other slot assignments.

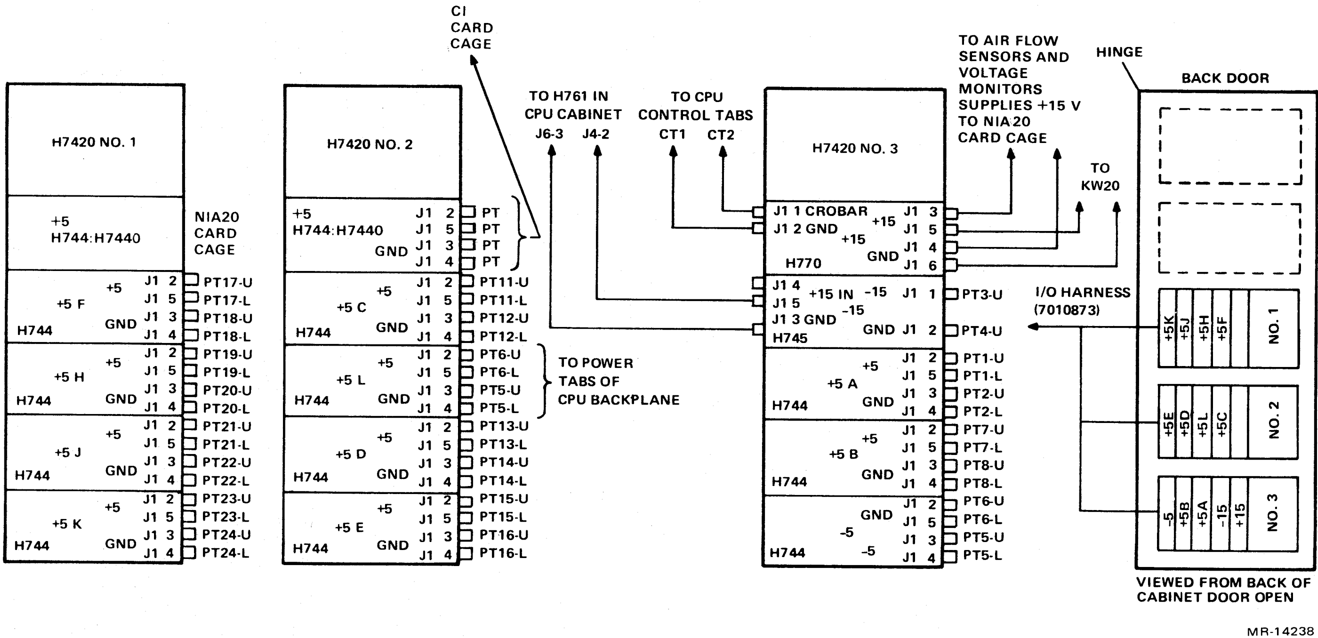
MBOX/EBOX

Slot	Module
31	M8532-YA PI BOARD PIC

GENERAL POWER SUPPLY SPECIFICATIONS

The following voltage measurements are to be made from the backplane of the option.

POWER SUPPLY TYPE	OUTPUT	TOLERANCE MAXIMUM	TOLERANCE MINIMUM	MAXIMUM RIPPLE IN MILLIVOLTS
702		VARIABLE		600
703	+10	+11.0	+9.4	300
	+1.8	+1.9	+1.7	N/A
	-3.0	-3.15	-2.85	N/A
705	+10	+11.0	+9.4	300
	-15	-16.5	-14.5	700
706	+50	+54.0	+49.0	1500
723	+8	+9.5	+7.8	600
725	-15	-18.0	-14.7	900
728	+10	+11.0	+9.5	700
	-15	-16.0	-14.5	700
732		VARIABLE		600
739	+53	+55	+52	250
	+65	+65	+63	250
742/7420	+25 V	+30	+20	N/A
	-15	-16.5	-13.5	N/A
	+3	+3.5	+2.5	N/A
744/7440	+5	+5.05	+4.95	150
745	-15	-15.05	-14.95	450
754	+20	+20.2	+19.8	450
	-5	-5.05	-4.95	150
761	-2	NONE		
	-5.2	NONE		
770	+15	+15.05	+14.95	450
778	-15	-16.5	-14.5	700
7131/7131A	+5	+5.07	+4.93	50
	+12	+12.18	+11.82	100
	-2	-2.03	-1.97	50
	-5.2	-5.28	-5.12	50



KL10 I/O DC POWER (DISTRIBUTION)

SWITCH SETTINGS
DC Voltage Monitor Board (DEC P/N 5414506-01)

Switch 1 should be set to ON. All other switches should be OFF.

CONTROL AND STATUS REGISTER (CSR)

BIT NO.	BIT DEFINITION	RD/WR	
		KL10	PORT
00	PORT PRESENT	R	H
01	DIAG RQST CSR	R	H
02	DIAG CSR CHNG	R/H	H
03		*	*
04	RQST EXAM OR DEP	R/H	R/S
05	RQST INTERRUPT	R/H	R/S
06	CARD PARITY ERR	R/C	H
07	MBUS ERROR	R	H
08		*	*
09		*	*
10		*	*
11	IDLE	R	R/W
12	DISABLE COMPLETE	R	R/W
13	ENABLE COMPLETE	R	R/W
14		*	*
15	PORT ID CODE 00	R	H
16	PORT ID CODE 01	R	H
17	PORT ID CODE 02	R	H
18	CLEAR PORT	W	*
19	DIAG TEST EBUF	R/W	*
20	DIAG GEN EBUS PE	R/W	*
21	DIAG SEL LAR	R/W	*
22	DIAG SINGLE CYC	R/W	*
23	SPARE	R/W	*
24	EBUS PARITY ERR	H/R/C	R/H
25	FREE QUEUE ERR	R/C	R/S
26	DATA PATH ERR	R/C	R/S
27	CMD QUEUE AVAIL	R/S	R/C
28	RSP QUEUE AVAIL	R/C	R/S
29		*	*
30	DISABLE	R/S	R/C
31	ENABLE	R/S	R/C
32	MPROC RUN	R/W	R/H
33	PIA 00	R/W	R
34	PIA 01	R/W	R
35	PIA 02	R/W	R

* = NOT DEFINED
R = READABLE
W = WRITABLE (SET OR CLEAR)
C = CLEARABLE ONLY
S = SETTABLE ONLY
H = HARDWARE CONTROLLED

MR-14248

CSR BIT DEFINITIONS

BIT	NAME	DEFINITION
00	PORT PRESENT	Indicates to the KL10 that the port is present (installed and powered-up).
01	DIAG RQST CSR	When set, this diagnostic bit indicates that the port has requested access to the CSR.
02	DIAG CSR CHNG	This diagnostic bit indicates that the contents of the CSR have changed since it was last read by the port micrporocessor.
03	UNUSED	Not used by either the port microprocessor or the KL10.
04	RQST EXAM OR DEP	Used by the port microprocessor to request an EBus interrupt on PI level 00 (Examine or Deposit function). The setting of this bit immediately generates the interrupt request.
05	RQST INTERRUPT	Used by the port microprocessor to request an EBus interrupt on PI levels 01 through 07. The setting of this bit immediately generates the interrupt request.
06	CRAM PAR ERR	Indicates that a control RAM (CRAM) parity error has been detected. If this bit is set, the port microprocessor will be immediately halted and RQST INTERRUPT (CSR bit 05) will be set. A hardware nonvectored (40 + 2n) interrupt will be forced. A CRAM PAR ERR may be forced in order to halt the port microprocessor at a specific location (break point). The port microprocessor cannot be restarted (CSR bit 32 set) until this bit is cleared.
07	MBUS ERR	Indicates that more than one MBUS driver has been turned on at the same time. That is, more than one set of port logic is trying to drive the MBUS at the same time. If this bit is set, the port microprocessor will be immediately halted and RQST INTERRUPT (CSR bit 05) will be set. A hardware nonvectored (40 + 2n) interrupt will be forced. The port microprocessor cannot be restarted (CSR bit 32 set) until this bit is cleared.

CSR BIT DEFINITIONS (Cont)

BIT	NAME	DEFINITION
08	UNUSED	Neither the port microprocessor or the KL10 uses this bit.
09	UNUSED	Neither the port microprocessor or the KL10 uses this bit.
10	UNUSED	Neither the port microprocessor or the KL10 uses this bit.
11	IDLE LOOP	Indicates that the port microprocessor is in the Idle Loop, and is not "hung" in some other microcode routine.
12	DISABLE COMPLETE	Informs the KL10 that the port microprocessor has placed itself in the DISABLED state.
13	ENABLE COMPLETE	Informs the KL10 that the port microprocessor has placed itself in the ENABLED state.
14	UNUSED	Neither the port microprocessor or the KL10 uses this bit.
15 16 17	PORT ID CODE 00 PORT ID CODE 01 PORT ID CODE 02 Hardwired so that:	Three-bit PORT IDENT CODE field. Informs software that this is a CI20 port and not an RH20 controller. 00 = 0 01 = 1 02 = 1
18	CLEAR PORT	When set by the KL10, this bit resets the port. The microprocessor is halted and all pertinent registers and control logic are placed in a reset state. The bit clears itself after the reset function is completed.
19	DIAG TEST EBUF	This diagnostic bit enables the KL10 to do an EBus interface loopback function by loading and reading the EBus buffer (EBUF). If the port is not running (CSR bit 32 is reset) and this bit is set, then a KL10: DATAO loads EBus data into the EBUF. DATAI places EBUF data on the EBus.
20	DIAG GEN EBUS PE	This diagnostic bit enables the KL10 to test the EBus parity checker by forcing it to decode an EBus parity error. When this bit is set, EBUS PAR ERR (CSR bit 24) will also be set on the same CONO, assuming there was no real EBus parity error.
21	DIAG SEL LAR	This diagnostic bit enables a KL10 DATAI to read the CRAM address, contained in the Latch Address Register (LAR). If this bit is set and bits 19 and 32 are reset, then the DATAI will cause the LAR contents to be asserted on EBus D01-D12.

CSR BIT DEFINITIONS (Cont)

BIT	NAME	DEFINITION
22	DIAG SINGLE CYC	This diagnostic bit enables the port microprocessor to be single cycled. If this bit is set and the KL10 sets MPROC RUN (CSR bit 32), the port microprocessor will execute one microcycle and halt. MPROC RUN will be cleared when the microprocessor halts. The current address to be executed is fetched from the RAM Address Register (RAR). The next address to be executed is stored in the LAR at the completion of the microcycle. The KL10 must read the address from the Latch Address Register (LAR) and load it into the RAR before executing the next single cycle.
23	SPARE	Reserved for future software use.
24	EBUS PARITY ERR	When read by the KL10, this bit indicated that an EBus parity error has been detected. When written as a "1" by the KL10, this bit will clear itself and CRAM PARITY ERR (CSR bit 06).
25	FREE QUEUE ERR	Used by the port to inform the Port Driver that there are no free queue entries available on either the Datagram Free Queue or the Message Free Queue.
26	DATA PATH ERR	Informs the Port Driver that the port microprocessor has detected an error in the DMA data path.
27	CMD QUEUE AVAIL	Used by the Port Driver to inform the port that it has placed a command queue entry on a previously empty command queue.
28	RESP QUEUE AVAIL	Used by the port to inform the Port Driver that it has placed an entry on the previously empty Response Queue.
29	UNUSED	Neither the port microprocessor or the KL10 uses this bit.
30	DISABLE	Used by the Port Driver to tell the port to place itself in the DISABLED state (set CSR bit 12).
31	ENABLE	Used by the Port Driver to tell the port to place itself in the ENABLED state (set CSR bit 13).
32	MPROC RUN	When set by the KL10, this bit causes the CRAM Control Register to reset and enables the port microprocessor clocks. The port will start cycling at the address contained in the RAM Address Register (RAR). The next and subsequent addresses will be fetched from the Am2910 sequencer.
33 34 35	PIA00 PIA01 PIA02	Three-bit KL10 EBus Physical Interrupt Assignment (PIA) field (PI level 01 through 07).

KL10 MEMORY PORT CONTROL BLOCK (PCB)

PCB FORMAT	OCT
COMMAND QUEUE INTERLOCK	0
COMMAND QUEUE FLINK	1
COMMAND QUEUE BLINK	2
RESERVED FOR SOFTWARE	3
RESPONSE QUEUE INTERLOCK	4
RESPONSE QUEUE FLINK	5
RESPONSE QUEUE BLINK	6
RESERVED FOR SOFTWARE	7
UNKNOWN PROTOCOL TYPE FREE QUEUE INTERLOCK	10
UNKNOWN PROTOCOL TYPE FREE QUEUE FLINK	11
UNKNOWN PROTOCOL TYPE FREE QUEUE BLINK	12
UNKNOWN PROTOCOL QUEUE ENTRY LENGTH	13
RESERVED FOR SOFTWARE	14
PROTOCOL TYPE TABLE STARTING ADDRESS	15
MULTI-CAST ADDRESS TABLE STARTING ADDRESS	16
RESERVED FOR SOFTWARE	17
ERROR LOGOUT 0	20
ERROR LOGOUT 1	21
EPT CHANNEL LOGOUT WORD 1 ADDRESS	22
EPT CHANNEL LOGOUT WORD 1 CONTENTS	23
PCB BASE ADDRESS	24
PIA ASSIGNMENT	25
RESERVED TO PORT	26
CHANNEL COMMAND WORD	27
READ COUNTERS DATA BUFFER STARTING ADDRESS	30

MR-14261

PORT CONTROL BLOCK
The Port Control Block is used to anchor the queues at a known point in the host memory and to provide certain initial parameters to the port. The queues are used to pass commands from the port-driver software to the port for either local execution or for transmission over the NI wire. The queues are also used by the port to pass responses back to the port driver software and to deposit packets received over the NI wire.

The Port Control Block (PCB) is a data structure based in the host memory that allows the sharing of the queues by the port driver and the port. The Port Control Block is pointed to by port register 2, the PCB Base register.

The port is informed of the location of the PCB at microcode initialization time by the port driver software. When this is detected by the port, it will cache the following variables from the PCB: the unknown protocol type queue entry length, the protocol type table starting address, and the multi-cast address table starting address.

Both the host port driver and port read and write locations in the PCB. There is exactly one PCB for each NI port controlled by the host system. The PCB is the main control structure for the NI port. It anchors all of the tables and queue structures. The PCB contains queue headers to anchor the command queue, the response queue, and the unknown protocol type free queue. Base pointers to the Multi-cast address table, and the protocol type table are located in the PCB. In addition, an error logout area, and several free words provided for the use of the driver program are included in the PCB. The reserved words will never be altered or examined by the NI port. The error logout area may be written at any time by the port to record an error event.

Queue Headers - A queue consists of a queue header which anchors the queue and a number of entries, each occupying a spot on the queue. All LCG NI queues are doubly linked structures. The queue header and each queue entry contain a forward link (FLINK) and a backwards link (BLINK). The forward link of a queue header points at the first entry of the queue. The forward link of a queue entry points to the next entry of the queue, if any. The backwards link of a queue header points at the last entry of the queue. The backwards link of a queue entry points back at the entry before the entry on the queue, if any. If an flink does not point to a queue entry, it points at the queue header flink. If a blink does not point to a queue entry, it points at the queue header flink.

Queue headers anchor a queue structure. A queue header may be located in the PCB, or located in host memory as a free standing structure. A queue header is composed of a reserved word, a FLINK, a BLINK, and a Queue Length. The FLINK (forward link) points to the first word, the FLINK, of the first entry of the The BLINK points to the FLINK word of the last entry of the queue. The first and last entries may be the same entry. If there are no entries on the queue, the queue header FLINK points to itself.

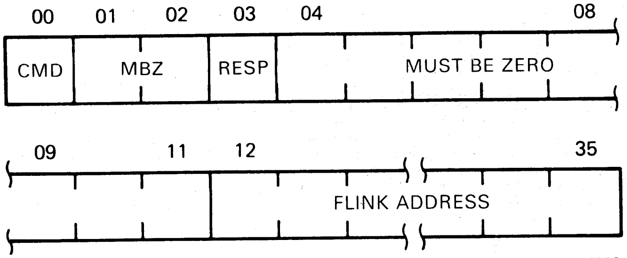
Queue Interlocks - The NI20 requires special KL10 microcode support to allow the NI20 to perform a memory increment operation using read-pause-write memory references. This is needed to allow the port to interlock the queues.

There is a separate interlock word for each queue. When a queue is available, the corresponding interlock word has a value of -1. When either the operating system or the port want to interlock the queue, they must perform a noninterruptable increment-store-test operation, such as an AOSE. If the incremented location has a value of zero, then the queue has been successfully interlocked and the process may now manipulate the queues. If the incremented value is greater than zero, then the queue is not available. The interlock word should not be set back to zero. When the process is finished with the queues, the interlock word must be set back to -1 (all ones). This marks the queue as available. Both the port driver and the port microcode are responsible for leaving the queues in a well defined state. The PCB must be allocated starting on a four word boundary.

The Error Words (Words 20, 21) are written by the port when it encounters fatal errors associated with Queue manipulation. This error reporting strategy requires the port to write as much information as possible directly into the host memory. This approach requires the smallest subset of port hardware and microcode to be working to report these errors.

The information in these words provides sufficient data for the port driver to determine the type of error and where the error occurred. When the error is detected, the port will write the contents of the Error Words in the PCB, enter the Disabled State, and generate a host interrupt.

The format of Error Word 0 is:

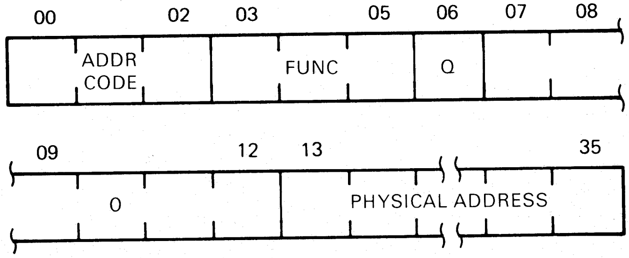


MR-14256

BITS	NAME	DESCRIPTION
0	CMD	Error occurred while reading a command queue entry.
1-2	MBZ	These bits will be zero.
3	RESPONSE	This bit is on if the error occurred while the port was attempting to build a response queue entry.
4-11	MBZ	These bits will be zero.
12-35	FLINK ADR	This is the address of the FLINK word of the queue in question.

Error Word 1 contains the API function word that the port processor used to access memory when the memory error occurred. This word is written here in the same format as it should have appeared on the EBUS.

The format of this word is:



MR-14257

Word 22 of the PCB is written by the port during initialization time with the address of the EPT Channel Logout Word 1 which the port gets from the port driver software. Words 22 and 23 of the PCB are used by the port during Channel Error recovery.

Word 23 contains the Channel Logout Word 1 written by port on any kind of channel error detected during or immediately after, a DMA transfer.

The format of Error Word 3 is:

00	01	02	03	04	05	08
1	MEM PE	−ADR PE	−WC =0	NXM	0	

09	10	11	12	13	14	35
LXE	0	LONG WC	SHORT WC	OVER RUN	ADDR OF CURR CCW+1	

MR-14255

BITS	NAME	DESCRIPTION
01	MEM PE	Memory Parity Error.
02	-ADR PE	Not Address Parity Error.
03	-WC=0	Chan Word Count did not = 0 when chan did a store to EPT.
04	NXM	Chan ref non exist mem.
09	LXE	Error detected after port term transfer. Chan aborts next transfer.
11	LONG WC	Port comp Xfer, but word count in CCW not reached.
12	SHORT WC	Chan Xferred data spec by CCW, but port still has data.
13	OVER RUN	If device read, Port sent data but chan buff was full. If device write, Port req data but chan buff was empty.

Word 24 of the PCB is the address of the first word of the PCB; the NI20 has no other way of finding the PCB.

Word 27 is reserved for the Channel Command Word. The port will write a CCW-style word here when it wishes to tranfer data over the KL10 CBus. The port driver is responsible for writing a Channel Jump Word into the appropriate EPT location corresponding to the RH20 backplane slot that the NI20 is installed in.

Word 25 is always reserved to the port microcode for its use; the port driver should never write this location nor depend upon its value.

Word 30 of the PCB is a pointer to the beginning of the Read Counter Data Buffer. This address is supplied by the port driver software at initialization.

When the NI20 is being initialized, the port driver must set up the channel to transfer the contents of the PCB into the port. This is done by setting up a CCW to transfer 3 words starting with word 24 of the PCB from KL10 memory to the channel. The port will start the channel and will read the contents of these locations. This provides the port with the base of the PCB, and its PI assignment.

It is important to realize that since the port will be using the channel to transfer large blocks of data, the channel will be writing logout information into the EPT. An error that the channel discovers will be reported in the usual manner via the EPT.

DIAGNOSTICS
The following abstracts describe the NIA20 diagnostic programs:

- DFPTA Port Basic Device Diagnostic
- DFNIE NIA Module (L0072) Diagnostic
- DFNIA NIA20 Functional Diagnostic
- DFNIB Network Interconnect Exerciser

DFPTA - PROGRAM ABSTRACT
DFPTA is the basic device diagnostic for the CI20/NI20 controller on the KL10. It is intended to detect and isolate hard or stuck at faults in the device. It is used by Manufacturing to repair CI20/NI20 modules. It is used by Field Service to verify the operation of a port or to isolate a fault to a replaceable module.

DFPTA tests one or two CI20/NI20 controllers. Each controller consists of three port modules residing in RH20 slot #5 or #7.

DFPTA isolates faults to a network of failing chips. Each network is a set of chips, generally functionally related. Typically, several networks are printed, with the first network being the most probable.

DFPTA consists of two major program sections.

- Test section - Contains all of the hardware tests. This section is used to debug a module or verify a port.

- Debug section - Allows detailed manipulation or inspection of the port.

DFNIE - PROGRAM ABSTRACT
DFNIE is the hardware diagnostic for the Network Interface Adapter (NIA) on the KL10. It is intended to detect and isolate hard or stuck at faults in the device. It is used by Manufacturing to repair NIA modules. It is used by Field Service to verify the operation of an NIA and to isolate a fault to the replaceable module (NIA).

DFNIE tests one NIA controller. Each controller consists of an NI20 port in RH20 slot 5.

DFNIE isolates faults to a network of failing chips. Each network is a set of chips, generally functionally related. Typically, several networks are printed, with the first network being the most probable.

DFNIE consists of two major program sections.

- Test section - Contains all of the hardware tests. This section is used to debug a module or verify proper hardware operation of the NIA.
- Debug section - Allows detailed manipulation or inspection of the PLI.

DFNIA - PROGRAM ABSTRACT

DFNIA is a functional diagnostic intended to verify the functionality of an NI20 port consisting of 3 Port Modules, an NIA module, H4000 Transceiver, and an NI cable. It attempts to isolate faults to either of - (1) 3 Port Modules, and (2) NIA module + H4000 Transceiver + cables.

DFNIA tests the NI port consisting of three port modules residing in RH20 slot #5 and an NIA module residing in a separate card cage.

DFNIA consists of two major program sections.

- o Test section - Contains all of the hardware tests. This section is used to functionally verify the port or to isolate an actual problem.

- o Debug section - Allows detailed manipulation or inspection of the port.

DFNIB - PROGRAM ABSTRACT

The Network Interconnect Exerciser exercises from a KL10 system the ability of all nodes on an NI network to communicate with each other. Testing is done using the Low Level Maintenance Operations (LLMOP) of the NI. DFNIB does not interfere with normal NI network traffic and runs concurrently with normal NI-DECNET network traffic. DFNIB is a USER mode-only diagnostic and requires TOPS-20 Version 6.0 or later with NI-DECNET support. DFNIB is a self-contained program and does not require any diagnostic support programs.

USERS AND USES

DFNIB is a part of the standard KL10 Diagnostic package and is available to those users who are eligible to receive KL10 diagnostics and related updates.

The program is designed for the use by engineers and technicians who are qualified to test and maintain NI networks. In order to run the diagnostic, the user will be required to have maintenance, wheel or operator privileges.

Some typical DFNIB uses:

- o Installation testing of a new node.
- o Installation acceptance of a new node.
- o Isolation of faulty nodes.
- o Verification of a repaired node.
- o Exercising an NI network.

PRE-REQUISITE SOFTWARE

DFNIB requires TOPS-20 with NI-DECNET running. TOPS-20 assumes that no solid faults exist in the KL10 cpu/memory/node Hardware. The following diagnostics should be run prior to running TOPS-20.

- o CPU and MEMORY diagnostics (all)
- o NI NODE diagnostics (all)

DESKEWING/ADJUSTMENT PROCEDURE

1. Deskew the port modules using a Tektronix 475 (or equivalent 100 MHZ minimum) oscilloscope by performing the following steps (see Figure).

2. Connect channel 1 of the oscilloscope to MTR MBOX CLK H, 4D33P1, on the CPU backplane. Use a ground clip.

3. Set the time base to 20 ns.

4. Set channel 1 vertical gain to 0.5 V/division. Set the ground reference to 1.3 volts above the horizontal center level of the oscilloscope. (MTR MBOX CLK H is an ECL signal.)

5. Set the oscilloscope sync to positive external.

6. Connect external sync input to CHTO H, 4H09K1 on the CPU backplane. Use a ground clip.

7. Connect channel 2 to CDS1, EBUS CLK L, 2A21F1 on the I/O backplane. Set the channel 2 vertical gain to 0.5 V/division. Use a ground clip. To measure TTL voltages, set the ground reference to 1.5 volts below the horizontal center line of the oscilloscope.

8. Press the Trigger View Switch of the oscilloscope and display the external sync. Adjust the display, so that the rising edge of the external sync aligns with the vertical center line of the oscilloscope.

9. Display MBOX CLK H, channel 1. Identify the rising edge of MBOX CLK H that occurs prior to the vertical center line of the oscilloscope. Display channel 1 and channel 2.

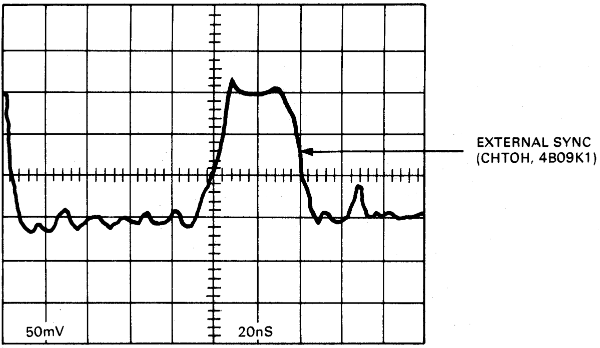
10. Put the KL10-E in the override fault state. Remove the I/O rear door to access the I/O backplane.

11. Locate the bottom potentiometer on the clock module (M8559) in slot 12 of the I/O backplane. Using this potentiometer, adjust the FALLING edge of channel 2, EBUS CLK L so that it crosses the RISING edge of MBOX CLK H. This crossing occurs on the horizontal center line of the oscilloscope.

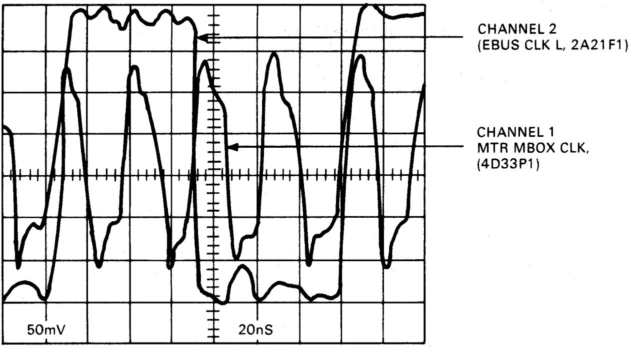
12. Disconnect all probes.

13. Mount the KLAD back on the front end RP06.

14. Load and run diagnostic DFPTA to verify proper functioning of the port modules. If the modules fail, troubleshoot as directed by the diagnostic. If the modules are functioning properly, continue with the installation.



EXTERNAL SYNC (CHTO H)



EBUS CKL L AND MTR MBOX CLK

MR-13732

NIA20 Deskewing Timing

NIA20 LABELS

CARD CAGE MUL LABEL

MODULE	LOCATION
7014103	RIGHT
L0072	LEFT

MR-14243

AIR FLOW FAULT LABEL

(PLACED OVER THE EXISTING CPU AIR FAULT MESSAGE DECAL ON THE 863 FAULT SWITCH)

AIR FLOW CPU/CI/NIA

MR-14244

Table of Contents

RA81 ERROR CODES

SYSTEM FAULT CODES
The RA81 drive errors and fault display codes are listed below.

NOTE
Loop-back plugs left in the drive following execution of Diagnostic Test C cause multiple false error codes to be displayed.

Drive Error Display Codes

Error Code	Name
01	Spindle motor speed transcuder timeout
02	Spindle speed of 100 r/min not reached in 6 seconds during spin-up
03	Spindle not accelerating during spinup
04	Spindle speed of 3420 r/min not reached in 40 seconds during spin-up
05	Start-up being inhibited by power sequencing grant not present or start in progress (SIP) present (this could be legitimate status)
06	Microcode fault
07	Level two message frame sequencing error
08	Level two message checksum error
09	SDI message framing error
0A	Invalid operation code parity for level 1 or 2 message
0B	Invalid operation code for a level 1 or 2 message
0C	Invalid operation length for a level 2 message

Drive Error Display Codes (Cont)

Error Code	Name
0D	Status error byte nonzero while attempting to execute a command
0E	Group select code nonzero while attempting to execute a command
0F	Write protect switch is in the PROTECT position while attempting to write enable the drive
10	Invalid error code sent by slave
13	Fine track not reached during detent mode
15	Too much time taken to execute a seek or recal command
16	Guard band detected while performing a seek command
17	Seek logic difference counter decremented past 0 before expected cylinder was reached
1A	Seek command contains an invalid cylinder address
1B	Velocity calibration detected too many bad seeks
1C	Unsuccessful recalibrate command detected during a velocity calibration
1D	Drive seeks are more than 10% too fast
1E	Drive seeks are more than 15% too slow
1F	A sector pulse is detected during the execution of a read or write of a sector
20	A parity is error detected on the controller real-time state line
21	Two or more pulses of the same polarity are detected on the controller real-time state line (control pulse error)
22	Two or more pulses of the same polarity are detected on the controller write command data line (data pulse error)
23	Spindle motor interlock broken (belt tension lever is released)
25	Servo error (SVOCHK) detected (off track during detent mode)
26	Spindle speed is detected as being less than 3420 r/min (3600-5%)
27	HDA temperature too high

Drive Error Display Codes (Cont)

Error Code	Name
28	Servo module temperature too high
29	Invalid error recovery level specified
2A	Invalid subunit specified
2B	Invalid region specified in a diagnose command
2C	Seek or recal command attempted thile the spindle is not spinning
2D	Invalid command timeout value given
2E	Controller flags are detected prohibiting drive spinup
2F	RUN/STOP switch is in the stop position while attempting a run command
30	Write current is turned on without write gate being asserted
31	A read command is attempted with write gate asserted
32	A read or write command is attempted while the drive is faulted
33	A burst error is detected while writing
34	Read data separator/encoder error
35	Write unsafe error detected while attempting a write command
36	Short circuit detected in head winding
37	No write current detected with write gate asserted
38	A read command is attempted with multiple heads selected
39	A write command is attempted while the positioner is off track (not detented)
3A	A write command is attempted while the drive is write protected
3B	Servo/HDA interlock broken
3C	Servo interlock broken
3D	Read/Write interlock broken

Drive Error Display Codes (Cont)

Error Code	Name
3E	Control panel interlock broken
3F	Personality module interlock broken
40	Invalid R/W region specified
41	SDI controller response timed out
42	Drive not in an on-line state while attempting a seek command
43	R/W READY not set while attempting a real-time command
44	Format enable not set while attempting a format command
45	Real-time command contains an invalid head (track) address
46	R/W safety interrupt occurred with no cause bits set
47	Disconnect command contains an incorrect TT bit
48	Invalid write memory offset or byte count
49	Invalid command found while in topology mode
4A	Drive is disabled by DD bit
4B	Index pulse error
4D	Bad embedded servo data found during a write command
50	Slave diagnostic timeout NOTE This error could be the result of the positioner lock lever in the lock position.
51	The sector/byte failed to count properly
52	Group 0 cannot be selected
53	Group 1 cannot be selected

Drive Error Display Codes (Cont)

Error Code	Name
54	R/W head select error while executing the head select multiplexer test
57	Master RAM timer faulty
60	R/W head select error while executing read/write test
61	R/W data setup error (diagnostic write data register not equal to diagnostic read data reg.)
62	The data from 3 or more heads is bad while executing the read-only test
63	The data from 1 or 2 heads is bad while executing the read-only test
65	Read/Write test sector could not be found within two revolutions of the disk
66	Read failure caused by servo being off-track
67	Write test cannot be executed because Test F was not run or failed
68	Read-only cylinder cannot be reformatted without jumper
69	Read/Write diagnostic comparison circuitry never detects an error
6F	Diagnostic write attempted while write protected
70	Command available timeout error during execution of PCB wrap test
71	SDI INIT always asserted
72	No control clock or framing error occurred during execution of PCB wrap test
73	Microprocessor and PCB board data bus communication error
74	Initial personality board status incorrect for off-line condition
75	Failure in control clock error detect circuit
76	Failure in data clock error detect circuit
77	Port A select latch failed
78	Port A data transmitter/receiver error
79	Port A control line transmitter/receiver error
7A	Port B select latch failure

Drive Error Display Codes (Cont)

Error Code	Name
7B	Spindle motor spinning while attempting PCB test
7C	Received error in frame code during PCB wrap test
7D	Received error in data byte during PCB wrap test
80	Consistency fault detected in master ROM
81	Command available did not reset after reading data
82	Frame code did not reset after reading data
83	Cannot receive SDI INIT from Port B
84	INIT bit not clear following INIT
85	Master RAM 0 failed RAM test
86	Master RAM 1 failed RAM test
87	Checksum error detected in master ROM 0
8F	Checksum error detected in master ROM 1
90	Port B data transmitter/receiver error
91	Port B control line transmitter/receiver error
92	Port A wrap-around error
93	Response serializer error
94	Loop-around frame not received properly
95	Loop-around frame was not decoded properly
96	Loop-around frame data byte was not received properly
97	Checksum error detected in master ROM 2
9F	Checksum error detected in master ROM 3
A0	Unable to clear faults at R/W safety register
A1	Unable to force head short error while writing

Drive Error Display Codes (Cont)

Error Code	Name
A2	Unable to force multiple head select error while reading
A3	Unable to force read gate and write gate active at the same time
A4	Unable to force write current without write gate
A5	Unable to force write gate without write current
A6	Unable to force a miscompare of the data from the write compensation logic and the read encoder logic
A7	Checksum error detected in master ROM 4
A9	Servo fault caused a R/W forced fault
AA	Servo fault caused a R/W forced fault
AA	Diagnostic firmware OK prompt code
AF	Checksum error detected in master ROM 5
B0	Test will not run due to a drive fault
B3	Bus test error in microprocessor module
B4	Bus test error in personality module
B6	UART transmitter/receiver error
B8	Stream test write clock not active
B9	Stream read clock not active
BA	Stream buffered servo clock not active
BB	Stream ECL write data not present
BC	Stream encoded data not present
BD	Stread decoded read data not present
BE	Stream ECL read data not present
BF	Unable to reset stream bit
C0	Fine track timer error

Drive Error Display Codes (Cont)

Error Code	Name
D9	Load from inner guard band failed
DA	Spindle not spinning
DB	Diagnostic seek failed
DC	Incorrect parameters entered during execution of drive-resident diagnostics
DD	Recalibrate error
DE	Low velocity seek error
DF	Spinup failed
E0	Random seek error
E1	Integrator error
E2	Slave ROM consistency error
E3	Checksum error on slave ROM 0
E4	Checksum error on slave ROM 1
E5	Checksum error on slave ROM 2
E6	Checksum error on slave ROM 3
E7	Test execution code
E8	Error in slave RAM 0
E9	Error in slave RAM 1
EE	Incorrect test parameter entered
F0	Slave uncommanded spin-down
F1	Slave load timeout
F2	Slave sent an unexpected message
F3	Slave received a bad command packet

Drive Error Display Codes (Cont)

Error Code	Name
F4	Slave operation code parity error
F5	Slave received an invalid operation code
F6	Master received a bad status packet from slave
F7	Slave recalibrate timeout NOTE This error could be the result of the positioner lock lever in the lock position.
F8	Slave seek timeout
F9	Slave offset timeout
FA	Slave spin-up timeout
FB	Slave spin-down timeout
FC	Slave send status timeout
FD	Slave initialization timeout
FE	Slave speed check timeout

FRONT PANEL FAULT IDENTIFICATION
To identify faults called out by the front panel indicator lights, the following table shows the status of the front panel lights for each type of error reported.

Drive Front-Panel Fault Identification Codes

Fault Condition	RUN/ STOP	FAULT	RDY	WRITE PROT	A	B	Status Byte 15 Hex Code
Index error	on	on					11
Master/slave error		on		on			12
Servo diagnostic test error	on	on		on			13
Microprocessor fault		on			on		14
Drive disabled by DD bit	on	on			on		15
Servo coarse positoning error		on		on	on		16
Diagnostic idle loop test error	on	on		on	on		17
Spin error		on				on	18
Write and bad embedded data	on	on				on	19
Servo fine positoning error		on		on		on	1A

Drive Front-Panel Fault Identification Codes (Cont)

Fault Condition	RUN/ STOP	FAULT	RDY	WRITE PROT	A	B	Status Byte 15 Hex Code
R/W diagnostic test error	on	on		on		on	1B
SDI error		on			on	on	1C
Write enable and write protect asserted error	on	on			on	on	1D
Spindle motor interlock error		on		on	on	on	1E
Servo or HDA overtemp error		on	on				30
Servo/microprocessor interlock error		on	on	on			32
R/W command error		on	on		on		34
Control panel/ microprocessor interlock error		on	on	on	on		36
R/W unsafe error		on	on			on	38
Read/write microprocessor interlock error		on	on	on		on	3A
Servo/HDA interlock error		on	on		on	on	3C
Personality/ microprocessor interlock error		on	on	on	on	on	3E
Microprocessor hardcore test error	on	on	on	on	on	on	3F
DC low	on		on	on	on	on	*

NOTE

The DC Low condition locks up the drive logic and therefore cannot have a hexadecimal code represented by the host system diagnostics.

RA60 FRONT PANEL CODES
The following table provides a description of the fault, indicates the front panel light configuration, and identifies the HEX code of the lights.

Insert Art

Description of Error	RUN/STOP	FAULT	RDY	WRITE PROT	A	B	Hex Code
Microcode error (not sent to front panel)	*	off	off	off	off	off	00
Heads home switch fault	*	on	off	off	off	on	01
Front panel fault	*	on	off	off	on	off	02
Long spin-up time fault	*	on	off	off	on	on	03
Bad servo samples fault	*	on	off	on	off	on	05
Bad velocity fault	*	on	off	on	on	off	06
Linear mode fault	*	on	off	on	on	on	07
Retry on seek fault	*	on	on	off	off	off	08
Lost servo samples fault	*	on	on	off	off	on	09
Heads home switch would not open	*	on	on	off	on	off	0A
Master processor fault	*	on	on	off	off	off	10
SDI fault	*	on	on	off	off	on	11
Invalid SDI level 1 command	*	on	on	off	on	on	13
Master processor fault	on	on	on	on	on	on	3F

*The run/stop switch may be on or off depending on the state of the drive.

RA60 FRONT PANEL FRU CALLOUT
The fault code, a description of the fault, and the most likely failing FRUs are shown below.

RA60 Front Panel Fault Codes

Front Panel Codes	Description	Most Likely FRU Failure
00	Microcode error (not sent to front panel	1. None
01	Heads home switch fault	1. Push heads home 2. Check P803 3. Heads home switch assembly
02	Front panel fault	1. Front panel module 2. Drive logic module 3. Front panel ribbon cable
03	Long spin-up time fault	1. Check spindle motor connections 2. Heat sink module 3. Spindle motor 4. Drive logic module

RA60 Front Panel Fault Codes (Cont)

Front Panel Codes	Description	Most Likely FRU Failure
05	Bad servo samples fault	1. Pack 2. Post amp/data separator module 3. R/W preamplifier module 4. Heads
06	Bad velocity fault	1. Pack 2. Post amp/data separator module 3. R/W preamplifier module 4. Drive logic module 5. Positioner assembly
07	Linear mode fault	1. Pack 2. Post amp/data separator module 3. Drive logic module

RA60 Front Panel Fault Codes (Cont)

Front Panel Codes	Description	Most Likely FRU Failure
08	Retry on seek fault	1. Pack 2. Post amp/data separator module 3. R/W preamplifier module 4. Drive logic module
09	Lost servo samples fault	1. Pack 2. Heads 3. Post amp/data separator module
0A	Heads home switch would not open	1. Shipping pin not removed 2. Check P803 3. Heads home switch assembly 4. Heatsink module

RA60 Front Panel Fault Codes (Cont)

Front Panel Codes	Description	Most Likely FRU Failure
10	Master processor fault	1. Drive logic module 2. SDI module 3. Post amp/data separator module 4. Backplane module
11	SDI Faults	1. SDI Module 2. Drive logic module 3. SDI cable 4. Backplane module

RA60 Front Panel Fault Codes (Cont)

Front Panel Codes	Description	Most Likely FRU Failure
13	Invalid SDI level 1 command	1. SDI module 2. Post amp/data separator module 3. R/W preamplifier module 4. Backplane module
3F	Master processor fault	1. Drive logic module 2. SDI module 3. Post amp/data separator module 4. Backplane module

RA60 ERROR MESSAGES
The following table provides the error code, a description of the error, and indicates the most likely failing Field Replaceable Units (FRUs).

RA60 Error Messages

Error Code	Description	Most Likely FRU Failure
01	Command cannot be executed with the drive in its current state	1. None
02	Nonexistent head or cylinder requested with seek	
03	Opcode was not one of six valid opcodes	
04	Cover was not closed when run was issued	
05	Lid not locked when run command was issued	
06	Program error during head load	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
08	Heads were not home when run command was issued	1. Push heads home 2. Check P803 3. Heads home switch assembly
09	Heads home switch opened during spin-up	
0A	Heads home switch would not close during unload	
0B	Cover or lid lock opened during head load	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
0C	Spin-up L not low during spin-up	1. Front panel module
10	Lid lock open during spin-up	2. Drive logic module
11	Cover open during spin-up	3. Front panel ribbon cable
12	Cover or lid lock opened during head load	
13	Lid not locked while run command was issued	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
18	Spin-up required too long a time	1. Check spindle motor connections
19	Motor sample would not change during spin-up	2. Heat sink module
20	Spin-up required to short a time	3. Spindle motor
24	Too long to acquire spindle speed control on head load	4. Drive logic module

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
26	More than 32 sectors to settle on track	1. Pack
27	More than 32 sectors to settle on track retry failure	2. Post amp/data separator module
28	Too many bad servo samples during RTZ	3. R/W preamplifier
29	Too long to acquire good samples on head load	4. Heads
2A	Head load seek did not return to track 0	
2B	Bad servo samples when bumping guard band on head load	
2C	Bad servo samples during RTZ on head load	
2D	Bad samples during linear mode	
2E	Off speed when in linear mode	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
2F	Off track in linear state	1. Pack
30	Velocity too high during RTZ on head load	2. Post amp/data separator module
31	Velocity too low during RTZ on head load	3. R/W preamplifier module
32	Bad Velocity on seek	4. Drive logic module
		5. Positioner assembly
38	Too long to get on track in linear mode	1. Pack
		2. Post amp/data separator module
		3. Drive logic module

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
40	Retry on seek due to bad servo samples	1. Pack 2. Post amp/data separator module 3. R/W preamplifier module 4. Drive logic module
41	Retry on seek due to bad servo samples failed	
42	Retry on seek due to bad guard band flags	
43	Retry on seek due to bad guard band flags failed	
44	Retry on see due to seek timeout	
45	Retry on see due to seek timeout failed	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
46	Retry on seek due wrong grey code	1. Pack
47	Retry on seek failed due to wrong grey code	2. Heads
48	Lost servo samples	3. Post amp/data separator module
50	Too long for heads home switch to open on head load	1. Shipping pin not removed 2. Check P803 3. Heads home switch assembly 4. Heatsink module

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
81	Inits and tests control panel	1. Drive logic module
82	Checks processor registers	2. SDI module
83	Tests RAM #0	3. Post amp/data separator module
84	Tests ROM #0	4. Backplane module
85	Tests ROM #1	
86	Tests ROM #2	
87	Verify ROM version numbers	
88	Tests the SDI clear interface bit	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
89	Tests the SDI control/status register	1. Drive logic module 2. SDI module 3. Post amp/data separator module 4. Backplane module
8A	Tests front panel and serial number ROM	
8B	Inits the UART	
8C	Test the error registers	
8D	Test the 8155 timer	
8E	Test the slave control port	
8F	Slave Diagnostics	
90	Command available or slave done already set when issuing a new slave command	
91	Slave command receiver timeout on opcode	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
92	Slave attention timeout	1. Drive logic module
93	Slave done timeout to slave stop (TOSTPO	2. SDI module
94	Slave done timeout to status (1 ms)	3. Post amp/data separator module
95	Slave done timeout	4. Backplane module
96	Slave attention timeout	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
97	Solenoid release timeout	1. Drive logic module
98	Watchdog timer detected master insane	2. Post amp/data separator module
99	Watchdog timer detected slave insane	3. SDI module
9A	Run switch or cover invalid at spin-up	
9B	Spindle not ready during recalibrate command	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
9C	SDI transfer error from DC703	1. SDI module
9D	Write or format when write protected	2. Post amp/data separator module
9F	Transfer command when drive error	4. Backplane module
A0	Sector read/write overrun error	
A1	Transfer command when read/write error	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
A2	SDI command checksum error	1. SDI module
A3	SDI frame error	2. Drive logic module
A4	SDI command parity error	3. Post amp/data separator module
A5	SDI command out of range	4. Backplane module
A6	SDI command length error	
A7	SDI error status byte was nonzero	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
A8	Slave done timeout to seek command	1. Drive logic module 2. SDI module 3. Post amp/data separator module 4. Backplane module
A9	Slave done timeout to RTZ command	
AA	Slave response error to RTZ command	
AB	Slave done timeout during first pass	
AC	Command receiver timeout during diagnostic stop or ASCII port command time = todgn 2	
AD	Slave done timeout during diagnostic stop or ASCII port time = 1 MS	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
AE	Drive not on-line	1. Drive logic module
AF	Command byte 4 & 5 are not zero	2. Post amp/data separator module
B0	Invalid group number	3. SDI module
B1	Invalid head select number	4. Backplane module
B2	Invalid cylinder range	
B3	Spindle not ready	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
B4	Controller timeout = 0	1. Drive logic module
B5	Spindle not ready	2. SDI module
B6	Illegal memory region	3. Post amp/data separator module
B7	DD bit set	4. Backplane module
B8	Tried to clear a hard fault	
B9	Slave done timeout (1 MS) to get status	

RA60 Error Messages (Cont)

Error Code	Description	Most Likely FRU Failure
BA	Nonzero level	1. Drive logic module
BB	Wrong subunit error	2. SDI module
BC	Diagnostic command, memory region hi not zero	3. Post amp/data separator module
BD	Diagnostic command, invalid parameters	4. Backplane module
BE	Write enable a write protected drive	
BF	Setting S7 = 1	

REGULATOR MODULE LEDs
The following table provides possible causes to Regulator Module LED indications which can be used in isolating power supply problems.

Regulator LEDs Fault Codes

Green D19 REF	Red D18 Thermal Fault	Red D17 +27.5 -27.5	Green D16 +5	Green D15 -5.2	Green D14 -15	Green D13 +15	Possible Causes Listed in order of most likely
1	0	0	1	1	1	1	Normal State
0	0	0	0	0	0	0	Suspect ref on Reg 1. Check AC cabling and switch plate assembly 2. Transformer 3. Regulator 4. Cap/Rec Assembly
0	0	1	1	1	1	1	27.5 volt failure 1. Heat Sink Module 2. Cap/Rec Assembly 3. Transformer 4. Regulator
0	0	0	at least one off				Suspect a short ckt 1. Logic module shorts 2. Heat Sink Module
0	0	0	0	0	1	1	Suspect short or over-voltage to +5 and -5.2 1. Logic Module Shorts 2. Check Ckt Breaker 3. Heat Sink Module 4. Capacitor Assembly 5. Regulator Module
0	0	0	1	1	0	0	Suspect shorts to 15 V 1. Logic Module Shorts 2. Check Ckt Breaker 3. Heat Sink Module 4. Capacitor Assembly 5. Regulator Module
0	1	0	1	1	1	1	Thermal Shutdown 1. Check Fans 2. Regulator Module
0	1	0	at least on off				Thermal Shutdown with Faulted Supply 1. Check Fans 2. Heat Sink Module 3. Regulator Module

HSC 50 ERROR CODES
The following table lists the operator control panel fault codes for the HSC50.

Operator Control Panel Fault Codes

	HEXA-DECIMAL	OCTAL	INIT	FAULT	ON LINE		
PORT PROCESSOR MODULE FAILURE	01	01	OFF	OFF	OFF	OFF	ON
DISK DATA CHANNEL FAILURE	02	02	OFF	OFF	OFF	ON	OFF
TAPE DATA CHANNEL FAILURE	03	03	OFF	OFF	OFF	ON	ON
I/O CONTROL PROCESSOR MODULE FAILURE	11	21	ON	OFF	OFF	OFF	ON
MEMORY MODULE FAILURE	12	22	ON	OFF	OFF	ON	OFF
TU58 FAILURE	13	23	ON	OFF	OFF	ON	ON
PORT BUFFER MODULE FAILURE	14	24	ON	OFF	ON	OFF	OFF
PORT LINK MODULE FAILURE	15	25	ON	OFF	ON	OFF	ON
MISSING REQUIRED FILES	16	26	ON	OFF	ON	ON	OFF
ERROR LOG ATTENTION REQUIRED	17	27	ON	OFF	ON	ON	ON
NOT ENOUGH WORKING REQUESTORS IN SUBSYSTEM	18	30	ON	ON	OFF	OFF	OFF
REBOOT BEFORE PREVIOUS BOOT COMPLETE	19	31	ON	ON	OFF	OFF	ON
SOFTWARE DETECTED INCONSISTENCY	1A	32	ON	ON	OFF	ON	OFF

MR-15101

NOTE
If the OCP Power indicator is not on, ensure that the ac power cord is plugged in and that the computer room circuit breakers are switched on. If the Power indicator still does not come on, call your field service office.

CLUSTER TROUBLESHOOTING

Table of Contents

NOTE

Material planned for this section has been rescheduled for the next update of this manual.

Table of Contents

MASSBUS REGISTERS
This section describes the Massbus registers and gives detailed information on the status and error bits of each.

Massbus Registers

Massbus Register Number (Octal)	Register Name		Mode of Operation
00	(RPCS1)	Control and Status Register	Read/Write
01	(RPDS)	Drive Status Register	Read Only
02	(RPER1)	Error Register 1	Read Only*
03	(RPMR1)	Maintenance Register 1	Read/Write
04	(RPAS)	Attention Summary Pseudo-Register	
05	(RPDA)	Desired Track/Sector Address Register	Read/Write
06	(RPDT)	Drive Type Register	Read Only
07	(RPLA)	Look Ahead Register	Read Only
10	(RPSN)	Serial Number Register	Read Only
11	(RPOF)	Offset Register	Read/Write
12	(RPDC)	Desired Cylinder Address Register	Read/Write
13	(RPCC)	Current Cylinder Address Register	Read Only
14	(RPER2)	Error Register 2	Read Only*
15	(RPER3)	Error Register 3	Read Only*
16	(RPEC2)	ECC Position Register	Read Only
17	(RPEC2)	ECC Pattern Register	Read Only

*Drive resident microdiagnostic routines test set and reset capabilities.

Register 00_8 - Control and Status Register (RPCS1)
This Read/Write register is used to initiate all RP07 command operations. It is physically shared by RP07 Device Control Logic (DCL) and the RH20 Controller. RP07 uses seven of the control register's 16 bits (0-5 and 11); RH20 contains the remaining 9 bits.

Bit 0 (GO) - A command (bits 1-5 in RPCS1) is always transmitted with the GO bit set. When set, GO causes the RP07 DCL to do the following.

1. Decipher the function code (bits 1-5 in RPCS1).

2. Determine if the function code is illegal and, if so, set the appropriate error bit.

3. Determine if the command is a data transfer command.

   a. If the decoded command is a data transfer command, assert the OCC (Massbus Occupied) line within 50 microseconds and execute the function if the RUN (Massbus Run) line becomes asserted.

   b. If the decoded command is not a data transfer command, the RP07 executes the function called for (a microdiagnostic routine, for example).

Bits 1-5 (Function Code) - Function Code bits contain the particular commands to be executed by the RP07. Function codes are listed in the table below.

Function Codes

Command Code Octal	Command	Function Code and GO F4	F3	F2	F1	F0	GO
01	No-Op	0	0	0	0	0	1
03	Illegal	0	0	0	0	1	1
05	Seek	0	0	0	1	0	1
07	Recalibrate	0	0	0	1	1	1
11	Drive Clear	0	0	1	0	0	1
13	Release	0	0	1	0	1	1
15	Offset	0	0	1	1	0	1
17	Return to Centerline	0	0	1	1	1	1
21	Read in Preset	0	1	0	0	0	1
23	No-Op	0	1	0	0	1	1
25	Illegal	0	1	0	1	0	1
27	Illegal	0	1	0	1	1	1
31	Search	0	1	1	0	0	1
33	Illegal	0	1	1	0	1	1
35	Microdiagnostic	0	1	1	1	0	1
37	Illegal	0	1	1	1	1	1
41	Illegal	1	0	0	0	0	1
43	Illegal	1	0	0	0	1	1
45	Illegal	1	0	0	1	0	1
47	Illegal	1	0	0	1	1	1
51	Write Check Data	1	0	1	0	0	1
53	Write Check Header and Data	1	0	1	0	1	1
55	Illegal	1	0	1	1	0	1
57	Illegal	1	0	1	1	1	1
61	Write Data	1	1	0	0	0	1
63	Format Track	1	1	0	0	1	1
65	Write Track Descriptor	1	1	0	1	0	1
67	Illegal	1	1	0	1	1	1
71	Read Data	1	1	1	0	0	1
73	Read Header & Data	1	1	1	0	1	1
75	Read Track Descriptor	1	1	1	1	0	1
77	Illegal	1	1	1	1	1	1

Bits 6-10 - These bits reside in the RH20.
Bit 11 (DVA) - Device Availability (DVA) is always set when RPCS1 is read by Massbus. The input to the buffer for DVA (J42 Pin 8) is hardwired to ground. In dual access mode the controller that has access sees DVA set; the controller that does not have access sees all zeros plus the parity bit set. In single access mode, DVA is always set when read by the controller.

Bits 12-15 - These bits reside in the RH20 controller.

Register 01_8 - Device Status Register (RPDS)
This read-only register contains nonerror indicators.

Bit 0 (OM) - Offset Mode (OM) sets when an Offset Command (Function Code and GO, bits 5-0 in RPSC1) is written.

OM Reset condition generating ATTENTION INTERRUPT:

- Return to Centerline command

OM Reset conditions - NO ATTENTION INTERRUPT generated:

- Any write command
- Seek
- Implied Seek
- Mid-transfer Seek
- Search
- Recalibrate
- Read in Preset
- Power on Reset
- Release

Bit 1 (EWN) - Early Warning (EWN) is set when RP07 thermal or air flow sensors detect an out-of-limit condition in the drive, and remains set as long as the condition persists.

EWN is a status bit only and will not cause an attention interrupt but will, when set, cause the UNSAFE indicator (located on the operator control panel) to blink.

If an early warning condition is detected <u>before</u> a spindle-start sequence, spindle-start is inhibited. Detection <u>after</u> a spindle-start sequence does not generate a spindle-stop sequence; however, the UNSAFE indicator blinks to inform the operator of an unsafe condition. Continued rise in temperature causes CB3 to trip, removing ac power to the drive.

Bit 2 (ILV) - Interleaved Sectors (ILV) is set by the logic that enables sector interleaving. ILV is field-programmable by a hardware jumper on the backpanel (J26). See Chapter 2 of the RP07 Service Manual for the jumper configuration.

Bits 3-5 - Always 0.

Bit 6 (VV) - Valid Volume (VV) is presented to the host as a 1.

Bit 7 (DRY) - When Drive Ready (DRY) is set, the drive is ready to accept commands. DRY is the complement (opposite state) of GO (bit 0 in RPCS1): if GO is set when RPDS is read, DRY is negated.

Bit 8 (DPR) - In a dual access drive, Drive Present (DPR) is set to the RH20 controller that has access to the RP07 and reset to the other RH20 controller. In single access mode, the DPR bit is always set when read by the RH20 controller.

Bit 9 (PGM) - In a dual access drive, the Programmable (PGM) bit is set when the drive is equally available to both controllers and the Access A, A/B, B switch on the operator control panel is in the A/B position at the time of drive transition from off-line to on-line. Transition occurs after successful completion of a start-spindle sequence or after the Online switch is placed from off-line to on-line position.

In single access mode, when DRQ (Drive Request Required, bit 11 in RPDT) is reset, PGM is negated (reset), the Access switch is ignored, and access "A" is forced true.

Bit 10 (LBT) - Last Block Transferred (LBT) is set by the RP07 DCL during a Data Transfer Command when data is being transferred to the last addressable sector of user media.

LBT resets under any of the following conditions.

- A new command is issued
- Massbus Initialize
- Drive Clear
- Power On Reset

Bit 11 (WRL) - Write Lock (WRL) reflects the true write protect condition of the drive logic as a result of assertion of the Write Protect switch on the operator control panel. WRL will not set if a write command is in progress; set is deferred until completion of the write operation.

Bit 12 (MOL) - Medium Online (MOL) is set when the drive is ready to accept commands after a successful spindle-start sequence and when the Online switch is in on-line position. MOL must be set prior to initiation of any command except when the RP07 is in microdiagnostic mode.

MOL is reset whenever the drive enters one of the following states in which commands cannot be executed.

- Power-down sequence

- Unsafe condition

- Change from on-line to off-line position at the operator control panel; off-line start is deferred until command completion

Bit 13 (PIP) - Position in Progress (PIP) is set whenever the drive positioner is in motion.

PIP is reset at completion of the movement.

The following table shows a list of relationships between PIP and the type of operation being performed.

DRY-PIP-ATA Status During Operations

Operation	DRIVE READY (DRY)	POSITION IN PROGRESS (PIP)	ATTENTION (ATA) AT END OF OPER
No operation	0	0	No
Recalibrate	0	1	Yes
Offset	0	0*	Yes
Drive Clear	0	0	No
Return to Centerline	0	0*	Yes
Seek (including 0 cylinder)	0	1	Yes
Write Check	0	0**	No
Write Data	0	0**	No
Write Header and Data	0	0**	No
Read Data	0	0**	No
Read Header and Data	0	0**	No
Implied Seek	0	1	No
Mid-Transfer Seek	0	1	No
Read in Preset	0	0	No
Search	0	0**	Yes
Microdiagnostic	0	0	Yes

*PIP sets if command execution time exceeds the current Massbus cycle time.
**PIP is set during the implied seek portion of the command.

Bit 14 (ERR) - Composite Error Status (ERR) is the OR (Inclusive OR) of all register error bits. ERR is reset by one of the following conditions - provided the error is not persistent.

- o Drive Clear
- o Massbus Initialize
- o Power On Reset

A composite error set at initiation of a command other than a Drive Clear or a Microdiagnostic command will inhibit execution of the command and prevent the GO bit from being set.

Bit 15 (ATA) - Attention Active (ATA) indicates the state of the Attention flip-flop for the switched/seized RH20 controller.
ATA is set under any of the following conditions.

- o Any error in the error registers
  At occurrence if GO bit is reset
  At completion of a command if GO bit is set

- o On a Write to any register when Composite error is set except the Attention Summary register (RPAS) or maintenance registers, writing Microdiagnostic command, or Drive Clear function codes with the GO bit in the Control register

- o Completion of a Seek, Search, Recalibrate, Offset, Return to Centerline, or Microdiagnostic command

- o Whenever MOL changes state

- o In a dual access RP07 when access request flip-flop (DRQ, bit 11 in RPDT) is set for one controller and the other controller releases.

ATA is reset under any of the following conditions.

- o Writing the GO bit when ERR is reset

- o Drive Clear (if error is not persistant and GO is not set)

- o Massbus Initialize

- o Writing a 1 into the Attention Summary Pseudo-Register (RPAS) bit position that corresponds to the RP07 logical drive address

Register 02_8 - Error Register 1 (RPER1)
This read only register contains individual error condition indicators.

The RP07 error conditions fit into one of two basic categories:

- CLASS A errors, which can be handled at the completion of a non-data transfer command, at a convenient block boundary

- CLASS B errors, which must be handled immediately; a class B error causes the drive to terminate command execution as soon as possible.

All nonpersistant error bits in Error Register 1 are reset (cleared) under any of the following conditions.

- Drive Clear
- Massbus Initialize
- Power On Reset

Bit 0 (ILF) - Illegal Function (ILF) is set when a function code and GO bit are written into the RPCS1 and the code does not correspond to an implemented command in the RP07.

ILF is a CLASS B error.

Bit 1 (ILR) - Illegal Register (ILR) is set when a read or write command is attempted to or from a nonexistent register. Trying to write into a read-only register does not set ILR.

ILR is a CLASS A error.

Bit 2 (RMR) - Register Modification Refused (RMR) is set when a write command is attempted to an existing drive register (except the RPAS) while the GO bit is set and an operation is in progress.

RMR is a CLASS A error.

Bit 3 (PAR) - The parity (PAR) error bit is set:

- By DPE (Data Parity Error, bit 3 in RPER3) when a parity error is detected on a Massbus data line when writing data on the media (CLASS A error), or

- When a parity error is detected on a Massbus control line when writing into a register (CLASS B error).

PAR applies to data or control information being transmitted only from the RH20 controller to the RP07; the RP07 checks for the presence of odd parity.

Bit 4 (FER) - Format Error (FER) is set after reading an entire header if bit 12 of the first header word does not match FMT (bit 12 in RPOF).

FER is a CLASS A error during a Read and/or Write Check Header and Data Command.

FER IS A CLASS B error for all others.

Bit 5 (WCF) - Write Clock Fail (WCF) is set during a write operation if the RP07 fails to receive a response to a request for data (write clock) from the RH20 within one word time.

WCF is a CLASS B error.

Bit 6 (ECH) - ECC Hard (ECH) error is set when a Data Check (DCK, bit 15 in this register) cannot be recovered by using ECC.

ECH is a CLASS B error.

Bit 7 (HCE) - Header Compare Error (HCE) is set while reading the header if one or more of the following occurs.

- The cylinder address bits, 0-9 in the first header word, do not match the contents of the Desired Cylinder Address register (RPDC) bits 0-9.
- The sector address bits, 0-6 in the second header word, do not match the contents of the Desired Sector/Track Address register, bits 0-6.
- The track address bits, 8-13 in the second header word, do not match the contents of the Desired Sector/Track Address register, bits 8-13.
- Bits 13, 11, and 10 in the first header word or bits 15, 14, and 7 in the second header word are not 0.

Bits 15, 14, and 12 of the first header word are ignored by the header compare logic.

HCE is a CLASS A error that causes termination of the command in progress after reading the entire header, unless the command in progress is a Read or Write Check Header and Data Command, or HCI (Header Compare Inhibit, bit 10 in RPOF) is asserted, in which case HCE is a CLASS A error.

Bit 8 (HCRC) - Header CRC (HCRC) is set when the CRC register is nonzero after reading the entire header and redundancy bytes.

In the above case HCRC is a CLASS B error, which causes termination of command in progress after reading the entire header, unless:

1. The command in progress is a Read or Write Check Header and Data, or
2. HCI (Header Compare Inhibit, bit 10 in RPOF) is asserted.

In the above two cases, HCRC is a CLASS A error.

Bit 9 (AOE) - Address Overflow Error (AOE) is set when the RH20 attempts to continue data transfer beyond the last user-available sector causing a cylinder address overflow. When AOE is set, the sector and track count in the RPDA and the cylinder value in the RPDC are incremented at EBL assertion.

AOE is a CLASS B error.

Bit 10 (IAE) - When the contents of the RPDC or the RPDA are invalid, Invalid Address Error (IAE) is set as a result of any of the following commands.

- Seek
- Search
- Read Header and Data
- Read Data
- Write Check Header and Data
- Write Check Data
- Format Track
- Write Data
- Write Track Descriptor
- Read Track Descriptor

See the following table for valid addresses.

Valid Addresses

Address	**Functional Mode** 16-Bit	18-Bit	**Diagnostic Mode** 16-Bit	18-Bit
Desired cylinder	0-629	0-629	0-631	0-631
Desired head	0-31	0-31	0-31	0-31
Desired sector	0-49	0-42	0-49	0-42

IAE is a CLASS B error.

Bit 11 (WLE) - When a write operation is attempted on a drive that is in write lock mode, Write Lock Error (WLE) is set. During a write command, if the Write Protect switch on the operator control panel becomes asserted, no error condition results; the current write operation completes.

WLE is a CLASS B error.

Bit 12 (DTE) - Drive timing error bit is set if during a data transfer a timing failure is detected by the drive logic. The DTE error is a CLASS B error that causes immediate termination of the command in progress.

Bit 13 (OPI) - Operation Incomplete (OPI) is set under any of the following conditions.

1. If during an implied seek the RP07 does not find the correct sector within three revolutions from the start of a search while executing a search or data transfer command.

2. Failure to detect INDEX pulse for three revolutions on commands that are oriented on the index marker, following:

   - Format track
   - Read Track Descriptor
   - Write Track Descriptor
   - Search
   - Read Check Header and Data for sector 0
   - Write Check Header and Data for sector 0

OPI is a CLASS B error.

Bit 14 (UNS) - Unsafe (UNS) is the inclusive OR of the following errors that make the RP07 unsafe for normal operation.

1. R/W Unsafe #1 (RWU1, bit 10 in RPER2)
2. R/W Unsafe #2 (RWU2, bit 11 in RPER2)
3. R/W Unsafe #3 (RWU3, bit 12 in RPER2)
4. DC Unsafe (DCU, bit 5 in RPER3)
5. Tach Calibration Failure
6. CPU Unsafe
7. All other permanent error conditions

All above error conditions are CLASS B errors.

Bit 15 (DCK) - Data Check (DCK) is set at completion of reading data and the ECC (Error Correction Code) field of a sector if the ECC register bits 11-31 are nonzero.

DCK is a CLASS A error if the ECC Inhibit (ECI, bit 11 in RPOF) is set.

DCK is a CLASS B error if ECI is reset. The command is then terminated at completion of the error correction process.

Register 03_8 - Maintenance Register (RPMR1)
Host processor software gains access to the drive-resident microdiagnostics through the Massbus Maintenance Register. This read/write register allows the host to initiate RP07 microdiagnostic routines and monitor microdiagnostic results.

Bits 0-7 - The host enters parameters to be used during execution of a specific microdiagnostic routine.

Bits 8-14 - Routine number bits, written by the host, direct the RP07 to run a specific microdiagnostic routine.

Bit 15 (DMD) - The Diagnostic Mode (DMD) bit is written by the host to enable operation in microdiagnostic mode. When set, bit 15 will:

- Disable write operations on all cylinders except FE cylinders

- Enable execution of a specific microdiagnostic routine

- Enable execution of commands with MOL reset

- Enable access to FE cylinders

Register 04_8 - Attention Summary Pseudo-Register (RPAS)
This read/write register is called a pseudo-register because it is implemented as one bit in each drive. Each RP07 has one flip-flop that, depending on the logical location of the drive, corresponds to the appropriate line on the asynchronous Massbus control lines. The RPAS allows the host to see where the drive requesting attention is located. To see the cause of ATA, the RH20 controller will then read the RPDS.

To read RPAS does not require that Massbus DS (Device Select) address lines be used; all drives respond each time the Massbus addresses RPAS. RPAS is the only register that may be read in this manner.

To write RPAS requires that Massbus DS address lines be used.

Bits 0-7 - Bit 0 is the Attention Active (ATA) bit of drive 0; bit 1 is the ATA bit of drive 1, and so on through drive 7.

Bits 8-15 - Bits 8-15 are 0s.

READING the RPAS - Because the host does not have to specifically address a drive in order to read the RPAS, the RH20 controller will generally request Attention Summary status from all drives simultaneously by indicating a "Read from Register 04_8" on the Massbus register select lines and raising the Demand pulse.

When "register 04_8" is selected, each drive recognizes the 04 address and places the output of its ATA flip-flop in its assigned position on one of the control lines. For example, drive 0 places ATA0 on Massbus line 0. The parity line is ignored since, on a read, parity cannot be generated in the drive.

After placing the Register 04 address on the Register Select line, asserting Demand, and receiving an ATA from each drive, the RH20 strobes the ATA bits in order to read the results.

RP07 will inhibit displaying the RPAS when it senses the negation of Demand.

WRITING in the RPAS - The attention summary flip-flop status on each drive can be altered by the RH20. Each drive receives a bit from the Massbus control lines; if the bit is set, the drive resets its ATA bit. To clear the attention bit the unit must be selected in RPCS2 and a 1 must be written into the appropriate bit. See the following table for the effect of writing an ATA bit.

Writing ATA Bits

Bit Written	ATA Before	ATA After
0	0	0
0	1	1
1	0	0
1	1	0

Writing a 1 causes a set bit to be reset.
Writing a 0 has no effect.

This write operation allows for reset of ATA bits that have already been seen and acted upon without accidentally resetting other ATA bits that may have become set in the meantime.

On a write, the controller presents the Register 04 address (176716_8) on the Massbus Register Select and DS lines and raises the Demand pulse.

Following the rise of Demand, the Massbus control bus lines with Attention Summary information are strobed by the RP07 selected by DS 2-0. The information is valid until negation of Demand.

For a write operation, parity will be generated by the RH20 but will only be checked on the controller that is switched/seized to RP07.

The RP07 must respond with the Transfer pulse.

When the RP07 sets the Attention Line without ERR (Composite Error, bit 14 in RPDS) set, drive logic will accept any command and reset ATA.

In the event of a hard (persistent) error, ATA must be reset by writing a 1 in the appropriate bit position so that all drives on the Massbus are not rendered inoperable by the attention line's constant assertion. The drive error will remain set. If the RH20 attempts a write in any register except the RPAS or attempts a command other than a Drive Clear or a microdiagnostic, the ATA flip-flop will set again.

Register 05_8 - Desired Sector/Track Address Register (RPDA)
This read/write register provides spiral transfer capability; it increments automatically at EBL (End of Block) during a data transfer command, relieving software from updating the register on multiple block transfers.

The RPDA is cleared by:

- Read in Preset
- Power On Reset
- Writing 0 via the Massbus

RPDA data will not change during a Massbus read cycle.

Bits 0-6 (SA) - These bits compose the Desired Sector Address (SA) field.

Bit 7 - Always 0.

Bits 8-13 (TA) - These bits compose the Desired Track Address (TA) field.

Bits 14-15 - Always 0.

RPDA resets after the final sector and/or final track is transferred. See the following table for examples (assume 16-bit mode).

Example Final Transfers

	Current Track Address	**Current Sector Address**
During Transfer	00_8	00_8
After EBL	00_8	01_8
During Transfer	00_8	61_8
After EBL	01_8	00_8

RPDA increments at EBL pulse.

Register 06_8 - Drive Type Register (RPDT)
This read-only register is used to provide the software with information distinguishing the RP07 from other Massbus devices.

Reading the RPDT Register will cause the drive to send the drive type number and the appropriate parity bit (odd parity) to the unseized RH20 controller.

Bit 1 - Always 1.

Bits 0, 2, 3, 4 - Always 0.

Bit 5 - Always 1.

Bits 6-10 - Always 0.

Bit 11 (DRQ) - Drive Request Required (DRQ) is field-programmable via backpanel jumper (J26 pins 1 and 2). DRQ is set for dual access configuration (which must be requested before use and released after use). DRQ is reset for single access configurations.

Bit 12 - Always 0.

Bit 13 - Always 1.

Bits 14-15 - Always 0.

Register 07_8 - Look Ahead Register (RPLA)
This read-only register contains the exact rotational position of the heads in relation to the data track. Rotational position is monitored by a sector counter in the RP07. The counter is set to 0 at each index pulse, then is incremented each time a sector mark is encountered. The RPLA remains stable during a Massbus read cycle; however, the value may be incorrect if sampled at the time the counter changes value.

RPLA presents a sequential binary count regardless of the interleave state.

Maximum count is specified by FMT (bit 12 in RPOF): 49 in a 16-bit format; 42 in an 18-bit format. Changing FMT has an immediate effect on the SC field of RPLA.

Bits 0-5 - Always 0.

Bits 6-11 (SC) - These bits compose the Sector Counter (SC) field and are the only ones used in the RPLA.

Bits 12-15 - Always 0.

Register 10_8 - Serial Number Register (RPSN)
This read-only register displays the last four digits of the RP07 serial number in BCD. The drive serial number is factory hardwired on the backpanel at J26.

Bits 0-3 - Least significant BCD digit of serial number.

Bits 4-7 - Tens BCD digit of serial number.

Bits 8-11 - Hundreds BCD digit of serial number.

Bits 12-15 - Most significant BCD bit of serial number.

Register 11_8 - Offset Register (RPOF)
This read/write register is used for control information.

Bits 0-9 - These bits are presented by the RP07 as 0.

Bit 10 (HCI) - Header Compare Inhibit (HCI) is set by software to inhibit all header errors. When header errors occur with HCI set during Read or Write Check Data commands, they are classified as A errors.

HCI is reset by any of the following.

- Read in Preset
- Writing a 0 in bit 10 (HCI)
- Power On Reset

Bit 11 (ECI) - Error Correction Inhibit (ECI) is set by software to inhibit attempts by the RP07 to recover from a DCK (Data Check Error, bit 15 in RPER1) and to allow a Data Transfer command to continue beyond the sector where DCK occurred.

ECI is reset by any of the following.

- Read in Preset
- Writing a 0 in bit 11 (ECI)
- Power On Reset

Bit 12 (FMT) - The Format (FMT) bit, when set by software, enables the RP07 to operate in 16-bit mode. When reset, FMT enables 18-bit operation. Format is determined by FMT and maintained in RPLA.

FMT is reset by any of the following.

- Read in Preset
- Writing a 0 in bit 12 (FMT)
- Power On Reset

Bit 13 - Always 0.

Bit 14 (MTD) - Move Track Descriptor (MTD), when set, causes the Track Descriptor Record to be written an additional 64 bytes after the index pulse when a Write Track Descriptor command is initiated.

MTD is reset by any of the following.

- Read in Preset
- Writing a 0 in bit 14 (MTD)
- Power On Reset
- Completion of any command (GO resets)

Bit 15 (CMD) - When set as a failsafe by software, the Command Modifier (CMD) bit allows the following header handling commands: Read Track Descriptor, Write Track Descriptor, and Format Track.

CMD is reset by any of the following.

- Read in Preset
- Writing a 0 in bit 15
- Power On Reset
- Completion of any command (GO resets)

Register 12_8 - Desired Cylinder Address Register (RPDC)
This read/write register is loaded by software with the address of the cylinder that the positioner will move to on a Seek, Search, or Data Transfer command.

The content of RPDC is subject to change while GO is set.

The RP07 provides spiral transfer capability. Spiral transfer means that the software can continue reading through data tracks on a normal read.

Spiral transfer capability is realized when the RP07 is transferring data from the final sector and final track providing that at EBL:

1. RUN line is active, and
2. No error condition exists.

The desired cylinder address will then increment, and a seek to that address is automatically initiated, provided the address is valid.

RPDC is reset by any of the following.

- Read in Preset
- Writing a 0 in this bit position
- Power On Reset

Bits 0-9 (DC) - Desired Cylinder (DC) bits compose the desired address field; LSB is 0.

Bits 10-15 - Always 0.

Register 13_8 - Current Cylinder Address Register (RPCC)
This read-only register reflects the address of the cylinder below the read/write heads.

The RPCC is updated at completion of a positioning operation.

The content of RPCC is subject to change while GO is set.

RPCC is reset by:

1. A recalibrate operation, or
2. An initial head load when the positioner is loaded.

Bits 0-9 (CC) - Current Cylinder (CC) bits comprise the current address field; LSB is 0.

Bits 10-15 - Always 0.

Register 14_8 - Error Register 2 (RPER2)
This read-only register contains error indicators associated with RP07 and its internal control logic.

Errors are classified into CLASS A and CLASS B:

- CLASS A errors can be handled at the completion of a non-data transfer command or at a convenient block boundary in a data command.

- CLASS B errors are handled immediately. The drive terminates command execution as soon as possible.

Provided that an error is not hard (persistent), RPER2 is reset by any of the following.

- Power On Reset
- Drive Clear
- Massbus Initialize

Bits 0-7 - These eight bits contain error status for errors that are processed by the drive's 8080 microprocessor. When drive-resident microdiagnostics are initiated by the host processor and an error results, an error code - the result of that particular microdiagnostic - will be:

1. Reflected in bits 0-7 of Error Register 2
2. Illuminated in the eight LEDs on A1A07 PCA
3. Displayed on the FE panel.

Bit 8 (WRU) - The Write Ready Unsafe (WRU) bit sets during a write operation if write current is active and drive logic determines that the positioner has moved beyond track centerline limits.

WRU set causes the RP07 to turn off write current immediately and abort the write command.

WRU is a CLASS B error.

Bit 9 (WOR) - The Write Over-run error bit will set if write current is active during both the leading and trailing edges on an index pulse. The detection of this condition will turn off write current immediately and subsequently abort the write command.

Bit 10 (RWU1) - The Read/Write Unsafe 1 (RWU1) error bit is set if no write transitions are detected by the drive read/write safety circuits during a write operation (write gate ON) within five microseconds.

Bit 11 (RWU2) - The Read/Write Unsafe 2 (RWU2) error bit is set if more than one head has been selected during a read or write operation.

RWU2 immediately:

1. Deselects all heads
2. Disables write current
3. Aborts the write command.

Detection of RWU2 causes illumination of the UNSAFE indicator on the operator control panel.

RWU2 is a CLASS B error.

Bit 12 (RWU3) - The Read/Write Unsafe 3 (RWU3) bit is set when drive read/write safety circuits detect write current when no write operation is in progress (write gate OFF).

Detection of RWU3 causes illumination of the UNSAFE indicator on the operator control panel.

RWU3 is a CLASS B error.

Bit 13 (CPU) - The CPU Unsafe (CPU) bit is asserted when the 8080 microprogram fails to retrigger the CPU Unsafe timer prior to time-out.

CPU is a CLASS B error.

Bit 14 (CPE) - CROM parity error.

Bit 15 (PGE) - The Program Error (PGE) bit is set if particular commands are attempted without the CMD (Command Modifier, bit 15 in RPOF) set. The following commands then set PGE.

- Write Track Descriptor
- Read Track Descriptor
- Format Track

PGE is a CLASS B error.

Register 15_8 - Error Register 3 (RPER3)
This read-only register contains error indicators that are classified CLASS A or CLASS B as in Error Registers 1 and 2:

- CLASS A errors can be handled at the completion of a non-data transfer command or at a convenient block boundary in a data command.
- CLASS B errors are handled immediately. The drive terminates command execution as soon as possible.

Provided that an error is not hard (persistent), RPER3 is reset by any of the following.

- Power On Reset
- Drive Clear
- Massbus Initialize

Bit 0 (RTO) - RUN Timeout (RTO) is set if after 30 milliseconds from assertion of GO the drive fails to detect the RUN line assertion.

Bit 1 (SCF) - Sync Clock Failure (SCF) sets if the RP07 sync clock counter has not gone to zero within the allocated time.

Bit 2 (SBE) - Sync Byte Error (SBE) is set if the sync byte associated with a data field or defect skip is not found.

Bit 3 (DPE) - Data Parity Error (DPE) sets during a write operation under either of the following conditions.

1. If a data parity error is detected (odd parity used)
2. If a buffer parity error is detected (odd parity used)

DPE sets causes PAR (bit 3 in RPER1) to set.

DPE is a CLASS A error.

Bit 4 (SDF) - SERDES Data Failure (SDF) is set as a result of timing failures relating to the drive data buffer. SDF is asserted under either of the following conditions.

1. An attempt to shift data into the buffer when it is not ready
2. An attempt to strobe buffer output when output data is not ready

Bit 5 (DCU) - DC Unsafe (DCU) sets when RP07 detects a low dc voltage.

DCU extinguishes the DC Safe Indicator and causes the UNSAFE indicator on the operator control panel to illuminate.

DCU is a CLASS B error.

NOTE

Overvoltage protection is provided by a crowbar circuit. DCU cannot be guaranteed to set for an overvoltage condition.

Bit 6 (IXU) - The Index Unsafe (IXU) error bit is asserted if an index error is detected during a Format Track command or a Write Track Descriptor command.

IXU set reflects either of two conditions:

1. Failure to detect an index pulse during Index Window
2. Detection of an index pulse outside the Index Window.

IXU is a CLASS B error.

NOTE

An index error condition cannot be reset until a valid index pulse is detected. IXU may therefore appear hard (persistent) for two complete media revolutions.

Bit 7 (DVC) - The Device Check (DVC) error bit is the inclusive OR of all error bits in RPER2 (8-15) and bits 0-15 in RPER3.

DVC does not cause an attention condition. ATA will be set in accordance with the specific error bit that causes DVC to assert.

Bit 8 (PHF) - The 8080 Processor Handshake Failure (PHF) error bit is set if the 8080 microprocessor fails to respond to a command.

Bit 9 (LCE) - Loss of Cylinder Error (LCE) is set if positioner movement is detected outside the cylinder boundary when no positioning operation is in progress. Detection of this condition causes the drive to automatically issue a recalibrate operation.

The Error bits (bits 0-7 in RPER2) and ATA will NOT become asserted until completion of the recalibrate operation. PIP (bit 13 in RPDS) is asserted during the positioning operation.

The proper error code for LCE is reflected in the eight LEDs on the A1A7 PCA (Servo Control) as well as in bits 0-7 of RPER2.

If a command is loaded with the GO bit (bit 0 in RPCS1) set during the processing of LCE, the command is deferred until completion of recalibration. Then the command is terminated and ATA is raised. If a data command is loaded, EXC (Exception) and EBL (End of Block) will be raised.

Bit 10 (LBC) - The Loss of Bit Clock (LBC) error bit is set under the following conditions.

1. A microcoded time-out occurred in which the word counter failed to attain the value loaded into the compare register within the time-out interval. This is detected by a failure of the end of branch condition to assert by the end of the time-out interval.

2. The end of branch condition failed to negate after the compare register (BC) was loaded with a new value, indicating that either:

   a. The word counter value is greater than or equal to the new BC register value at the time the end branch condition is tested.

   b. The end branch signal failed to reset with the writing of the BC register.

Bit 11 (CLF) - Control Logic Failure (CLF) is set by RP07 logic as a result of any of the following conditions.

- An attempt to write in the 8080 Communications Register when the register is full

- An interrupt failure in the 2901

- An invalid function code interrupt

Bit 12 (WSF) - Write Current Sense Failure (WSF) is set by the drive logic when the device fails to sense Write Current after write gate has been enabled.

Bit 13 (DSE) - Defect Skip Error (DSE) is set by the drive logic when an invalid value results during defect skip calculation.

Bit 14 (SKI) - Seek Incomplete (SKI) is set when the drive logic detects any of the following conditions.

Error Codes

Error Code	Condition
0A	Seek too long
0B	Guard band detected during seek
0C	Seek overshoot
44	Guard band detect failure during recalibrate
45	Reference gap or GB pattern; detection failure (recalibrate)
46	Seek error during recalibrate
4A	Attempt to land in guard band during recalibrate

Detection of a SKI error causes the RP07 to automatically initiate a recalibrate. The error bits (0-7 in Error Register 2) and ATA will NOT set until completion of the recalibration operation.

That particular error code for SKI is reflected in the eight LEDs on A1A07 PCA as well as in bits 0-7 of RPER2.

Bit 15 (BSE) - Bad Sector Error (BSE) is set at the completion of a CRC character check if either bit 14 or bit 15 of the first header word is found to be zero.

If the command is Read or Write Check Header and Data, command termination occurs at normal EBL (End of Block) time for the current sector.

If the command is not Read or Write Check Header and Data, command termination occurs at completion of the CRC check.

Register 16_8 - ECC Position Register (RPEC1)
This read-only register contains the binary address minus 1 of the first bit of an error burst in the data and ECC field. The contents reflect the completion of a Data Transfer that results in DCK (Data Check Error, bit 15 in RPER1), without ECH (ECC Hard Error, bit 6 in RPER1).

If ECH (ECC Hard Error, bit 6 in RPER1) or ECI (Error Correction Inhibit, bit 11 in RPOF) is set, the contents of RPEC1 are irrelevant.

Bits 0-12 - These bits are binary weighted.

Bits 13-15 - Always 0.

Register 17_8 - ECC Pattern Register (RPEC2)
This read-only register contains an 11-bit error burst that is XORed (exclusive ORed) with the data in main memory (located by the position count) to correct the error burst. The contents reflect the completion of a Data Transfer that results in DCK (Data Check Error, bit 15 in RPER1) without ECH (ECC Hard Error, bit 6 in RPER1).

If ECH (bit 6 in RPER1) or ECI (Error Correction Inhibit, bit 11 in RPOF) is set, the contents of RPEC2 are irrelevant. Valid counts include the entire ECC redundancy field.

RPEC2 is reset by any of the following.

- Drive Clear
- Massbus Initialize
- Power On Reset
- Initiation of a command Function Code and GO bit (bits 0-5 in RPCS1)
- Command continuance (RUN assertion at the fall of EBL).

Bits 0-10 (PAT) - The Pattern (PAT) bits compose the 11-bit error burst field; bit 0 is LSB.

Bits 11-15 - Always 0.

MICROPROCESSOR INTERFACE REGISTER
The 2901 and 8080 MPUs interact as Master and Slave depending on mode of operation. Interaction takes place through the Communications Register on A1A08 PCA (Command/Index/Sector).

The Communications Register is composed of four 8-bit latches that supply the data path from the 2901 to the 8080 via the Y Bus (16 bits) and from the 8080 to the 2901 via the S Bus (16 bits).

CONTROL AND STATUS (RPCS1)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R/W	0	0	0	0	DVA	0	0	0	0	0	F4	F3	F2	F1	F0	GO

MR-11393

DEVICE STATUS (RPDS)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	ATA	ERR	PIP	MOL	WRL	LBT	PGM	DPR	DRY	VV=1	0	0	0	ILV	EWN	OM

MR-11394

ERROR 1 (RPER1)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	DCK	UNS	OPI	DTE	WLE	IAE	AOE	HCRC	HCE	ECH	WCF	FER	PAR	RMR	ILR	ILF

MR-11395

MAINTENANCE 1 (RPMR1)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R/W	DMD								P	A	R	A	M	E	T	E R
		DIAG 64	DIAG 32	DIAG 16	DIAG 8	DIAG 4	DIAG 2	DIAG 1								

MR-11396

ATTENTION SUMMARY PSEUDO (RPAS)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R/W	0	0	0	0	0	0	0	0	ATA 7	ATA 6	ATA 5	ATA 4	ATA 3	ATA 2	ATA 1	ATA 0

MR-11397

DESIRED TRACK/SECTOR ADDRESS (RPDA)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R/W	0	0	TA 32	TA 16	TA 8	TA 4	TA 2	TA 1	0	SA 64	SA 32	SA 16	SA 8	SA 4	SA 2	SA 1

MR-11398

DRIVE TYPE (RPDT)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	0	0	1	0	DRQ	0	0	0	0	0	1	0	0	0	DT 1	0

MR-11399

LOOK AHEAD (RPLA)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	0	0	0	SC 64	SC 32	SC 16	SC 8	SC 4	SC 2	SC 1	0	0	0	0	0	0

MR-11400

SERIAL NUMBER (RPSN)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	8K	4K	2K	1K	800	400	200	100	80	40	20	10	08	04	02	01

MR-11401

OFFSET (RPOF)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R/W	CMD	MTD	0	FMT 16	ECI	HCI	0	0	0	0	0	0	0	0	0	0

MR-11402

DESIRED CYLINDER ADDRESS (RPDC)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R/W	0	0	0	0	0	0	512	256	128	64	32	16	8	4	2	1

MR-11403

CURRENT CYLINDER ADDRESS (RPCC)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	0	0	0	0	0	0	512	256	128	64	32	16	8	4	2	1

MR-11404

ERROR 2 (RPER2)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	PGE	CPE	CPU	RWU 3	RWU 2	RWU 1	WOR	WRY UNS	E	R	R	—	C	O	D	E

MR-11405

ERROR 3 (RPER3)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	BSE	SKI	DSE	WSF	CLF	LBC	LCE	PHF	DVC	IXU	DCU	SDF	DPE	SBE	SCF	RTO

MR-11406

ECC POSITION (RPEC1)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	0	0	0	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

MR-11407

ECC PATTERN (RPEC2)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
R	0	0	0	0	0	BIT 10	BIT 9	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0

MR-11408

RP20 FSC LIST

FSC	Error Description
0000	FSC NOT GENERATED (LOAD ROUTINE 30)
0900	CHANNEL BUS OUT PARITY ERROR
0901	INVALID COMMAND
0902	INVALID SEQUENCE (CCW)
0903	CCW COUNT LESS THAN REQUIRED
0904	DATA VALUE NOT AS REQUIRED
0905	DIAG. WRITE INHIBITED BY FILE MASK
0906	CHANNEL ABORTED RETRY
0907	CHANNEL RETURNED WITH INCORRECT RETRY CCW
0908	MPL FILE NOT READY
0909	MPL FILE PERMANENT SEEK CHECK
090A	MPL FILE PERMANENT READ CHECK
090B	IMPROPER ALTERNATE TRACK POINTER
090C	PERMANENT DATA OVERRUN DETECTED
090D	INDEX DETECTED IN GAP OF RECORD
1000	FALSE DEVICE INTERFACE CHECK
1001	DEV INTF CHECK-TAG BUS PARITY CHECK
1002	DEV INTF CHECK-BUS OUT PARITY CHECK
1003	DEV INTF CHECK-TAG BUS AND BO PAR CHK.
11FF	HDA MODE PAR CHK-MULTI OR NOT FORMAT MODE
11XX	HDA SEQUENCE CHECK STATE 6
1200	TIMEOUT CK IN RECAL, ST 0-MOVE OUT
1201	TIMEOUT CK IN RECAL, ST 1-RESET
1206	TIMEOUT CK IN REZERO, ST 6-REZERO LIN MODE
1208	TIMEOUT CK IN SEEK, ST 8-DECELERATE
120A	TIMEOUT CK IN SEEK, ST A-ACCELERATE
120C	TIMEOUT CK IN SEEK, ST C-SEEK LIN MODE
120E	INV. T.O. CK IN SEEK, ST E-ON TRACK
1210	TIMEOUT CK IN REZERO, ST 10-MOVE OUT
1212	TIMEOUT CK IN REZERO, ST 12-TURN AROUND
1216	TIMEOUT CK IN REZERO, ST 16-MOVE IN
12XX	TIMEOUT CK IN AN INVALID CONTROL STATE
1301	SECTOR NON COMPR WITHIN TWO INDEX MARKS
1310	FALSE DRIVE CHECK
1400	FALSE R/W CHECK
1401	WRITE I CHECK
1402	TRANSITIONS CHECK
1404	CONTROL CHECK
1408	DELTA I/W CHECK
1410	INDEX CHECK SEL
1420	WRT OVERRUN LATCH
1440	CAPABLE ENABLE CHECK
1480	MULTICHIP CHECK
14F4	PAD GATE ERROR 5
14F8	HEAD SHORT LATCH
14XX	MULTIPLE R/W CHECKS
1500	OVERSHOOT CHECK DURING REZERO
1506	RECALIBRATE TRACK 0 OVERSHOOT CHECK
1508	OVERSHOOT CK IN SEEK, ST 8-DECELERATE
150A	OVERSHOOT CK IN SEEK, ST A-ACCELERATE
150C	OVERSHOOT CK IN SEEK, ST C-LIN MODE
150E	OVERSHOOT CK, LOST SERVO TRACK FOLLOWING
1510	OVERSHOOT CK DURING REZERO
1512	OVERSHOOT CK DURING REZERO
1516	OVERSHOOT CK DURING REZERO
15XX	OVERSHOOT CK IN AN INVALID STATE
160E	SERVO OFF TRACK ERR DURING ON TRACK STATE
16XX	SERVO OFF TRACK ERR DURING AN INV CTRL STATE
	OR SET R/W ACTIVE DURING ACCESS MOTION
1910	ERROR ALERT
1911	TRANSMIT TARGET ERROR
1912	MICROPROGRAM DETECTED ERROR (SENSE BYTE 18)
1913	DIFF COUNTER OR HAR FAILED TO REST ON A REZERO
1914	SYNC OUT TIMING CHECK
1915	UNEXPECTED FILE STATUS AT INTL SELECTION
1916	TRANSMIT CAR ERROR
1917	TRANSMIT HAR ERROR
1918	TRANSMIT DIFF COUNTER ERROR
1919	UNEXPECTED FILE STATUS IN READ IPL
191A	SEEK VERIFICATION CHECK
191B	SECTOR COMPARE CHK IF BYTE 9, BIT 1 ON
	TIMEOUT CHECK IF BYTE 16, BIT 0 ON
	OVERSHOOT CHECK IF BYTE 16, BIT 1 ON
191C	NO INTERRUPT FROM DRIVE (MISSING ATTENTION)
191D	DEFECT SKIPPING REORIENTATION ERROR
191E	UNABLE TO DETERMINE DEVICE FORMAT MODE
191F	RETRY REORIENTATION CHECK
2100	FILE INTERFACE CHECK

RP20 FSC LIST (Cont)

FSC	Error Description
2102	FILE INTERFACE TRANSFER CHECK
2104	FILE INTERFACE BUS OUT CHECK
2108	FILE INTERFACE TAG BUS PARITY CHECK
2110	FILE INTERFACE UNEXPECTED END CHECK
2120	FILE INTERFACE BUFFER PARITY CHECK
2140	FILE INTERFACE SELECT ACTIVE CHECK
2180	FILE INTERFACE LOGIC CHECK
2186	FILE INTERFACE LOGIC AND TRANSFER CHK
2188	FILE INTERFACE TAG BUS PARITY CHECK
21XX	IF BIT 0, 1, 2 OR 3 OF BYTE 20 IS ON, SUSPECT DEVICE
2202	COMPARE ASSIST CHECK
2204	LOAD S REG CHECK
2210	DATA TRANSFER CHECK
2220	INTERFACE CHECK CHANNEL B
2221	INTERFACE CHECK CHANNEL D
2240	INTERFACE CHECK CHANNEL A
2241	INTERFACE CHECK CHANNEL C
2280	CHANNEL BUFFER PARITY CHECK
2290	DATA TRANSFER CHECK
2920	CHECK 2 WITH NO BIT IN BYTE 11 OR 20
2923	S REGISTER LOAD ERROR
2924	CHECK 2-CI REGISTERS ARE VALID A CHECK 2 DETECTED IN A SEL SEQ WITH NO BITS IN BYTE 11 OR 20 REG TO 2920
3XXX	A FAILURE IN THE CHECK 1 REG CAN CAUSE ANY FORMAT 3 SYMPTOM
31XX	ERROR DETECTED IN CU CLOCK
3204	ERROR DETECTED IN CD DECODE CIRCUITRY
3220	ECC LOGIC FAILURE
3240	DOUBLE BIT ERROR
3260	ECC LOGIC AND DOUBLE BIT ERROR
3381	ERROR DETECTED IN SPEC. OP DECODE CIRC.
3382	ERROR DETECTED IN STATUS REG. OP CH/CL BRANCH CIRCUITRY
3402	MPL FILE NOT READY
3410	ERROR DETECTED IN CTRL STOR WRITE BUS 1/3
3420	ERROR DETECTED ON CTRL STOR WRITE BUS 0/2
3430	ERROR DET. ON CTRL STOR WRITE BUS 1/3 AND 0/2
3440	STORAGE ADR BUS 8-15 CHECK (IF BYTE 10, BIT 5 ON
3448	STORAGE ADR BUS 8-15 CHECK REF TO 3504)
3480	STORAGE ADR BUS 0-7 CHECK
3488	STORAGE ADR BUS 0-7 CHECK
34C0	STORAGE ADR BUS 0-7 AND 8-15 CHECK
34C8	STORAGE ADR BUS 0-7 AND 8-15 CHECK
3501	MPL READ CHECK, MPL PAR ERROR DETECTED
3502	ALU CHECK
3504	B REGISTER CHECK
3506	B REGISTER AND ALU CHECK
3508	A REGISTER CHECK
350A	A REGISTER AND ALU CHECK
3530	CHECK 1 ERROR BUT NO BITS ON IN BYTES 10 + 11
3930	CHECK 1 ERROR BUT NO BITS ON IN BYTES 10 + 11
4940	ECC DATA CHECK HA FIELD
4941	ECC DATA CHECK COUNT FIELD
4942	ECC DATA CHECK KEY FIELD
4943	ECC DATA CHECK DATA FIELD
4944	NO SYNC BYTE FOUND HA FIELD
4945	NO SYNC BYTE FOUND COUNT FIELD
4946	NO SYNC BYTE FOUND KEY FIELD
4947	NO SYNC BYTE FOUND DATA FIELD
4949	NO AM FOUND DURING RETRY
9001	MISSING TAG VALID ON R/W OPERATION
9002	NORMAL OR CHECK END MISSING FOLLOWING R/W OR ECC OPERATION
9003	NO RESPONSE FROM A CTRL MODULE ON A CONTROL OPERATION
9004	TIMEOUT WAITING FOR INDEX
9005	ECC HARDWARE CHECK
9006	MULTIPLE CONTROLLERS SELECTED
9007	PRESELECTION CHECK
9008	REPETITIVE CMD OVERRUNS ON G1 OPS.
9009	REP. CMD OVERRUNS ON G2 OR G3 OPS.
900A	PHYSICAL ADDRESS CHECK (WRONG ADR. RET.)
900B	BUSY MISSING AFTER SEEK START IS ISSUED
900E	DEVICE INTERFACE FAILURE
900F	ATTENTION CHECK (DEV ATTN FAILED TO RESET)
9101	REORIENT COUNTER CHECK
9102	TRACK COUNTER CHECK
9104	WRITE FAIL
9108	CONTROLLER BUS IN PARITY CHECK

RP20 FSC LIST (Cont)

FSC	Error Description
9110	DEVICE BUS IN PARITY CHECK
9118	DEV AND CONTR BUS IN PAR CHECK
9120	CHECK 1 OF 8
9140	BUS OUT PARITY CHECK
9180	TAG BUS PARITY CHECK
91FF	CONTR INTERFACE BUS IN ASSEMBLY FAILURE
91XX	SOME FAILURES CAUSE MULTIPLE FSC's
9200	FALSE CONTROLLER CHECK
9201	ECC 0 COMPARE (NORMAL COMPL OF R/W)
9202	ECC HARDWARE CHECK
9204	STATUS MONITOR CHECK
9208	WRITE DATA PARITY ERROR
9210	GAP COUNTER CHECK
9220	SHIFT REGISTER ERROR
9240	MISSING SERVO DATA
9280	VFO PHASE ERROR
92C0	MISSING READ DATA
92XX	SOME FAILURES CAUSE MULTIPLE FSC's
93XX	INVALID FAULT SYMPTOM CODE

MODULE ADDRESS JUMPERS RP20

CJ03 OR CJ04	
ON	1
C	2
OFF	3
ON	4
C	5
OFF	6
ON	7
C	8
OFF	9

RP20 COMMANDS

Command		MT OFF*	MT ON*	Count
Control	Orient(c)	28		Nonzero
	Recalibrate	13		Nonzero
	Seek	07		6
	Seek Cylinder	0B		6
	Seek Head	1B		6
	Space Count	0F		3(a); nonzero (d)
	Set File Mask	1F		1
	Set Sector (a,f)	23		1
	Restore (executes as a no-op)	17		Nonzero
	Vary Sensing(c)	27		1
	Diagnostic Load (a)	53		1
	Diagnostic Write (a)	73		512
Search	Home Address Equal	39	B9	4
	Identifier Equal	31	B1	5
	Identifier High	51	D1	5
	Identifier Equal or High	71	F1	5
	Key Equal	29	A9	KL
	Key High	49	C9	KL
	Key Equal or High	69	E9	KL
	Key and Data Equal (d)	2D	AD	Number of bytes (including mask bytes) in search argument
	Key and Data High (d)	4D	CD	
	Key and Data Equal or Hi (d)	6D	ED	
Continue Scan	Search Equal (d)	25	A5	
	Search High (d)	45	C5	
	Search High or Equal (d)	65	E5	
	Set Compare (d)	35	B5	
	Set Compare (d)	75	F5	
	No Compare (d)	55	D5	
Read	Home Address	1A	9A	5
	Count	12	92	8
	Record 0	16	96	Number of bytes to be transferred
	Data	06	86	
	Key and Data	0E	8E	
	Count, Key and Data	1E	9E	
	IPL	02		
	Multiple Count, Key, Data (b)	5E		> Max. track len.
	Sector (a,f)	22		1
Sense	Sense I/O	04		24 (a), 6 (d)
	Sense I/O Type (b)	E4		7
	Read, Reset Buffered Log (b)	A4		24
	Read Buffered Log (c)	24		128
	Device Release (e)	94		24 (a); 6 (d)
	Device Reserve (e)	B4		24 (a); 6 (d)
	Read Diagnostic Status 1 (a)	44		16 or 512
Write	Home Address	19		5, 7, or 11
	Record 0	15		8+KL+DL of R0
	Erase	11		8+KL+DL
	Count, Key and Data	1D		8+KL+DL
	Special Count, Key and Data	01		8+KL+DL
	Data	05		DL
	Key and Data	0D		KL+DL

* Code same as MT Off except as listed.
a. Except 2314, 2319
b. 3330-3340-3350 series only.
c. 2305/2835 only.
d. 2314, 2319 only.
e. String switch or 2-channel switch required.
f. Special feature required on 3340.

NONLINKED ROUTINES

Routine Number Hex	
D8	SERVO ADJUSTMENT
D9	INCREMENTAL SEEK
DA	CYL-CYL SEEK
DB	RANDOM SEEK
DC	PUMPED RESONANCE
DD	CRASH STOP
DE	SERVO MARGIN
E0	SYNC UTILITY
E1	READ UTILITY
E2	DISPLAY HA
E5	DISPLAY DRV CONFIG/SN
E6	DISPLAY MEMORY
E7	MEMORY SCAN
E8	FE PANEL
E9	HDA STATE ANALYSIS
EA	DISPLAY SENSE DATA
EB	TAG UTILITY
EC	STRING SW/DUAL PORT (FEATURE)
EE	MANUAL INTERVENTION
F2	TRACK ANALYSIS

LINKED SERIES ROUTINES

Routine Number Hex		
C1	CONTROL INTERFACE	
C2	DRIVE INTERFACE	
C3	BASIC SERVO	Drive must be ready
C4	INDEX AND SECTOR	Drive must be ready
C5	GAP COUNTER	Drive must be ready
C6	BASIC READ-WRITE	Drive must be ready
C7	PADDING	Drive must be ready
C8	ECC LOGIC	Drive must be ready
C9	REORIENT CTR/TR CTR	Drive must be ready
CA	COMPLEX SERVO	Drive must be ready
CB	R/W RELIABILITY	Drive must be ready
CC	R/W MARGIN	Drive must be ready
CD	AM DETECTION	Drive must be ready
CE	OVERWRITE	Drive must be ready
CF	REFORMAT FE TRACKS	Drive must be ready

PROGRAM CONTROL DATA DISPLAYS

Program Control		Data
82	ROUTINE LOADING	RTN NO.
8C	ROUTINE RUNNING	RTN NO.
8D	DYNAMIC ERROR DISPLAY REPEAT ERROR TEST AFTER ERROR	ERROR NO.
CO	INVALID ROUTINE OR SYSTEM RESET	RFTN NO. 00
CA	ROUTINE READY FOR EXECUTION	RTN NO.
CE	MANUAL INTERVENTION REQUIRED OR DISPLAY COMPLETE	RTN NO.
CF	NORMAL END	RTN NO.
DX	PARAMETER ENTRY REQUIRED	RTN NO.
E1	ERROR/MESSAGE STOP	ERROR/MSG NO.
EX	ERROR/MESSAGE BYTE BITS 4-7: BYTE NUMBER BEING DISPLAYED	BYTE

CONTROL OPTIONS

Hex	Entry
CX	DRIVE SELECTION
00	START/STOP ROUTINE EXECUTION
10	PARAMETER ENTRY
20	START OR ADVANCE ERROR/MESSAGE DISPLAYS
30	RESET DIAGNOSTIC CONTROL
	8000 MOD-II AND ISC-RESTORE FAULT
	SYSTEM CODE GENERATOR IN OVERLAY AREA.
	MOUNT FUNCTIONAL FLOPPY IF ISC.

COMMON ERROR STOPS

Program Control Display: E1

Data Display:

01	INVALID PARAMETER ENTRY
02	DRIVE NOT ONLINE
03	NO PHYSICAL ADDRESS FROM FE DRIVE. CHECK SWITCHES
04	NOT WRITE ENABLED
05	MULTIPLE DRIVE SELECT CHECK FE MODE SWITCHES
06	INCORRECT DRIVE TYPE
07	HDA SEQUENCE ERROR, CHECK FOR DRIVE READY OR RUN ROUTINE E9
08	NO TAG VALID

RUN OPTIONS

Hex	Entry
01	DYNAMIC ERROR DISPLAY REPEAT TEST AFTER ERROR
02	LOOP ROUTINE(S)
03	DYNAMIC ERROR DISPLAY/LOOP ROUTINE(S)
04	INHIBIT LINKING
05	INHIBIT LINKING/DYNAMIC ERROR DISPLAY
06	LOOP SINGLE ROUTINE
07	LOOP SINGLE ROUTINE/DYNAMIC ERROR DISPLAY
08	RESET OPTION

BIT SIGNIFICANCE OF RUN OPTION ENTRY:

```
DYNAMIC ERROR DISPLAY
REPEAT TEST AFTER ERROR
LOOP ROUTINE(S)
INHIBIT LINKING
RESET OPTION

0  1  2  3  4  5  6  7

0  0  0  0  X  X  X  X
```

INFORMATION COMMON TO ALL FORMATS

BYTE \ BIT	0	1	2	3	4	5	6	7
0	COMMAND REJECT	INTERVENTION REQUIRED	BUS OUT PARITY	EQUIPMENT CHECK	DATA CHECK	OVERRUN	NOT USED	NOT USED
1	PERMANENT ERROR	INVALID TRACK FORMAT	END OF CYLINDER	NOT USED	NO RECORD FOUND	FILE PROTECT	WRITE INHIBIT	OPERATION INCOMPLETE
2	NOT USED	CORRECTABLE	NOT USED	ENVIRONMENTAL DATA BYTE 8-23	EMULATION	NOT USED	NOT USED	NOT USED
3	RESTART COMMAND BYTE 1 BIT 7 = 0 - LAST CHANNEL COMMAND IN THE CCW BYTE 1 BIT 7 = 1 - OPERATION IN PROGRESS WHEN OBR WAS GENERATED							
4	PHYSICAL DRIVE IDENTIFICATION							
	DRIVE 8 OR 0	DRIVE 9 OR 1	DRIVE A OR 2	DRIVE B OR 3	DRIVE C OR 4	DRIVE D OR 5	DRIVE E OR 6	DRIVE F OR 7
5	LOGICAL CYLINDER ADDRESS LOW							
	128	64	32	16	8	4	2	1
6	CYL ADD HIGH 1024	CYL ADD HIGH NAT/-11 512	CYL ADD HIGH NAT/-11 256	LOGICAL TRACK (HEAD)				
				16	8	4	2	1

MR-6083

FORMAT 0 MESSAGE ONLY SENSE BYTES 8-23 NOT USED SENSE BYTE 7 FORMAT/MESSAGE

	0	1	2	3	4	5	6	7
MESSAGE	NO MESSAGE	INVALID COMMAND	INVALID SEQUENCE	CCW COUNT LOW	DATA ARGUMENT INVALID	DIAG/W INHIBITED BY FILE MASK	CHANNEL ABORTED RETRY	CHANNEL CCW 1 INCORRECT ON RETRY
	8	9	A	B	C	D	E	F
MESSAGE	MPL FILE NOT READY	MPL FILE PERMANENT SEEK CHECK	MPL FILE PERMANENT READ CHECK	COMMAND OVERRUN	DATA OVERRUN	DEFECTIVE TRACK	ALTERNATE TRACK	NOT USED

MR-6084

FORMAT 1 DRIVE EQUIPMENT CHECK SENSE BYTE 7 FORMAT/MESSAGE

	0	1	2	3	4	5	6	7
MESSAGE	NOT USED	TRANSMIT TARGET ERROR	MICRO-PROGRAM DETECTED ERROR	TRANSMIT DIFFERENCE HIGH ERR	SYNC OUT TIMING ERROR	UNEXPECTED DRIVE STATUS AT INITIAL SELECTION	TRANSMIT CYL ADDR REGISTER ERR	TRANSMIT HEAD ERROR
	8	9	A	B	C	D	E	F
MESSAGE	TRANSMIT DIFFERENCE ERR	DRIVE STAT NOT AS EXPECTED DURING RD IPL	SEEK VER CHECK ON PHYSICAL ADDRESS	SEEK INCOMPLETE OR SECTOR COMPARE CHECK	NO INTR FROM DRIVE	DEFECT SKIPPING OR REORIENTATION CHECK	NOT USED	RETRY REORIENTATION CHECK

MR-6085

FORMAT 1 SENSE BYTES 8-13

<table>
<tr><th>BYTE \ BIT</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>DRIVE STATUS BYTE 19 BIT 0=0
8</td><td rowspan="2">CONTROLLER CHECK (REF. BYTE 17&20) CC170</td><td>TAG BUS OR BUS OUT PAR DG050</td><td rowspan="2">ACCESS CHECK SECTOR NONCOMPARE DP050</td><td rowspan="2">RD/WR CHECK (REF BYTE 12&19) DE005</td><td rowspan="2">ONLINE DF030</td><td>ACCESS HDA ATTN DF030</td><td>BUSY DF040</td><td>SK CMPT SK SCTR PAD CMPT DF070</td></tr>
<tr><td>BYTE 19 BIT 0=1 WRITE OR READ</td><td>WRITE 1 DD010</td><td>PAD IN PROGRESS DE030</td><td>INDEX MARK DE015</td><td>3330 MODES DH005</td></tr>
<tr><td>9
CHECK STATUS</td><td>PAD STATUS DE030</td><td>SECTOR NON COMPARE DH060</td><td>MOTOR AT SPEED LTH DH010</td><td>AIR SWITCH ON LTH DH010</td><td>WRITE ENABLE DH010</td><td>FIXED HEADS DH005</td><td colspan="2">3330-1 10
8350 . . . . 01*
3330-11 . . . . 11*
DH005</td></tr>
<tr><td>10
HDA/SEQ CONTROL</td><td>FMT ERR/ FMT LTH ERROR</td><td colspan="3">HDA SEQUENCE STATE LATCH
4 | 2 | 1</td><td>HDA SEQUENCE CHECK LTH</td><td>INHIBIT HDA RECYCLE</td><td>GEMINI HDA</td><td>ODD TRACK STAT 7</td></tr>
<tr><td>11
LOAD SW STATUS</td><td>DRIVE START LTH DH010</td><td>GUARD BAND LTH DA005</td><td>TARGET VELOCITY DB030</td><td>TRACK CROSSING DA025</td><td>NOT USED</td><td>AIR SWITCH ON DH085</td><td>GEMINI HDA DOUBLE DENSITY</td><td>MOTOR AT SPEED DH010</td></tr>
<tr><td>12</td><td>MULTICHIP CHECK DE010</td><td>CAPABLE ENABLE CHECK DE010</td><td>WRITE OVERRUN LTCH DE015</td><td>INDEX CHECK SEL DA005</td><td>DELTA I/W CHECK ** DE025</td><td>CONTROL CHECK DE010</td><td>TRANSITIONS CHECK DE035</td><td>WRITE 1 CHECK DE035</td></tr>
<tr><td>13
MESSAGE CODE 2 AND C</td><td colspan="8">BUS OUT AT TIME OF ERROR WHEN SENSE BYTE 18=01,03,05,06 OR 0C</td></tr>
<tr><td rowspan="2">13
MESSAGE CODE A OR B</td><td colspan="8">LOGICAL CYL ADDRESS LOW PRIOR TO SENSE BYTE 5</td></tr>
<tr><td>128</td><td>64</td><td>32</td><td>16</td><td>8</td><td>4</td><td>2</td><td>1</td></tr>
</table>

MR-6091

FORMAT 1 SENSE BYTES 13-18

BYTE \ BIT	0	1	2	3	4	5	6	7
13 MESSAGE CODE 1,3,5,6, 7,8,&9	EXPECTED DRIVE STATUS/DATA							
14 MESSAGE OTHER THAN A&B	CONTROL INTERFACE BUS IN AT TIME OF FAILURE							
14 MESSAGE CODE A&B	IF BIT 0&1 = 11 FIXED HEAD LOGICAL CYL ADDR HIGH PRIOR TO BYTE 6		LOGICAL TRACK PRIOR TO SENSE BYTE 6 TRACK = HEAD					
	512	256	32	16	8	4	2	1
15	CONTROL INTERFACE TAG BUS AT THE TIME OF THE DETECTED ERROR							
16	TIME OUT CHECK DF040	OVER SHOOT CHECK DF030	SERVO OFF-TRACK DF010	REZERO MODE LATCH DF050	SERVO LATCH DF010	LINEAR MODE LATCH DF010	CONTROL LATCH DF010	WAIT LATCH DF040
17	VFO PHASE CK 01 = MISSING SERVO DATA 10 = VFO PHASE ERR 11 = MISSING READ DATA CC140		SHIFT REG ERR CC140	GAP CNTR CHECK CF120	WRT DATA PARITY ERROR CC140	STATUS MONITOR CHECK CJ160	ECC HARDWARE CHECK CD100	ECC O COMPARE CD100
18	NOT USED				CODED ERROR CONDITION (BITS 4-7 HEX) LISTED BELOW			

BYTES 18, BITS 4-7

0	1 !	2	3 !	4	5 !	6 !	7
NOT USED	NO TAG VALID ON R/W OPERATION	NO NORMAL OR CHECK END ON R/W OR ECC OPERATION	NO RESPONSE FROM CNT MOD ON CNT OPERATION	TIME OUT WAITING FOR INDEX	ECC HARDWARE CHECK	MULTIPLE OR NO CNT MOD SELECTED	PRESELECTION CHECKS
8	**9**	**A**	**B**	**C !**	**D**	**E**	**F**
REPETITIVE CMND OVERRUNS ON G1 OPERATIONS	REPETITIVE CMND OVERRUNS ON G2 OR G3 OPERATIONS	POLL OR 1 OF 8 DECODE ERROR	BUSY MISSING AFTER SEEK START IS ISSUED	DEVICE TYPE ERROR	CHANNEL SELECT ERROR	PRESELECTION DISK CONTROL INTERFACE BUS	UNRESETABLE INTERRUPT

NOTE:
! SENSE BYTES 13,14,15 ARE VALID FOR THESE MESSAGES

MR-6092

FORMAT 1 SENSE BYTES 19-23

BYTE \ BIT	0	1	2	3	4	5	6	7
19	SET R/W OPERATION 85 CH100	NOT USED	NOT USED	NOT USED	HEAD SHORT LATCH DE015	PAD GATE ERROR 5 DE110	1,2 MB FILE	ALWAYS ON
20	TAG BUS PARITY CHECK CH120	BUS OUT PARITY CHECK LATCHED CH120	CHECK 1 OF 8 CJ150	DEVICE BUS IN PARITY CHECK LATCHED CJ150	CONTROLLER BUS IN PARITY CHK LATCHED CD180	CURRENT (I) WRITE CHECK CC170	TRACK COUNTER CHECK GTD CC170	REORIENT COUNTER CH150
20 MESSAGE A AND BYTE 0 BIT 3 = 1	128	64	LOGICAL CYLINDER ADDRESS LOW 32	16	8	4	2	1
21 MESSAGE A AND BYTE 0 BIT 3 = 1	BITS 0 & 1 = 11 FIXED HEAD LOGICAL CYLINDER ADDRESS HIGH 512	256	LOGICAL TRACK (HEAD) 32	16	8	4	2	1
21	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED	BUS OUT PARITY (BO PAR)	TAG BUS PARITY
22 - 23	FAULT SYMPTOM CODE							

MR-6093

FORMAT 2 DCU ERROR SENSE BYTE 7 FORMAT/MESSAGE

	0	1	2	3	4	
MESSAGE	NO MESSAGE	NOT USED	NOT USED	S REG LOAD CHECK	CTL INTF REG VALID SENSE BYTES 13-15	MESSAGE 5-F NOT USED

MR-6086

FORMAT 2 DCU CHECK SENSE BYTES 8-23

BYTE \ BIT	0	1	2	3	4	5	6	7
8-10	NOT USED							
11 CONTROL CHECK	CHANNEL BUFFER PARITY CHECK	INTERFACE CHECK CHANNEL A OR C	INTERFACE CHECK CHANNEL B OR D	DATA TRANSFER CHECK	CONTROL INTERFACE LOGIC CHECK	LOAD 5 REGISTERS CHECK	COMPARE ASSIST CHECK	CHANNEL C/D OR MULTI-CONNECT
12	SET TO 0							
13	CONTROL INTERFACE (CONTENTS OF TA REGISTERS, VALID ONLY IF SENSE BYTE 7 IS 24)							
14	CONTROL INTERFACE BUS-IN (CONTENTS OF MA REGISTER, VALID ONLY IF SENSE BYTE 7 IS 24)							
15	CONTROL INTERFACE BUS-IN (CONTENTS OF MD REGISTER, VALID ONLY IF SENSE BYTE 7 IS 24)							
16-19	NOT USED, SET TO 0							
20	CONTROL MODULE CHECK	CONTROL MODULE ACTIVE CHECK	CONTROL MODULE BUFFER PARITY CHECK	CONTROL MODULE UNEXPECTED END CHECK	CONTROL MODULE TAG BUS CHECK	CONTROL MODULE BUS-OUT CHECK	CONTROL MODULE TRANSFER CHECK	NOT USED
21	NOT USED, SET TO 0							
22-23	FAULT SYMPTOM CODE							

MR-6094

FORMAT 3 SENSE BYTE 7 FORMAT/MESSAGE

MESSAGE	FORMAT 3 SENSE BYTE 7/MESSAGE NOT USED SELECTIVE RESET

MR-6087

FORMAT 3 SENSE BYTES 8-23

BYTE \ BIT	0	1	2	3	4	5	6	7
8	FAILING ADDRESS (BACK-UP ADDRESS REGISTER BUS 0-7)							
9	FAILING ADDRESS (BACK-UP ADDRESS REGISTER BUS 8-13)							
10	BIT 0 = 1 EARLY ERROR	CLOCK ERROR	0	0	0	0	0	SPECIAL OP ERROR
	BIT 0 = 0 LATE ERROR				A REG CHECK	B REG CHECK	ALU CHECK	MPL READ CHECK
11 SENSE BYTE 10 BIT 0 = 1	0	STORE MULTIPLE READ ERROR	STORE ECC LOGIC ERROR	0	0	CD DECODE ERROR	0	0
11 SENSE BYTE 10 BIT 0 = 0	STORE ADDRESS BUS 0-7 CHECK	STORE ADDRESS BUS 8-13 CHECK	STORE WRITE BUS 2/3 CHECK	STORE WRITE BUS 0/1 CHECK	0	0	MPL NOT READY	0
12	SYNDROME REGISTER							
13	TC REGISTER (THIS REGISTER IS RESET IF SELECTIVE RESET OCCURRED IN RESPONSE TO DISCONNECT IN)							
14	TG REGESTER (THIS REGISTER IS RESET IF SELECTIVE RESET OCCURRED IN RESPONSE TO DISCONNECT IN)							
15-23	NOT USED. SET TO 0							

MR-6095

FORMAT 4 DATA CHECKS UNCORRECTABLE SENSE BYTE 7 FORMAT/MESSAGE

	0	1	2	3	4	5	6	7
MESSAGE	HA ECC DATA CHECK	COUNT FIELD DATA CHECK	KEY FIELD DATA CHECK	DATA FIELD DATA CHECK	HA FIELD NO SYNC BYTE FOUND	COUNT FIELD NO SYNC BYTE FOUND	KEY FIELD NO SYNC BYTE FOUND	DATA FIELD NO SYNC BYTE FOUND
	8	9						
MESSAGE	NOT USED	AM DETECTION FAILURE ON RETRY	A-F NOT USED					

MR-6088

FORMAT 4 DATA CHECKS NOT PROVIDING DISPLACEMENT INFORMATION SENSE BYTES 8-23

BYTE \ BIT	0	1	2	3	4	5	6	7
8	CYLINDER ADDRESS OF THE RECORD IN ERROR							
	0	0	0	0	0	0	512	256
9	CYLINDER ADDRESS OF THE RECORD IN ERROR							
	128	64	32	10	8	4	2	1
10	HEAD ADDRESS OF THE RECORD IN ERROR							
	0	0	0	0	0	0	0	0
11	HEAD ADDRESS OF THE RECORD IN ERROR							
	0	0	0	16	8	4	2	1
12	RECORD NUMBER (UNRELIABLE MESSAGE 0 OR 4, ERROR HA) (UNRELIABLE MESSAGE 1 OR 5, ERROR COUNT FIELD)							
13	SECTOR NUMBER OF THE RECORD IN ERROR							
	128	64	32	16	8	4	2	1
14-21	NOTE BYTE 15 = RETRY COUNT; NOT USED							
22-23	FAULT SYMPTOM CODE							

MR-6096

FORMAT 5 DATA CHECKS CORRECTABLE FORMAT/MESSAGE

	0	1	2	3	
MESSAGE	NOT USED	NOT USED	NOT USED	DATA FIELD CORRECTABLE DATA CHECK	MESSAGES 4 - F NOT USED

MR-6089

FORMAT 5 DATA CHECKS PROVIDING DISPLACEMENT INFOMATION SENSE BYTES 8-23

BYTE \ BIT	0	1	2	3	4	5	6	7
8	CYLINDER ADDRESS OF THE RECORD IN ERROR							
							512	256
9	CYLINDER ADDRESS OF THE RECORD IN ERROR							
	128	64	32	16	8	4	2	1
10	HEAD ADDRESS OF THE RECORD IN ERROR							
11	HEAD ADDRESS OF THE RECORD IN ERROR							
	0	0	0	0	0	0	0	0
12	RECORD NUMBER SET TO 0 IF ERROR OCCURRED IN HA							
				16	8	4	2	1
13	SECTOR NUMBER OF THE RECORD IN ERROR							
14	NOT USED							
15,16,17	IDENTIFIES THE NUMBER OF BYTES PROCESSED BY THE DCU FROM THE INITIATION OF DATA TRANSFER AND THE END OF THE DATA FIELD							
18-19	ERROR DISPLACEMENT, SPECIFIES THE FIRST BYTE IN ERROR WITHIN THE DATA FIELD WITH RELATIONSHIP TO THE END OF THAT DATA FIELD							
20-21	ERROR CORRECTION PATTERN (EACH BIT IN ERROR WILL BE INDICATED BY A 1)							
22	ALWAYS 0							
23	NOT USED							

MR-6097

FORMAT 6 USAGE AND OVERRUN ERROR STATISTICS SENSE BYTES 8-23

BYTE \ BIT	0	1	2	3	4	5	6	7
8-11	NUMBER OF BYTES READ OR SEARCHED (KEY AND DATA FIELD ONLY)							
12-13	NOT USED SET TO ZERO							
14-15	NUMBER OF DATA CHECKS SUCCESSFULLY RETRIED							
16-17	NUMBER OF ACCESS MOTIONS							
18	CHANNEL SELECT FOR SENSE BYTES 20-23. BIT 0=0 INFORMATION APPLIES TO INTERFACES A AND B. IF BIT 0=1 INFORMATION APPLIES TO INTERFACES C & D. BITS 1-7 NOT USED							
19	TOTAL SEEK ERRORS RETRIED							
20	COMMAND OVERRUNS A (C)							
21	DATA OVERRUNS A (C)							
22	COMMAND OVERRUNS B (D)							
23	DATA OVERRUNS B (D)							

MR-6098

FORMAT 6 USAGE AND OVERRUN ERROR STATISTICS

MESSAGE	FORMAT 6 - MESSAGE 0-F ARE NOT USED

MR-6090

Table of Contents

GENERAL DESCRIPTION

Introduction

This section provides all the information required by an experienced Field Service engineer to use the S/X Bus Recorder to diagnose KL10 SBus errors. Included are instructions for connecting the recorder into a system, explanations of the functions of all controls and indicators, and error interpretation.

Physical Description

The S/X Bus Recorder is contained in a portable, suitcase-like aluminum housing. All controls and indicators are located on a console panel. The SBus input and output connectors, and the power connector are stored in a compartment at the back of the case. The S/X Bus Recorder is available in two models:

1. Part number 9307042-00; 117 Vac, 60 Hz
2. Part number 9307042-01; 234 Vac, 50 Hz

Console Description - All controls and indicators for the S/X Bus Recorder are located on the console panel. These controls and indicators consist of a function select switch, register and RAM address select thumbwheel switches, a 6-digit octal LED readout, a 2-digit octal LED readout, and a series of toggle switches for various control functions.

Logic Modules - All logic modules in the S/X Bus Recorder are plugged into a 9-slot hexadecimal backplane. The board complement consists of five double height modules and five hexadecimal modules in eight slots. One slot is unused. The module utilization is as follows (see Figure 1).

Slot	Module	Function
IA/F1	G5348 Hex	Recorder logic
ICD2	G5349 Double	Console
IA/F3	G5347 Hex	Translator
IA/F4	G5347 Hex	Translator
ICD5	M9006 Double	Cable connector
IEF5	M9006 Double	Cable connector
IA/F6	M8572 Hex	Cable connector
ICD7	M9005 Double	Terminator or cable connector
IEF7	M9005 Double	Terminator or cable connector
IA/F8	M8572 Hex	Terminator

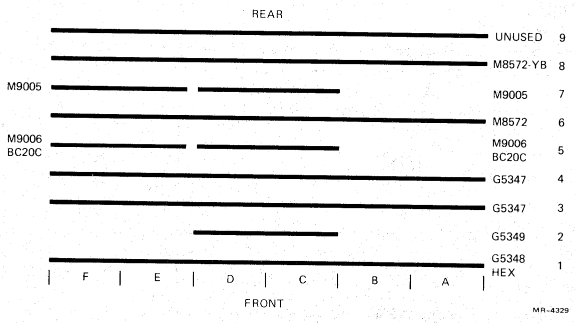


Figure 1 Backplane Layout (Top View)

Power Supply - DC operating voltages for the S/X Bus Recorder are provided by an L-H research model TM-34 power supply. The power supply outputs are as follows.

1. -5.2 Vdc @ 13 A
2. -2.0 Vdc @ 8 A
3. 5 Vdc @ 5.5 A
4. 5 Vdc @ 1.5 A

The power supply is contained on a subassembly that is 69.9 cm X 15.24 cm X 33 cm, 2.04 kg (2.75 in. X 6.0 in. X 13.0 in. and weighs 4.5 pounds).

The power supply is adjusted during its manufacture and should not normally require readjustment. If the voltages must be readjusted, proceed as follows.

1. Remove all screws from the bottom of the suitcase and remove eight screws from the recorder console top.

2. Rotate the two console panel locking screws 90 degrees and lift panel upward. Disconnect the Mate-N-Lok plug connecting the two cooling fans to ac power.

3. Remove the recorder assembly from the suitcase to gain access to the backplane for voltage measurements.

4. Verify that the following modules are present in the backplane to provide a load: 2-G5347, G5348, G5349.

5. Plug recorder into ac receptacle or set power switch to ON.

6. Connect a digital voltmeter to the test points listed below and adjust the appropriate potentiometer for the correct voltage. Access to the potentiometer is gained through a rectangular hole in the recorder housing. Refer to Figure 2.

Voltage	Test Point	Potentiometer
-5.20	C03B2	V1
-2.00	C03B1	V2
+5.00	C03A2	V3
+5.00	+5 V TAB (IND.BD)	V4

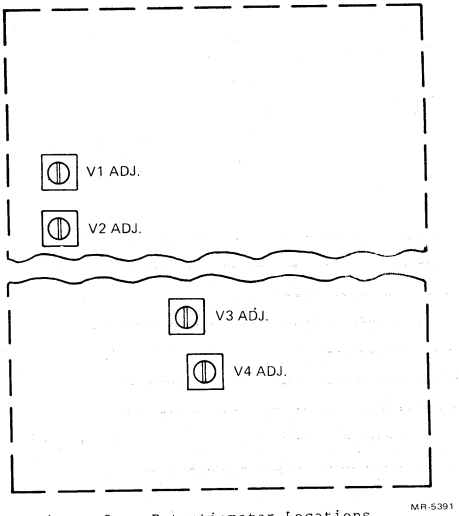


Figure 2 Potentiometer Locations

Cooling - Cooling for the S/X Bus Recorder is provided by two 4-inch fans located in the suitcase assembly. The fans are different for the 50 Hz and 60 Hz versions of the S/X Bus Recorder.

INSTALLATION

Introduction

This section provides all the information required to connect the S/X Bus Recorder into a system and disconnect it after testing is completed.

There are three variations of cabling, depending on the type of system being tested.

1. SBus using MA20 or MB20 memory
2. SBus using DMA
3. XBus using MF20/MG20 memory

After installation of the cables, the KL10 SBus recorder combination must be powered up and deskewed to align the clock in the S/X Bus Recorder to the clock in the KL10. This procedure is common to all cabling variations. After the deskewing procedure is completed, testing can proceed using the S/X Bus Recorder console panel and SBus diagnostic. On completion of testing, all recorder clocks must be checked and deskewed, cabling disconnected, and bus terminations replaced.

Cabling into System

Perform Procedures 1, 2, or 3 below, as appropriate, to connect the S/X Bus Recorder into the system. Set the bus and address switches as described in the Switch Settings section. Then proceed to the Deskewing Procedure section to perform the deskewing procedure.

Procedure 1: SBus with MA20/MB20 Memory

1. Power down the KL10.

2. Open the lid of the S/X Bus Recorder.

3. Remove the BC20 cables from the cable compartment in the recorder.

4. Rotate the two fasteners at the front of the console panel on the recorder one-quarter turn counterclockwise and hinge the panel upward.

5. Remove the two double height SBus terminator boards from MA20/MB20 memory in slots 1AB52 and 1CD52.

6. Install the terminator board removed from 1AB52 in slot 1CD7 of the S/X Bus Recorder.

7. Install the terminator board removed from 1CD52 in slot 1EF7 of the S/X Bus Recorder.

8. Plug one end of a BC20C-6C cable into slot 1CD5 of the recorder, and plug the other end into slot 1AB52 of the MA20/MB20.

9. Plug one end of the other BC20 cable into slot 1EF5 of the recorder, and the other end into slot 1CD52 of the MA20/MB20.

10. This completes cabling the recorder into the system. Proceed to the Switch Settings section and set the bus and address switches.

Procedure 2: SBus with DMA20

1. Power down the KL10.

2. Open the lid of the S/X Bus Recorder.

3. Remove the BC20C-6C cables from the cable compartment in the recorder.

4. Rotate the two fasteners at the front of the console panel on the recorder one-quarter turn counterclockwise and hinge the panel upward.

5. Remove the BC20C-6C cable from DMA slot 1AB01 and install it in recorder slot 1CD5.

6. Connect one end of a BC20C-6C cable into slot 1CD7 on the recorder, and the other end to slot 1AB01 on the DMA.

7. Remove the BC20C-6C cable from DMA slot 1CD01 and install it in recorder slot 1EF5.

8. Connect the remaining BC20C-6C cable from slot 1EF7 on the recorder, and slot 1CD01 on the DMA.

9. This completes cabling the recorder into the system. Proceed to the Switch Settings section and set the bus and address switches.

Procedure 3: XBus with MF20/MG20 Memory

1. Power down the KL10.

2. Open the lid of the S/X Bus Recorder.

3. Rotate the two fasteners at the front of the console panel on the recorder one-quarter turn counterclockwise and hinge the panel upward.

4. Remove the M8572YB board from slot 1A/F8 of the recorder.

5. Remove four terminations from PC22, PD22, PE22, and PF22 in the MF20/MG20.

6. Install these terminations on the M8572YB module.

7. Plug the M8572YB module into slot 1A/F8 of the recorder.

8. Connect the cables from the M8572 in slot 1A/F6 as follows.

From	To
P1	MF20/MG20/PC22
P2	MF20/MG20/PD22
P3	MF20/MG20/PE22
P4	MF20/MG20/PF22

9. This completes cabling the recorder into the system. Proceed to the Switch Settings section and set the bus and address switches.

Switch Settings - Switches on the G5347 boards select whether the recorder is used with the SBus or XBus. A switch on the G5348 board selects the recorder address if two recorders are used.

The bus select switches are S001 on each of two G5347 boards located in slots three and four of the recorder. (Refer to Figure 3.)

Set both switches to ON if the recorder is to be used on the XBus, or OFF for the SBus.

If two recorders are used, one is assigned address 36 and the other 37. The address switch is located at S001 on the G5348 board in slot one of each recorder, as shown in Figure 4.

Set the switch to OFF if the recorder address is 36, and ON if the recorder address is 37.

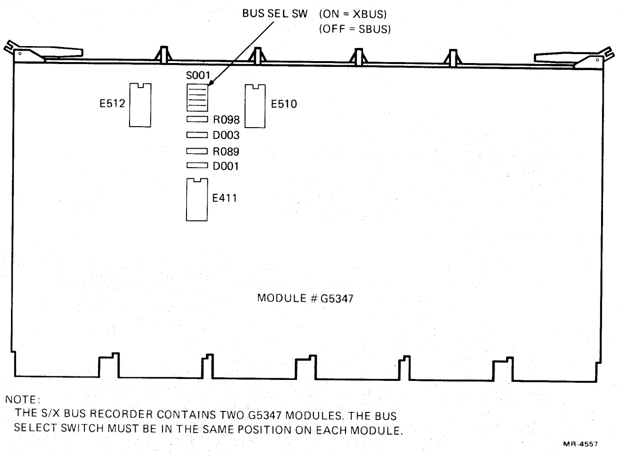


Figure 3 Bus Select Switch Location

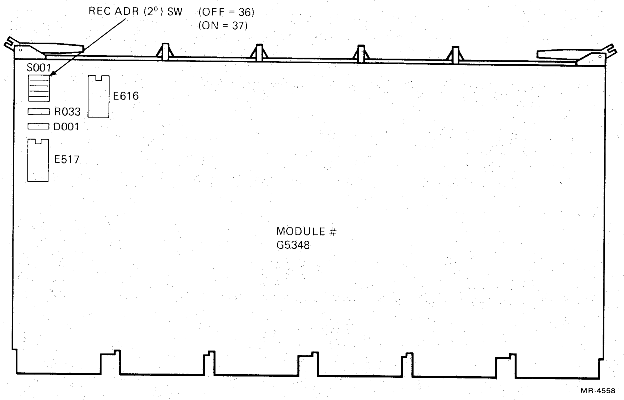


Figure 4 Recorder Address Switch Location

Deskewing Procedure - Perform either the MA20/MB20 or DMA20 deskew procedure, as appropriate. These procedures are in the Checks and Adjustments section of the KL10 Maintenance Guide, Volume I (EK-0KL10-MG). Then perform the following procedure to align the recorder clock to the KL10 clock. This procedure requires a Tektronix 475 oscilloscope or equivalent (100 MHz). Use identical probes and short ground clips.

1. Power up the KL10 and select CR0 on the KL10.
2. Type MR and FX1 to turn on the clock.
3. Set oscilloscope for external trigger, negative edge, and trigger from A CHANGE COMING L, at 4E22F2 on the KL10 backplane.
4. Attach channel one probe to MTR BOX CLK C, 4D33P1. Set the scope to 0.5 V/cm with the ground reference 1.3 V above the centerline.
5. Press TRIGGER VIEW and observe that the relationship of MTR MBOX CLKC to A CHANGE COMING L corresponds to that shown in Figure 7.
6. Set the leading edge of the first A phase clock on the first division of the scope graticule.
7. Connect channel two probe of the scope to R31 on the G5348 in the recorder, as shown in Figure 5.
8. Adjust CLKA delay line on the G5349 in slot 1CD2 of the recorder (see Figure 6) so that the leading edge 50% point crosses the leading edge 50% point of MBox A phase clock (see Figure 7).
9. Connect channel two probe to R32 on the G-5348 board in the recorder, as shown in Figure 5.
10. Adjust CLKB delay line on the G5349 board in slot 1CD2 of the recorder (see Figure 6) so that the leading edge 50% point crosses the leading edge 50% point of MBox phase B clock (see Figure 7).
11. This completes the deskewing procedure for the S/X Bus Recorder.

Removal of Interconnecting Cables
At the completion of testing, remove all cables and replace terminations in the original locations. This procedure is the reverse of installation. Recheck clocks and deskew if necessary.

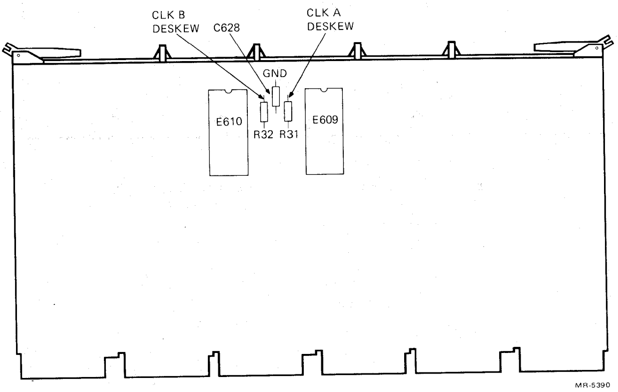


Figure 5 Oscilloscope Connection Points

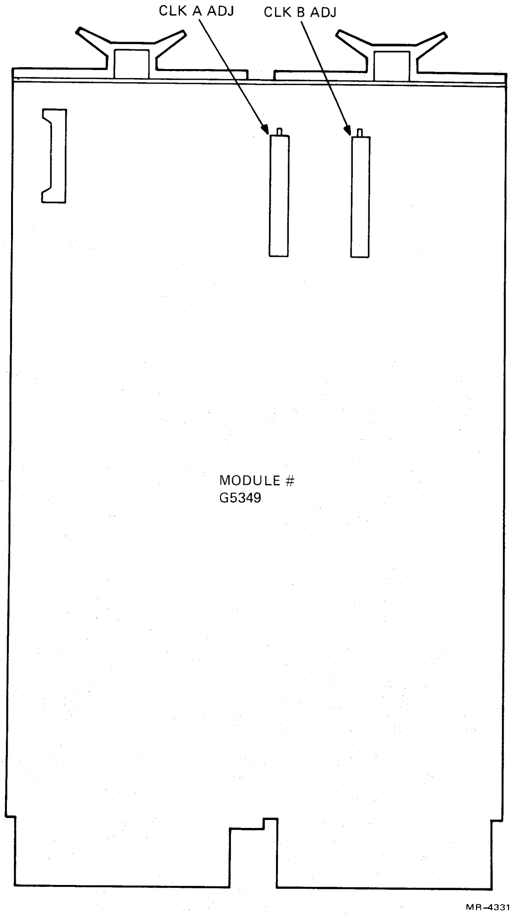


Figure 6 Deskewing Adjustments

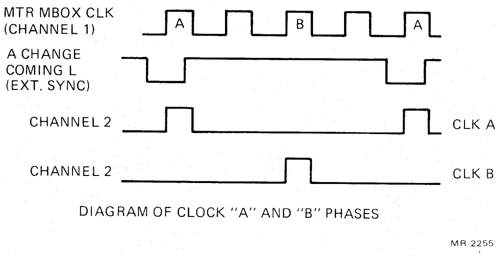


Figure 7 Deskewing Waveforms

OPERATION

Introduction
The following paragraph describes the controls and indicators on the recorder console panel. Refer to Figure 8.

Controls and Indicators

1. LOCKOUT switch - The LOCKOUT switch is a 2-position toggle. In the OFF position, the S/X Bus Recorder functions are controlled by the software and the START PULSE and STOP switches. In the ON position, the console panel functions are enabled for manual intervention, and the software is preempted.

2. STOP ON ERROR switch - The STOP ON ERROR switch is a 2-position toggle. In the OFF position, the S/X Bus Recorder continuously samples the S/X bus, and the REC decimal point remains lit. In the ON position, any error will cause the recorder to stop recording, and the REC decimal point goes out.

3. STOP switch - The STOP switch is a 2-position toggle. In the ON position, recording is terminated by clearing the record flip-flops at the next START "A" or START "B" from the S/X bus. In the OFF position, recording commences at the next START "A" or START "B" from the S/X bus, or START pulse from the recorder.

4. START PULSE - The START PULSE is a pushbutton switch. When the STOP switch is ON, depressing the START PULSE switch only clears the record flip-flop. When the STOP switch is OFF, depressing the START PULSE switch clears the recorder and begins recording on the next START "A" or START "B".

5. NXT/LD and FUNC SELECT switches - The NXT/LD switch is a pushbutton switch that is functional only when the recorder is not recording. The function of this switch is determined by FUNC SELECT. When FUNC SELECT is set to DEC, pressing the NXT/LD switch decrements the RAM ADDRESS by one. If the FUNC SELECT is set to INC, pressing the NXT/LD switch increments the RAM ADDRESS by one. When FUNC SELECT is set to LD MADR, pressing the NXT/LD switch loads the memory address into bits 13-35 of the DRR/MEMADR register.

   If the FUNC SELECT is set to LD RADR, pressing the NXT/LD switch loads the RAM register with the address selected by the REG SEL/RAM ADR thumbwheel switches.

6. REG SEL/RAM ADR thumbwheel switches - The low-order (right-hand) thumbwheel switch is the only switch to control the LED display (except for lamp test). Both thumbwheels are used to select a RAM address from 0-37. Address 40 is used for lamp test of LED D7. Address 42 and the lamp test option in the S/X Bus Recorder diagnostic program test all other LEDs (D1-D6 and D8).

7. MEM ADR indicator - When lit, the MEM ADR decimal point indicates that the DR register is loaded with memory address data. If this decimal point is not lit, the DR contains normal data.

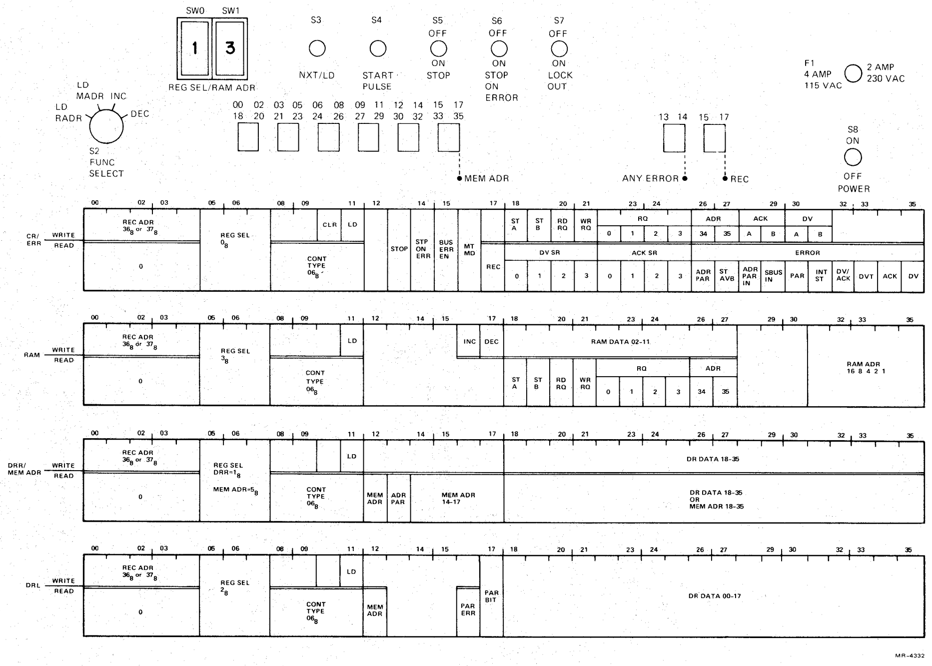


Figure 8   Console Panel Controls and Indicators

Register Descriptions

Control/Error Register (CR/ERR)

Bits 00-04 (REC ADR) - The S/X Bus Recorder is assigned a 5-bit discrete address (36 or 37 octal) for device selection when the SBus diag line is asserted. These bits are returned as zeros on the read portion of the SBus diag cycle. (Write-only bits)

Bits 05-07 (REG SEL) - Use a 3-bit field (zero octal) to select the CR/ERR register for writing and reading when the SBus diag line is asserted. (Write and read bits)

Bits 08-11 (CONT TYPE) - This is a hardwired controller type (06 octal) bit. If a recorder is connected to the SBus or XBus, these bits will be asserted for the read portion of the SBus diag cycle to indicate recorder present. The recorder's address (bits 00-04) must accompany the SBus diag cycle issuing a standard SBus diagnostic function zero to the recorder, which will read the recorder's controller type, and the CR/ERR register's read bits. (Read-only bits)

Bit 10 (CLR) - If the bit is a 1 and the console LOCKOUT switch is in the OFF position, clear and initialize the recorder's logic. (Write-only bit)

Bit 11 (LD) - If the bit is a 1 and the LOCKOUT switch is in the OFF position, allow the register's writable bits (18-35) to write and read all register bits. If the bit is a 0 and the LOCKOUT switch is in the OFF position, read all register bits. When the bit is a 1 or 0 and the LOCKOUT switch is in the ON position, no function is performed. (Write and read bits)

Bit 12 - Not used.

Bit 13 (STOP) - When this bit is a 1 and the console LOCKOUT switch is set to OFF, the REC flip-flop is cleared, the MAN STP flip-flop is set, its status is read, and the recording of bus signals is inhibited. If this bit is a 0 and the CLR bit is asserted, the MAN STP flip-flop is cleared. If the STOP and CLR bits are asserted simultaneously, the MAN STP flip-flop remains set. (Write and read bit)

Bit 14 (STP ON ERR) - If the bit is a 1 and the console LOCKOUT switch is in the OFF position, set the STP on ERR flip-flop and read its status. If the bit is a 0 and the CLR bit is asserted, clear the STP on ERR flip-flop. Simultaneous assertion of the STP on ERR and CLR bits leaves the STP on ERR flip-flop set. (Write and read bit)

Bit 15 (BUS ERR EN) - When the bit is a 1 and the console LOCKOUT switch is in the OFF position, set the BUS ERR EN flip-flop. This allows the recorder to assert SBUS ERR when an ERR is detected. If the bit is a 0 and the stop or CLR bit is asserted, clear the BUS ERR EN flip-flop. Simultaneous assertion of the BUS ERR EN, STOP, or CLR bits leaves the BUS ERR EN flip-flop set. (Write and read bit)

Bit 16 (MT MD) - When the bit is a 1 and the console LOCKOUT switch is in the OFF position, set the maintenance mode flip-flop and read its status. The MT MD flip-flop provides data paths from the SBus data lines to the selected register or to the RAM when loading data with the SBus diag line asserted. If the bit is a 0 and the CLR bit is asserted, clear the MT MD flop. Simultaneous assertion of the MT MD and CLR bits leaves the MT MD flip-flop set. (Write and read bit)

Bit 17 (REC) - When the bit is a 1, the REC (record) flip-flop is set. (Read-only bit)

> NOTE
>
> Writable bits 18-31: The MT MD flop must be set and the LD bit asserted for writable bits 18-31 of the CR/ERR register (refer to bits 11 and 16).

Bit 18 (ST A) - If the bit is a 1, load the required number of ACKNs and DVs per bits 20-25 into the ACK and DV shift registers. Also load the DV timeout counters, check for an INT ST ERR, and enable the ACK and DV shift registers to be shifted on the next and subsequent SBus diagnostics per bits 28-31.

Bit 19 (ST B) - When the bit is a 1, load the required number of ACKNs and DVs per bits 20-25 into the ACK and DV shift registers. Also load the DV timeout counters, check for an INT ST ERR, and enable the ACK and DV shift registers to be shifted on the next and subsequent SBus diagnostics per bits 28-31. When ST B is asserted, readable bits 18-35 are not guaranteed valid on the same SBus diagnostic cycle.

Bits 20-25 (RD RQ and WR) - RQ in conjunction with RQ 00/01/02/03 determine the number of ACKNS and data valids to be loaded when ST A or ST B is asserted.

Bits 26-27 - ADR 34 and ADR 35 in conjunction with RQ 00/01/02/03 are checked for an INT ST ERR when the ST A or ST B is asserted.

Bits 28-29 - ACK A and ACK B produces one shift of the ACK SR for either ACK A or ACK B. (Reference bit 18 - ST A and bit 19 - ST B)

Bits 30-31 - DV A and DV B produces one shift of the DV SR for either DV A or DV B. (Reference bit 18 - ST A and bit 19 - ST B)

Bits 32-35 - Not used.

NOTE

Readable bits 18-35: These bits are read during the read part of the SBus diagnostic cycle. (Refer to bit 11.)

Bits 18-21 (DV SR) - These bits reflect the status of the data valid shift register.

Bits 22-25 (ACK SR) - These bits reflect the states of the ACK shift register.

Bit 26 (ADR PAR ERR) - If the bit is a 1, the recorder has detected even parity in a memory address asserted on the SBus.

Bit 27 (ST A/B ERR) - When the bit is a 1, SBUS START "A" and START "B", or START "A", or START "B" and SBUS DIAG, were detected simultaneously.

Bit 28 (ADR PAR IN ERR) - If the bit is a 1, the SBUS ADR PAR ERR line was asserted.

Bit 29 (SBUS in ERR) - When the bit is a 1, the SBUS ERR line was asserted.

NOTE

Memory controllers (MF20) that generate SBUS ERR for a nonfatal error condition will stop the recorder.

Bit 30 (PAR ERR) - When the bit is a 1, even parity was detected in a SBus data word during the cycle. If MT MD is set, ACKN A or B, or DV A or B, does not clock the DR.

Bit 31 (INT ST ERR) - If the bit is a 1, an initial start error was detected and one of the following conditions occurred.

1. RQ OX = ADR 34/35 - The SBus request associated with SBUS ADR 34/35 was not asserted.

2. RMW = 1 WD RQ - More than one word was requested on the SBus for a read-modify-write cycle.

3. RD/WR RQ = 0 - No SBUS RD RQ or WR RQ was asserted for a valid SBus cycle.

Bit 32 (DV/ACK ERR) - If the bit is a 1, a data valid occurred before an ACKN.

Bit 33 (DV Timeout) - When the bit is a 1, a missing data valid has been detected. Timeout is produced when the memories' read-access time (ACKN to data valid) is exceeded.

Bit 34 (ACK ERR) - If the bit is a 1, ACK A and ACK B were detected simultaneously.

Bit 35 (DV ERR) - If the bit is a 1, data valid A and data valid B were detected simultaneously, or a data valid was detected when RD RQ was not asserted.

Data Register Left (DRL)

Bits 00-04 (REC ADR) - The S/X Bus Recorder is assigned a 5-bit discrete address (36 or 37 octal) for device selection when the SBus diag line is asserted. These bits are returned as zeros on the read portion of the SBus diag cycle. (Write-only bits)

Bits 05-07 (REG SEL) - Use a 3-bit field (02 octal) to select the DRL register for writing and reading when the SBus diag line is asserted. (Write and read bits)

Bits 08-11 (CONT TYPE) - This is a hardwired controller type (06 octal) bit. If a recorder is connected to the SBus or XBus, these bits will be asserted for the read portion of the SBus diag cycle to indicate recorder present. The recorder's address (bits 00-04) must accompany the SBus diag cycle issuing a standard SBus diagnostic function zero to the recorder, which will read the recorder's controller type, and the CR/ERR register's read bits. (Read-only bits)

Bit 11 (LD) - If the bit is a 1 and the LOCKOUT switch is in the OFF position, allow the register's writable bits (18-35) to write and read all register bits. Also set the MT MD flip-flop and read its status. This flip-flop provides data paths from the SBus data lines to a selected register or RAM when loading data with the SBus diag line asserted. If the bit is a 0 and the LOCKOUT switch is in the OFF position, read all register bits. If the bit is a 0 and the CLR bit is asserted, clear the MT MD flip-flop. Simultaneous assertion of the MT MD flip-flop and the CLR bit leaves the MT MD flip-flop set. When the bit is a 1 or a 0 and the LOCKOUT switch is in the ON position, no function is performed. (Write and read bits)

Bit 12 (MEM ADR) - If the bit is a 1, DR bits 14-35 contain a mem adr; DR bits 00-11 and 13 should be disregarded. This bit is set when the mem adr is read via the SBus diag or console and resets when other than a mem adr is clocked into the DR. (Read-only bit)

Bits 13-5 - Not used.

Bit 15 (PAR ERR) - If the bit is a 1, even parity was detected in the data register. If MT MD is set, loading the DRL will check the parity of the DR. (Read-only bit)

NOTE

The terms LOAD and NOT LOAD refer to the diagnostic and are used for diagnostic testing only.

Bit 17 (PAR BIT) - If the bit is a 1 and LOAD, set the DR parity flip-flop and read its status. If NOT LOAD, read the status of the DR parity flip-flop. (Write and read bit)

Bits 18-35 (DATA) - If LOAD, write bits 18-35 (SBus data) into the DRL (DR00-17) and read its contents. If NOT LOAD, read the contents of the DRL.

Data Register Right (DRR)

Bits 00-04 (REC ADR) - The S/X Bus Recorder is assigned a 5-bit discrete address (36 or 37 octal) for device selection when the SBus diag line is asserted. These bits are returned as zeros on the read portion of the SBus diag cycle. (Write-only bits)

Bits 05-07 (REG SEL) - This 3-bit field is used to select the DRR (01 octal) or the contents of the latches (05 octal). When the mem adr select (05 octal) is asserted, mem adr latches bits 14-35 are clocked into DR bits 14-35 on the write part of the SBus diag cycle and the mem adr flag (bit 13) is set.

Bits 08-11 (CONT TYPE) - This is a hardwired controller type (06 octal) bit. If a recorder is connected to the SBus or XBus, these bits will be asserted for the read portion of the SBus diag cycle to indicate recorder present. The recorder's address (bits 00-04) must accompany the SBus diag cycle issuing a standard SBus diagnostic function zero to the recorder, which will read the recorder's controller type, and the CR/ERR register's read bits. (Read-only bits)

Bit 11 (LD) - If the bit is a 1 and the LOCKOUT switch is in the OFF position, allow the register's writable bits (18-35) to write and read all register bits. Also set the MT MD flip-flop and read its status. This flip-flop provides data paths from the SBus data lines to a selected register or RAM when loading data with the SBus diag line asserted. If the bit is a 0 and the LOCKOUT switch is in the OFF position, read all register bits. If the bit is a 0 and the CLR bit is asserted, clear the MT MD flip-flop. Simultaneous assertion of the MT MD flip-flop and the CLR bit leaves the MT MD flip-flop set. When the bit is a 1 or a 0 and the LOCKOUT switch is in the ON position, no function is performed. (Write and read bits)

Bit 12 (MEM ADR) - If the bit is a 1, DR bits 14-35 contain a mem adr; DR bits 00-11 and 13 should be disregarded. This bit is set when the mem adr is read via the SBus diag or console and resets when other than a mem adr is clocked into the DR. (Read-only bit)

Bit 13 (ADR PAR) - If the bit is a 1, the address parity bit was asserted on the SBus/XBus.

Bits 14-17 (MEM ADR) - If mem adr (bit 13) is a 1, DR bits 14-17 reflect a mem adr. (Read-only bits)

Bits 18-35 (DATA) - If LOAD and DRR select, write bits 18-35 (SBus data) into DRR (DR18-35) and read its contents. If NOT LOAD and DRR select, read the contents of DRR (bits 18-35). If NOT LOAD and mem adr select, read the contents of the mem adr latches. The mem adr latches cannot be loaded via the SBus diag.

RAM Data Register - On each SBus transaction certain data is checked and other data is captured. The 10 bits that are captured are stored in a 32_{10} (37_8) word RAM. This data consists of START "A", START "B", qualifiers for read and write memory, four requests (0,1,2,3), and address bits 34 and 35 of the physical address. The RAM data is in bits 18-27 of the RAM register, as shown in Figure 8. During each RAM data cycle, either START "A" or START "B" must be asserted. RAM data should never be stored without START "A" or START "B" active. The bit definitions are as follows.

Bit 18 - START "A"
Bit 19 - START "B"
Bit 20 - Read Request
Bit 21 - Write Request
Bit 22 - Request 0
Bit 23 - Request 1
Bit 24 - Request 2
Bit 25 - Request 3
Bit 26 - Address Bit 34
Bit 27 - Address Bit 35

S/X Bus Recorder Operation and Dumping

S/X Bus Recorder Operation - After the deskewing of clock signals to the memories and S/X bus recorder, the LOCKOUT switch should be placed in the OFF position. Diagnostic DGSBA should be run in order to check out the functionality of the recorder. DGSBA is an 11-based diagnostic that can read and write registers contained within the recorder. After successful completion of DGSBA, the "B" command string should be run with the LOCKOUT switch in the ON position. All diagnostics should be run without errors, except for the following.

DHKBA	Test No. 19, Subtest 1, PC 31064
DHKBB	Test No. 1, Subtest 1, PC 22554
DGKBA	Test No. 19
DGKBB	Test No. 1

If no errors are found (except for those noted) the recorder is ready for system operation. To reset the recorder, proceed as follows.

1. Place the STOP switch in OFF position.

2. Place the LOCKOUT in ON position.

3. Enable the recorder to stop on error by placing the STOP ON ERROR toggle switch to the ON position.

4. Press the START PULSE button.

5. The recorder is now initialized. All errors are cleared and the ANY ERROR decimal point should now be OFF. The ACK and DV shift counters are initialized and recording will begin on the next START "A" or START "B" signal on the SBUS. Once the START "A" or START "B" signal is received by the recorder, the RECORD decimal point will light to indicate that the recorder is storing SBUS activity.

Steps 4 and 5 clear and arm the recorder. If the recorder detects an error, as described in the CR/ERR register bits 26-35 (refer to the Control/Error Register (CR/ERR) section), and the STOP ON ERROR switch is ON, recording will cease. Several methods are available in order to dump the contents of the recorder:

1. Manual (via switches on console).

2. Through the diagnostic DGSBA.

3. Through CCL file for KLDCP (another will be available with RSX-20F).

4. Through SBus DIAGs.

Dumping the Recorder via KLDCP or RSX20F - For RSX20F you must have Version 14. A new command has been added to Version 14. This command is the TAKE command, which will read and execute a command file. The available .CMD files for the S/X Bus recorder are

```
SB0.CMD   ;For SBus recorder addressed as #36
SB1.CMD   ;For SBus recorder addressed as #37
```

If you are using KLDCP, the I command for .CCL processing should be used. The .CCL files available are

```
SB0.CCL   ;SBus recorder addressed as #36
SB1.CCL   ;SBus recorder addressed as #37
```

Dumping the Recorder via DGSBA - To dump the recorder via DGSBA, perform the following steps.

1. To KLDCP type P DGSBA.

2. Ensure that the recorder's LOCKOUT switch is in the OFF position.

3. When KLDCP has returned with the >. prompt, type:

   SED/D

4. When the diagnostic starts, you will be required to select which SBus recorder you want. The diagnostic will print out the following information.

```
SBUS/XBUS RECORDER DIAGNOSTIC CONSOLE PACKAGE BEGINS

SBUS/XBUS RECORDER AVAILABLE:
RECORDER #0     ADDRESS 36

SELECT SBUS/XBUS RECORDER (0,1 OR B FOR BOTH) - 0

SBUS/XBUS RECORDER SELECTED:
REC #0     REC ADR 36

TYPE HLP<CR> IF YOU WANT INSTRUCTION
```

5. Once you reach SBA>, the command you give is RAL (READ ALL).

   This will give you a readout of all the registers.

```
SBA>RAL

CR REG = 000610 356140
DRL REG = 004602 351400
DRR REG = 002620 010316
MEM ADR REG = 012660 137144

RAM REG = 00660 336001
RAM REG = 00660 276000
RAM REG = 00660 336037
RAM REG = 00660 276036
RAM REG = 00660 276035
RAM REG = 00660 476034
RAM REG = 00660 276033
RAM REG = 00660 476032
RAM REG = 00660 276031
RAM REG = 00660 476030
RAM REG = 00660 276027
RAM REG = 00660 476026
RAM REG = 00660 276025
RAM REG = 00660 476024
RAM REG = 00660 276023
RAM REG = 00660 476022
RAM REG = 00660 276021
RAM REG = 00660 476020
RAM REG = 00660 276017
RAM REG = 00660 476016
RAM REG = 00660 276015
RAM REG = 00660 476014
RAM REG = 00660 276013
RAM REG = 00660 476012
RAM REG = 00660 276011
RAM REG = 00660 476010
RAM REG = 00660 276007
RAM REG = 00660 476006
RAM REG = 00660 276005
RAM REG = 00660 476004
RAM REG = 00660 276003
RAM REG = 00660 476002
SBA>EXT
CMD:
```

Manual Dumping of the S/X Bus Recorder - To manually dump the S/X Bus recorder, proceed as follows.

1. Adjust the following button and two switches.

   a. Put the LOCKOUT switch to the ON position.
   b. Put the STOP switch to the ON position.
   c. Press the ST PLS button, which will ensure that REC is reset.

   CAUTION
   Depressing the ST/PLS button with the STOP switch in the OFF position will clear the recorder.

2. Put a 00 in the REG SEL/RAMADR thumbwheel switch. This will gate out the CR/ERR register information, which is contained in bits 18-35, into the LEDs. Record this information.

3. Put a 01 in the REG SEL/RAMADR thumbwheel switch. This will gate out the contents of the Data Reg Right (DRR).

4. Put a 02 in the REG SEL/RAMADR thumbwheel switch. This will gate out the contents of the Data Reg Left (DRL).

5. Turn the rotary switch to the MEMADR position.

   CAUTION
   Before performing the next step make sure you have copied down the DRR, as it will be overwritten with MEMADR.

6. Press the NXT/LD switch. The MEMADR decimal point should light, indicating that memory address data is now loaded in the DRR.

7. Put a 05 in the REG SEL/RAMADR thumbwheel switch. This will gate out the contents of the MEMADR.

8. Put a 03 in the REG SEL/RAMADR thumbwheel switch.

9. Turn the function select rotary switch to DEC position and press the ST/PLS switch. The LEDs now display information for the SBus cycle in which the error(s) was (were) detected.

Operation with MOS Memory and TGHA - In order for the S/X Bus Recorder to run without detecting single-bit recoverable errors, TGHA must not be run.

Under TOPS-10, the OPR.ATO file must be edited so that TGHA will not run. This can be done by inserting an exclamation point before logging, defining, and running TGHA.

```
Example:   !  :SLOG
           !  :DEF   TGHA=
           !  TGHA-  R  TGHA
```

In the case of TOPS-20, rename the file <SYSTEM> TGHA.EX1 rather than <SYSTEM> TGHA.EXE.

NOTE

Once you are done using the S/X Bus Recorder, remove the exclamation point from OPR.ATO for TOPS-10 or rename the file <SYSTEM> TGHA.EXE (instead of <SYSTEM> TGHA.EX1).

Error Interpretation
The S/X Bus Recorder has the capability of detecting write parity errors on the other S/X Bus. This is possible because every time a data valid is received in the recorder, parity is checked. You can determine that the error occurred on the other bus by examining the CR/ERR register. If the CR/ERR register contains a 046040 in the right half, which indicates DV #3, ACK #2, ACK #3, and parity error, a parity error occurred on the write portion of a read-pause-write on the other S/X bus.

The following are five examples of errors that can occur on the S/X Bus, and how to interpret them using the recorder.

Internal Memory Failures - The following five examples of failures are internal memory failures and XBus errors.

Data Parity Error #1 - This error is an example of a data parity error on word #3, with the transfer starting on word zeros boundary.

The following program was put in location 100 and 101 of the internal memory:

```
100/ MOVE 10, 1000
101/ JRST 4,   101
```

Paging was set up so that page one (right half of location 600 of Exec Base register) was pointing at physical page 1000. The EBR was at page zero. The program was run one time in order to move the program to cache. Next, cache look and load was turned off (by executing a CONO PAG,20000). Bad parity was then put into loc 1003 by executing the following instructions.

```
CONO PI,200000   (700600 200000)  ;write even parity
MOVEM 1003  (202000 1003) ;store away AC0 in location 1,,001003
;with bad parity
CONO PI,0 700600 0 ;turn off write even parity
```

Cache look and load was then enabled with a CONO PAG,620000 (701200 620000). Next, the physical page 1001 was invalidated in cache to allow a 4-word memory request (the program at loc 100 and 101 was in cache and was valid).

When the machine was started at location 100 it executed a MOVE to AC0 location 1000. Since cache was enabled and those words were not in cache at the time, a 4-word request was generated on the SBus starting at location 1000. In analyzing the contents of the CR/ERR register note that all the data valid and ACK shift registers are cleared. The recorder disables the clearing of data valid and ACK shift registers when an error is detected. The fact that all the shift registers are cleared indicates that the error happened on word #3. By looking at the data contained in the memory address register (MEM ADR), determining the amount of requests that were sent and finding out which data valid was cleared out last, one can determine the failing word. The physical location of the failing word is 1,,001003.

The program was started at location 100. The following is the console printout and the execution of the CCL file called SB0.CCL This is at KLDCP level. An * indicates a comment inserted to explain data.

```
>.
KL10 HALTED PC /000100 VMA/000101
>.I SB0
;S/X BUS RECORDER DUMPER
;THIS WILL WORK IF RECORDER IS ADDRESSED AS 36
;
;******  THE LOCKOUT SWITCH MUST BE IN THE OFF POSITION***
;
;FIRST STOP THE RECORDER
DM10:740020 0
EX700500 10
;NEXT GET CR/ERR REGISTER
EM11
000011/000630 000040   *Bit 30 set (parity error).  Nothing was
                       *left in data valid and ACK shift registers.
;
;NOW GET DATA REG LEFT (DRL)
DM10:744000 0
EX700500 10
EM11
000011/004603 275500   *Bit 16 indicates parity error.
;
;NOW GET DATA REG RIGHT (DRR)
DM10:742000 0
EX700500 10
EM11
000011/002600 000400
;
;NOW GET MEMORY ADDRESS REGISTER (MEMADR)
DM10:752000 0
EX700500 0
EM11
000011/012641 001000   *Indicates address as being 1,,001000
;
;NOW GET ALL THE RAMS
DM10:746001 0
EX700500 10            *This command is executed twice in order
EX700500 10,EM11       *to get into the correct RAM address.
000011/006600 336007   *This is the RAM that was valid at time of
                       *error.  It shows that it was started on
                       *START "B", was a READ REQUEST, and was a
                       *4-word request.
EX700500 10,EM11
000011/006600 336006
EX700500 10,EM11
000011/006600 243405
EX700500 10,EM11
000011/006600 336004
EX700500 10,EM11
000011/006600 503403
EX700500 10,EM11
000011/006600 243402
EX700500 10,EM11
000011/006600 503401
EX700500 10,EM11
000011/006600 536000
EX700500 10,EM11
000011/006600 520037
EX700500 10,EM11
000011/006600 520036
EX700500 10,EM11
000011/006600 520035
EX700500 10,EM11
000011/006600 520034
EX700500 10,EM11
```

```
000011/006600 520033
EX700500 10,EM11
000011/006600 520032
EX700500 10,EM11
000011/006600 520031
EX700500 10,EM11
000011/006600 520030
EX700500 10,EM11
000011/006600 520027
EX700500 10,EM11
000011/006600 520026
EX700500 10,EM11
000011/006600 520025
EX700500 10,EM11
000011/006600 520024
EX700500 10,EM11
000011/006600 260023
EX700500 10,EM11
000011/006600 320022
EX700500 10,EM11
000011/006600 336021
EX700500 10,EM11
000011/006600 536020
EX700500 10,EM11
000011/006600 443417
EX700500 10,EM11
000011/006600 336016
EX700500 10,EM11
000011/006600 310415
EX700500 10,EM11
000011/006600 320014
EX700500 10,EM11
000011/006600 450413
EX700500 10,EM11
000011/006600 305012
EX700500 10,EM11
000011/006600 510411
EX700500 10,EM11
000011/006600 320010
;NOW RESET THE RECORDER AND SET STOP ON ERROR
DM10:740210 0
EX700500 10,EM11
000011/000610 000000
```

Data Parity Error #2 - This error is an example of a combination data parity error and data valid timeout (starting on word zero boundary).

In this example, the same program was initialized as in Data Parity Error #1 except that the parity error was put into physical location 1,,001002, or paged location 1002. In analyzing the contents of the CR/ERR register, two errors can be found: 1) a parity error, and 2) a data valid timeout. Once the data valid timers are started they will continue even after an error is detected. In this case, the real error is the parity error. Since the one data valid is still left, the recorder will inhibit clearing of any ACK or DV bits on detection of an error. The error occurred prior to receiving the data valid #3. If you look at the RAM at the time of the error, it had a 536000, which is a 4-word request beginning on word zero boundary. Since the recorder stopped on error, the data valid shift register has DV #3 left; it was a 4-word request, the error happened on word #2. The address contained in the MEM ADR register is 1,,001000. This is the initial address requested. By knowing the error happened on word two, the parity error occurred at location 1,,001002.

The printout is as follows.

```
CMD:
>.
>.I SBO
;S/X BUS RECORDER DUMPER
;THIS WILL WORK IF RECORDER IS ADDRESSED AS 36
;
;******  THE LOCKOUT SWITCH MUST BE IN THE OFF POSITION***
;
;FIRST STOP THE RECORDER
DM10:740020 0
EX700500 10
;NEXT GET CR/ERR REGISTER
EM11
```

```
000011/000630 040044  *Bit 30 = parity error,
                      *Bit 33 = data valid timeout
                      *Bit 21 = data valid #3 is not shifted out

;
;NOW GET DATA REG LEFT (DRL)
DM10:744000 0
EX700500 10
EM11
000011/004602 777777  *Bit 16 indicates parity error
                      *Bit 17 indicates parity bit

;
;NOW GET DATA REG RIGHT (DRR)
DM10:742000 0
EX700500 10
EM11
000011/002600 777777
;
;NOW GET MEMORY ADDRESS REGISTER (MEMADR)
DM10:752000 0
EX700500 0
EM11
000011/012641 001000  *Indicates address as being 1,,001000
;
;NOW GET ALL THE RAMS
DM10:746001 0
EX700500 10
EX700500 10,EM11
000011/006600 536000 *This is the RAM that was valid at time of
                     *error.  It shows that it was started on
                     *START "B", was a READ REQUEST and was a
                     *4-word request with address bit 34 and 35 =
                     *0. (This 4-word request begins on even quad
                     *word boundary, i.e., word zero.)
EX700500 10,EM11
000011/006600 520037

;
;
```

Data Parity Error #3 - This error is an example of the combination of a data parity error and a data valid timeout (starting in other than a word zero boundary).

In this example, the starting location for the 4-word request was 1,,001001. The contents of the CR/ERR register contain the same data as in Data Parity Error #2. In this example, however, the beginning of the 4-word request was not on word zero of the 4-word request, but word one of the 4-word request. Determine what was the original word requested and how many requests were asked for. After you have established this, see what is left in data valid and ACK shift registers in the CR/ERR register. In this example, data valid #3 is left. Again, as in example #2, it is the cause of the DVT (data valid timeout). The parity error is the real error on word #2. Since the quad word fetch started on word one of a 4-word fetch, the address that failed was 1,,001003.

REMEMBER: The data valid and ACK shift registers are modulo four type registers and that the data contained in them are relative to the beginning address of the transfer (RAM ADR bits 34 and 35). SBus address bits 34 and 35 are contained in RAM register bits 26 and 27.

The printout is as follows.

```
CMD:
>.
>.I SB0
;S/X BUS RECORDER DUMPER
;THIS WILL WORK IF RECORDER IS ADDRESSED AS 36
;
;****** THE LOCKOUT SWITCH MUST BE IN THE OFF POSITION***
;
;FIRST STOP THE RECORDER
DM10:740020 0
EX700500 10
;NEXT GET CR/ERR REGISTER
EM11
000011/000630 040044  *Bit 30 = parity error
                      *Bit 33 = data valid timeout
                      *Bit 21 = one data valid not shifted out
```

```
;
;NOW GET DATA REG LEFT (DRL)
DM10:744000 0
EX700500 10
EM11
000011/004603 000000  *Bit 16 indicates parity error
;
;NOW GET DATA REG RIGHT (DRR)
DM10:742000 0
EX700500 10
EM11
000011/002600 000001
;
;NOW GET MEMORY ADDRESS REGISTER (MEMADR)
DM10:752000 0
EX700500 0
EM11
000011/012641 001001  *Indicates address as being 1,,001001
;
;NOW GET ALL THE RAMS
DM10:746001 0
EX700500 10
EX700500 10,EM11
000011/006600 536400 *This is the RAM that was valid at time of
                     *error.  It shows that it was started on
                     *START "B", was a READ REQUEST, and was a
                     *4-word  request  with  address  bit 34 = 0
                     *and address bit 35 = 1.
EX700500 10,EM11
000011/006600 520037
EX700500 10,EM11
000011/000610 000000
;
```

Data Parity Error #4 - This error is similar to Data Parity Error #2, except that the word that failed was word zero of the transfer.

In this example, the parity error was put in location 1,,001000. The instruction in location 100 was a MOVE 10,1000. This would create a 4-word request (with cache enabled). At the CR/ERR register, notice that three data valids were not shifted out, and that the last error occurred on the first word of the transfer.

The address in the MEM ADR register is 1,,001000. Since it was the first word transferred that got the error, address 1,,001000 is the bad one.

The printout is as follows.

```
>.
>.I SBO
;S/X BUS RECORDER DUMPER
;THIS WILL WORK IF THE RECORDER IS ADDRESSED AS 36
;
;******  THE LOCKOUT SWITCH MUST BE IN THE OFF POSITION***
;
;FIRST STOP THE RECORDER
DM10:740020 0
EX700500 10
;NEXT GET CR/ERR REGISTER
EM11
000011/000630 340044  *Data valid 1, 2 and 3 still left parity
                      *error and data valid timeout.
;
;NOW GET DATA REG LEFT (DRL)
DM10:744000 0
EX700500 10
EM11
000011/004602 777777
;
;NOW GET DATA REG RIGHT (DRR)
DM10:742000 0
EX700500 10
EM11
000011/002600 777777
;
;NOW GET MEMORY ADDRESS REGISTER (MEM ADR)
DM10:752000 0
EX700500 10
EM11
000011/012641 001000  *Initial address latched = 1,,001000
```

```
;NOW GET ALL THE RAMS
DM10:746001 0
EX700500 10
EX700500 10,EM11
000011/006600 536000  *START "A", read request, 4-word request
                      *(RQ0,1,2,3), address 34 = 0, address 35 = 0
EX700500 10,EM11
000011/006600 520037
EX700500 10,EM11
000011/006600 520036
EX700500 10,EM11
                      *The rest of the RAMs are not displayed in
                      *this example since they are not needed.
```

Read-Pause-Write Failure (Write Portion) - On Read-Pause-Write (RPW) operations, a 1-word request is always generated. On receiving a RPW operation, the recorder will load the ACK/DV shift register with two ACKs and two data valids. This is done to identify which portion of the RPW cycle had the problem. Since the memory will respond with one data valid on the read portion of the cycle, this will clear one of the data valids that was loaded in the ACK/DV shift register. When the CPU sends the data out on the write portion, the CPU will send data valid to the memory, which will also clear the other data valid contained in the ACK/DV shift register. Therefore, if the ACK/DV shift register contained one data valid, the error happened on the READ portion. If there are no data valids left, the error occurred on the write portion.

In the following example an AOS instruction was executed after enabling write even parity. This latched an error on the write portion of the cycle.

```
>.
>.I SB0
;SB0.CCL -- CCL FILE FOR DUMPING RECORDER # 0
;LSGMEG 9-NOV-79
;THIS WILL WORK IF THE RECORDER IS ADDRESSED AS 36
;
;***** THE LOCKOUT SWITCH MUST BE IN THE OFF POSITION***
;
;FIRST STOP THE RECORDER
DM10;740020 0,EX700500 10
;NEXT GET CR/ERR REGISTER
EM11
000011/000630 002040   *This indicates a parity error and one ACK
                       *left to be shifted.
                       *This ACK is normal case on RPW.
;NOW GET DATA REG LEFT (DRL)
DM10:744000 0,EX700500 10,EM11
000011/004603 000000
;NOW GET DATA REG RIGHT (DRR)
DM10:742000 0,EX700500 10,EM11
000011/002620 000001
;NOW GET MEMORY ADDRESS REGISTER (MEM ADR)
DM10:752000 0,EX700500 10,EM11
000011/012660 000100
;
;NOW GET ALL THE RAMS
DM10:746001 0,EX700500 10
EX700500 10,EM11
000011/006600 560000   *This is the RAM at the time of failure.
                       *It indicates the cycle started on
                       *START "A", had RD and WR asserted, and
                       *request zero.
```

Since there were no data valids left, this happened on the write portion.

Read-Pause-Write Failure (Read Portion) - In this example the AOS instruction was executed again. Since the previous example had written bad parity into core, that same location now failed on the read portion of the RPW cycle as follows.

```
>.
>.I SBO
;SBO.CCL -- CCL FILE FOR DUMPING RECORDER # 0
;LSGMEG 9-NOV-79
;THIS WILL WORK IF RECORDER IS ADDRESSED AS 36
;
***** THE LOCKOUT SWITCH ON THE RECORDER MUST BE IN THE OFF
POSITION ***
;
;FIRST STOP THE RECORDER
DM10;740020 0,EX700500 10
;NEXT GET THE CR/ERR REGISTER
EM11
000011/000630 042040  *Bit 21 indicates data valid #3 left.
                      *Bit 25 indicates ACK #3 left (normal on
                      RPW).
                      *Bit 30 indicates parity error.

;NOW GET THE DATA REG LEFT (DRL)
DM10:744000 0,EX700500 10,EM11
000011/004603 000000
;NOW GET THE DATA REG RIGHT (DRR)
DM10:742000 0,EX700500 10,EM11
000011/002620 000001
;NOW GET THE MEMORY ADDRESS REGISTER (MEM ADR)
DM10:752000 0,EX700500 10,EM11
000011/012660 000100
;
;NOW GET ALL THE RAMS
DM10:746001 0,EX700500 10
EX700500 10,EM11
000011/006600 560000  *This indicates the cycle started with a
                      *START "B", and a one word request with
                      *read and write asserted.
```

There is still one data valid left to be shifted out. This means that the error latched on the first half of the cycle (read).

DMA Errors - (This section will be supplied at a later date.)

NOTE

DMA errors involving an incomplete cycle may point to an incorrect address. Read or write errors point to the correct address.

Loading and Running Diagnostics

Program Abstract - DGSBA is a diagnostic for the S/X Bus Recorder only. It is not a diagnostic for the KL10 memory system. However, DGSBA should be used to verify that the recorder is properly installed.

DGSBA is only a functional diagnostic. It was not designed to do any gate- or board-level callout. The error messages indicate the symptoms of an error as a result of what type of operation occurs.

Requirements

Preliminary Software - DGKAA, DGKAB, and DGKBA should be run first. Next, the memory should be configured; then DGSBA should be run. If the memory could not be configured, DGSBA should be run anyway because it may be the recorder that is polluting the memory bus.

Operating Procedure

Loading Procedure - DGSBA is supplied as an ".All" file and can be loaded from DECtape, flexible diskette, or a front-end RP04 or RP06 pack by selecting the device (see KLDCP operating procedures, EK-0KL10-MG) and typing: "P DGSBA.All".

Operation Modes and Switches - If DGSBA is started with all switches 0, and no / switches, it will run all tests that do not require operator intervention. Any and all recorders will be tested.

If the memory is configured, the last set of tests will be run. If the memory is not configured, the last set of tests will be skipped, and a message stating that a test has been skipped due to no memory will be printed.

SED/H The /H switch will type out the help test. The printout of the test is as follows.

```
SBD    DO SBUS DIAG
LCE    LOAD CONTROL/ERROR REGISTER
LDL    LOAD DATA REGISTER LEFT
LDR    LOAD DATA REGISTER RIGHT
LRM    LOAD RANDOM-ACCESS MEMORY
RCE    READ CONTROL/ERROR REGISTER
RDL    READ DATA REGISTER LEFT
RDR    READ DATA REGISTER RIGHT
RRM    READ RANDOM-ACCESS MEMORY
RAL    READ ALL REGISTERS AND RANDOM-ACCESS MEMORY
IRM    INCREMENT RAM ADDRESS
DRM    DECREMENT RAM ADDRESS
EXT    BACK TO KLDCP
HLP    PRINT HELP MESSAGE
SWI    PRINT CURRENT STATE OF SWITCH
IDT    IDENTIFY SBUS/XBUS RECORDER NUMBER
CFG    CHANGE SELECTION OF SBUS/XBUS RECORDER
HLT    HALT PROGRAM OPERATION
LPT    LAMP TEST
```

SED/D The /D switch will put DGSBA into diagnostic debug mode. This mode allows the operator to examine and modify the registers in the selected recorder.

SED/S The /S switch will ask the operator which of the recorders that are on-line are to be tested. This switch forces the operator to select recorders even if the OPRSEL switch is not up.

Test Control Switches

```
Name       Switch     Function

ABORT =    100000     ;Abort at program pass completion

RSTART =   40000      ;Restart test
TOTALS =   20000      ;Print test totals
NOPNT =    10000      ;Inhibit all printout (except forced)

PNTLPT =   4000       ;Print on line printer
DING =     2000       ;Ring TTY bell on error
LOOPER =   1000       ;Loop on error

ERSTOP =   400        ;Halt on error
PALERS =   200        ;Print all errors
RELIAB =   100        ;Reliability run mode

TXTINH =   40         ;Test inhibit
INHPAG =   20         ;Inhibit paging
MODDVC =   10         ;Modify device code

INHCSH =   4          ;Inhibit cache
OPRSEL =   2          ;Operator selection
CHAIN =    1          ;Chain control switch
```

Table of Contents

INSTALLATION

Off-Line Operation

To energize and check out the ARM-10LS, perform the following procedures.

Power Connection - The power connection is made at the EPO assembly. Receptacles located at the rear of this assembly are illustrated in Figure 1. All receptacles are reached via the accessible side panel. The following paragraphs describe the power connection procedures.

NOTE

Refer to the OPERATION section for a complete description of all controls and indicators.

Before power is applied to the memory, ensure that all connectors are secured.

CAUTION

Applying incorrect power will severely damage the equipment. Use 208 V, 3-phase, 60 Hz, or 220/240 V, 1-phase, 50 Hz for standard order units.

Before plugging the power cable into the site power outlet, check the power at the outlet with an ac voltmeter to verify correct voltage levels and proper receptacle wiring. Once site power is verified, plug the male of the cabinet power cable into the site power outlet.

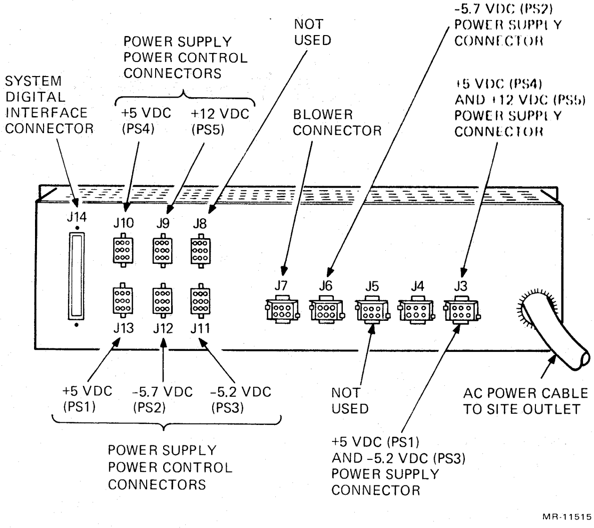


Figure 1 Rear View of EPO Assembly

The information on the ARM-10LS is included here by permission of Ampex Corporation.

Initial Power-On Procedure - Perform the following procedure to apply power to the memory. Controls and indicators are illustrated in the OPERATION section.

1. At the rear of the blower module, check that the circuit breaker button is pushed in.

2. On EPO front panel:

   a. Set MAIN POWER circuit breaker to OFF.

   b. Loosen the two inner thumbscrews on the panel and remove the EPO subpanel assembly. Ensure that the five DIP switches at the top of the EPO PWBA are set correctly, as shown in Figure 2. Replace the assembly.

   c. Set REMOTE/LOCAL switch to LOCAL.

   d. Set all VOLTAGE MARGIN switches to the center position.

   e. Set MAIN POWER circuit breaker to ON.

   f. Press POWER ON pushbutton and hold for three seconds. This step ensures proper power-up sequence.

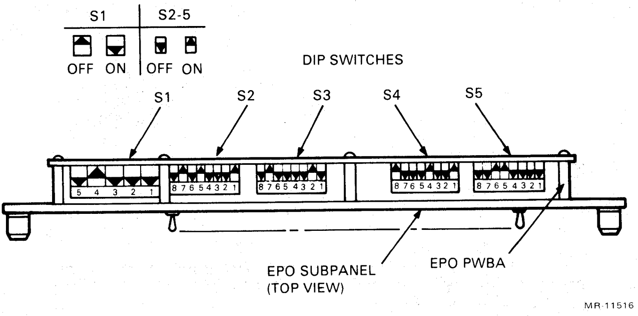


Figure 2   EPO PWBA Switch Settings

3. Check that the fans are operating.

NOTE
Memory will not turn on unless fans are operating.

4. Voltage Check: Verify/adjust power supply outputs to normal voltage defined in Table 1. Figure 3 shows the voltage test points, and Figure 4 shows the power supply voltage adjustment locations.

Table 1 DC Voltage Parameters

Power Supply	Nominal DC Output	Tolerance Low	High
PS1	+5 Vdc	+4.95 Vdc	+5.05 Vdc
PS2	-5.7 Vdc	-5.75 Vdc	-5.65 Vdc
PS3	-5.2 Vdc	-5.25 Vdc	-5.15 Vdc
PS4	+5 Vdc	+4.95 Vdc	+5.05 Vdc
PS5	+12 Vdc	+11.95 Vdc	+12.05 Vdc

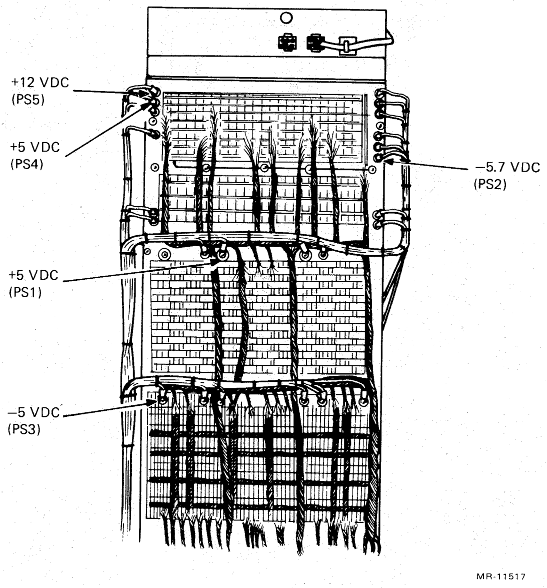


Figure 3 ARM-10LS Voltage Check Points

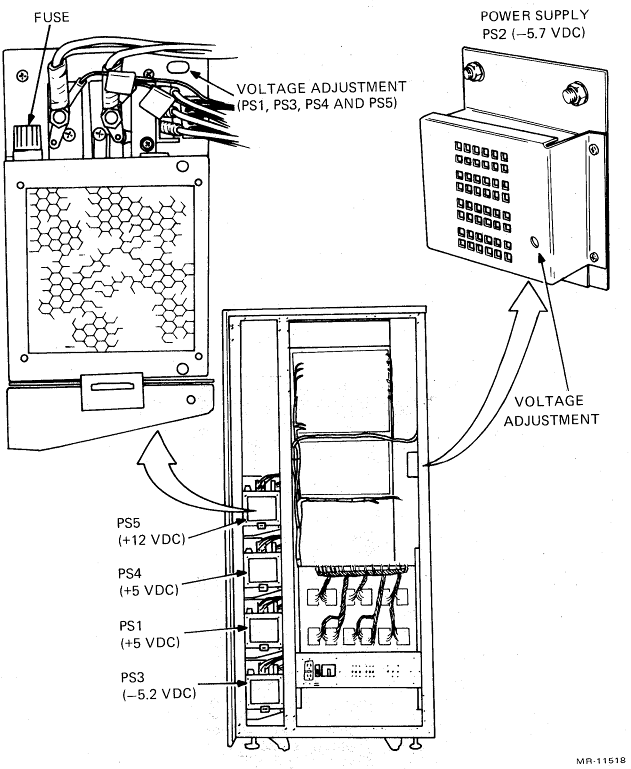


Figure 4 Power Supply Voltage Adjustment Locations

5. VTH Adjustments: The following adjustments must be made for all Transceiver PWBAs. (Card locations: CJ102-CJ109, CJ111-CJ118, CJ120-CJ127.)

   a. Set the digital multimeter to the lowest practical scale and measure dc voltage at test point lugs on the Transceiver PWBA. Refer to Figure 5 for location.

   b. Adjust potentiometer R4 until the voltage read on the digital multimeter is nominally -1.65 ± 0.01 Vdc.

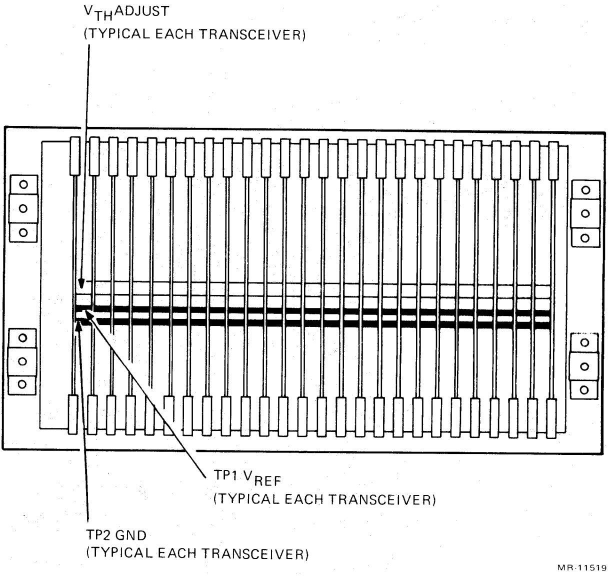


Figure 5 VTH Test Points

General Component Information

Figure 6 shows the placement of all PWBA boards in the rear of the ARM-10LS. (A similar chart appears on the inside of the unit's rear door.)

Tables 2, 3, and 4 cross reference the Ampex PWBA board part numbers with Digital's part numbers for the boards. Because Digital now supports the ARM-10 memory, you can order parts directly from Digital.

Figure 7 shows the locations of the various terminator and resistor assemblies and voltage test points on the backpanel.

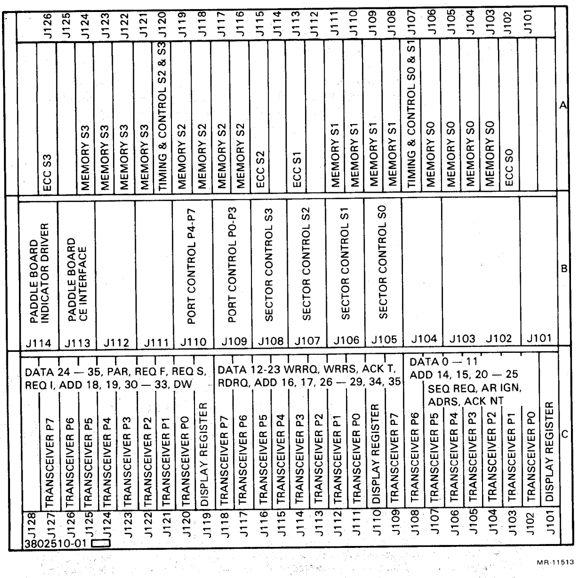


Figure 6 Card Placement Locations

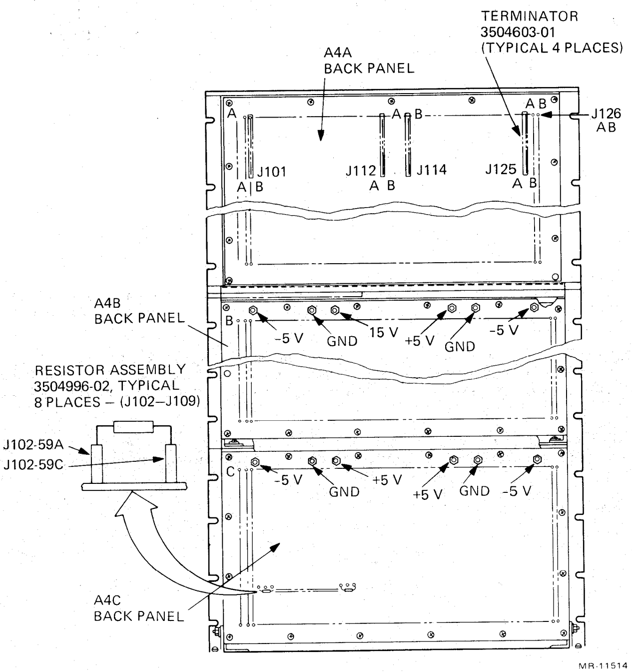


Figure 7 Terminator and Resistor Assemblies

Table 2 PWBA Table - 16 K Modules with Validate Storage

Abbr.	Title	Ampex Part No.	Digital Part No.
DR	Display Register	3506766-01**	29-80348
TR	Transceiver	3506195-01*	29-80350
CI	Control Interface	3506182-01**	29-80383
ID	Indicator Driver	3502283-01	29-80369
SC	Sector Control	3506180-01**	29-80352
PC	Port Control	3504400-__+	29-80371+
TC	Timing & Control	3502924-02	29-80374
M	Memory BD	3502915-02	29080372
ECC	ECC	3504406-03	29-80373

* Alternate part - 3281609-01
** Contains validate storage feature
\+ Part number depends on storage size:

Storage	Ampex Part No.	Digital Part No.
256 K	3504400-01	--
512 K	3504400-02	--
768 K or 1024 K	3504400-03	29-80371

Table 3 PWBA Table - 16 K Modules (No Validate Storage)

Abbr.	Title	Ampex Part No.	Digital Part No.
DR	Display Register	3280998-01	29-80366
TR	Transceiver	3281609-01*	
CI	Control Interface	3504408-01	
ID	Indicator Driver	3502283-01	
SC	Sector Control	3504402-01	
PC	Port Control	3504400-__+	
TC	Timing & Control	3502924-02	
M	Memory BD	3502915-02	
ECC	ECC	3504406-03	

* Alternate part - 3506195-01
\+ Part number depends on storage size:

Storage	Ampex Part No.
256 K	3504400-01
512 K	3504400-02
768 K or 1024 K	3504400-03

Table 4 PWBA Table - 64 K Modules

Abbr.	Title	Ampex Part No.	Digital Part No.
DR	Display Register	3506766-01	29-80348
TR	Transceiver	3506195-01	29-80350
CI	Control Interface	3506204-01	29-80349
ID	Indicator Driver	3506293-01	29-80351
SC	Sector Control	3506180-01	29-80352
PC	Port Control*	3506206-__+	29-803__+
TC	Timing & Control	3506678-02	29-80356
M	Memory BD*	3506685-02	29-80358
ECC	ECC	3506208-01	29-80359

* Refer to Tables 20 and 21 for external interleave card locations.
\+ Part number depends on storage size:

Storage	Ampex Part No.	Digital Part No.
1024 K	3506206-01	29-80353
2048 K	3506206-02	29-80354
3072 K or 4096 K	3506206-03	29-80355

Equipment Configuration

Assign Unit Starting Address - To set the unit starting address, determine the amount of memory below the Ampex unit; then refer to Table 5 for the value of addresses, bits 14-20. Set UNIT STARTING ADDRESS switches accordingly. Unit Starting Address switches are located on the ARM-10LS Control Panel.

NOTE

After setting the starting address switches, the reset switch must be pressed to properly configure system size and last address boundary.

Table 5 Unit Starting Address

Qty Of Memory Below Unit	Unit Starting Address Switches A14	A15	A16	A17	A18	A19	A20
0 K	0	0	0	0	0	0	0
32 K	0	0	0	0	0	0	1
64 K	0	0	0	0	0	1	0
128 K	0	0	0	0	1	0	0
192 K	0	0	0	0	1	1	0
256 K	0	0	0	1	0	0	0
320 K	0	0	0	1	0	1	0
384 K	0	0	0	1	1	0	0
448 K	0	0	0	1	1	1	0
512 K	0	0	1	0	0	0	0
576 K	0	0	1	0	0	1	0
640 K	0	0	1	0	1	0	0
704 K	0	0	1	0	1	1	0
768 K	0	0	1	1	0	0	0
832 K	0	0	1	1	0	1	0
896 K	0	0	1	1	1	0	0
960 K	0	0	1	1	1	1	0
1024 K through 1984 K	0	1	*	*	*	*	*
2048 K through 3008 K	1	0	*	*	*	*	*
3072 K through 4096 K	1	1	*	*	*	*	*

* Repeat all of above.

Assign Logical Sectors to Physical Sectors - The two Sector Index switches on the ARM-10LS Control Panel determine the assignment of logical sectors to physical sectors. For normal operation, set both switches to the 0 position (down). In this position logical sector numbers correspond to physical sectors, i.e., when the CPU addresses Sector 0, it will select physical Sector 0 in the ARM-10LS. Table 6 lists the logical/physical sector assignments for all combinations of Sector Index switch settings.

Table 6 Sector Index Switch Selections

Sector Selected By CPU	Physical Sector Selected* (Memory Busy Indicator On) SW11	SW10	SW01	SW00
S0	S1	S2	S3	S0
S1	S2	S3	S0	S1
S2	S3	S0	S1	S2
S3	S0	S1	S2	S3

*Sector Display Switches = 00

Establish Interleave Level - Two sets of switches, shown in Figure 8, are used to establish the interleave level of the unit. The INTERLEAVE switch determines the Internal level of interleave. The EXT INTERLEAVE switch and the SELECT (MSB, LSB) switches determine external interleave level.

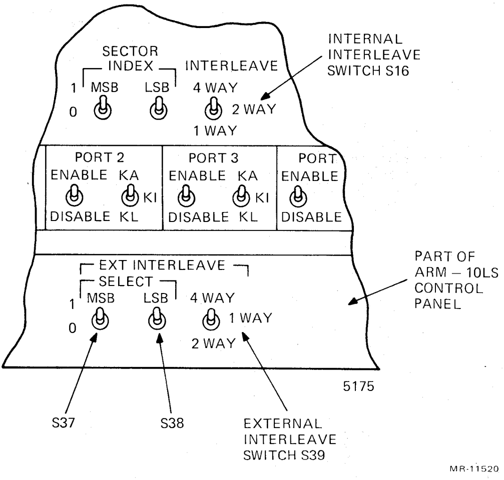


Figure 8 Control Panel Interleave Switch Locations

Internal Interleave Switch Settings - Set the internal INTERLEAVE switch as required by site conditions. (Set to 4-way for external interleave.) The EXT INTERLEAVE switch is set to the center (1-WAY) position for normal operation.

External Interleave Switch Settings - This mode is required for proper operation in multiprocessor applications such as shared multiprocessor (SMP) systems. Set the INTERLEAVE switch to the 4-WAY position.

When the EXT INTERLEAVE switch is set to 2-WAY, the unit memory size displayed at the control panel is twice the unit size. The unit can be operated as unit 0 or unit 1, depending on the position of the MSB SELECT switch.

When the EXT INTERLEAVE switch is set to 4-WAY, the unit memory size displayed at the control panel is four times the unit size. The unit can be operated as unit 0, 1, 2, or 3 depending on the positions of the MSB and LSB SELECT switches. Refer to Table 8 for EXT INTERLEAVE switch settings.

Table 8 External Interleave Switch Positions

Interleave Level	ARM 10LS Unit No.	EXT INTERLEAVE Switch Settings MSB	LSB	INTL SW S39
2-Way External Interleave	0	0	X	2-Way
	1	1	X	2-Way
4-Way External Interleave	0	0	0	4-Way
	1	0	1	4-Way
	2	1	0	4-Way
	3	1	1	4-Way

X = Not Used

NOTE

Set interleave switch (S16) to 4-WAY when using EXT INTERLEAVE. Each ARM-10LS unit must have the same starting address.

Enable Memory Modules and Set Memory Board Select Address - Switch locations are shown and identified in Figures 9 and 10. At each Memory PWBA location in Row A4A, set the enable toggle switch to the down position. Set the thumbwheel switch on each Memory PWBA as required by Table 9.

Note that board selection is not dependent upon slot location; therefore, the installer may set memory board thumbwheel switches in any orderly sequence, as long as sector and storage size requirements are observed. For example, in a 256 K system, four Memory PWBAs are required (one/sector). The Memory PWBAs may be inserted into any memory slot, when there is only one board per sector and the thumbwheel switch is set to 0.

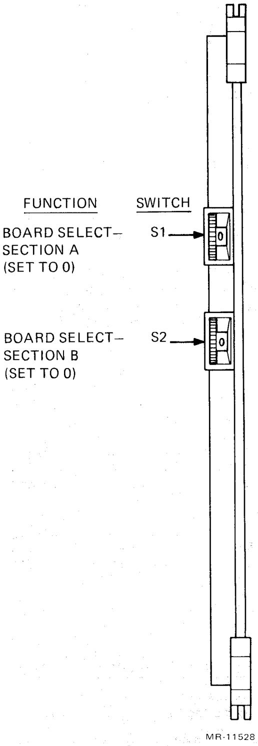


Figure 9 Timing and Control PWBA

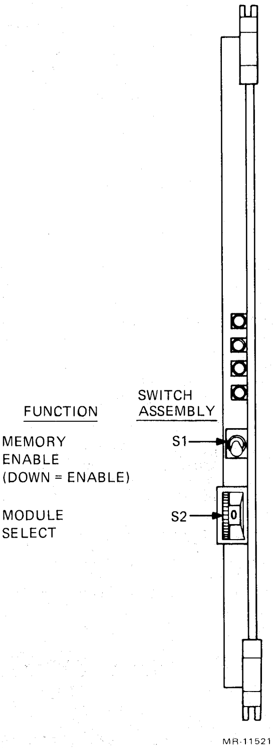


Figure 10 Memory PWBA Showing Switches

Table 9 Memory PWBA Thumbwheel Switch Settings

Unit Storage Size	Sector 0	Sector 1	Sector 2	Sector 3	Thumbwheel Switch Settings
256 K	J103	J108	J116	J121	0
512 K	J104	J109	J117	J122	1
768 K	J105	J110	J118	J123	2
1024 K	J106	J111	J119	J124	3

Set Memory Timing and Control Switch Settings - Memory Timing and Control PWBAs, located at AJ107 and AJ120, have card edge-mounted thumbwheel switches S1 and S2. (See Figures 9 and 10.) Both switches on each card must be set to the number 0.

Set ECC PWBA Switches - Each of the four ECC PWBAs (AJ102, AJ113, AJ115, and AJ126) has four card edge-mounted switches, SW1, SW2, SW3, and SW4. (See Figure 11 for switch locations.) Switches SW3 and SW4 are for maintenance display use. However, switches SW1 and SW2 must be set in the down position for normal operation.

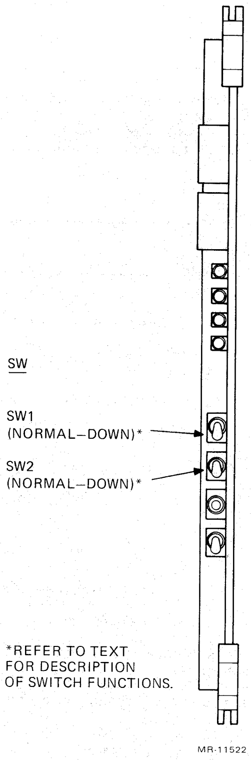


Figure 11   ECC PWBA Switches

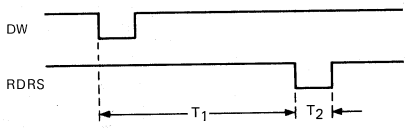
Checks and Adjustments

RDRS Adjustment - Perform the following procedure to adjust the RDRS signal.

1. Make certain that the ARM-10LS address is above the first 20 (octal) system addresses. Run memory diagnostic scope loops, single-word read, in 4-bus mode. Use display panel indicators to determine the port-to-sector address, then move scope probe to the appropriate port for each sector. Observe RDRS and Data Warning signals at the following locations.

Signal	IO Panel (A4D) Location
DATA WARNING (DW)	PIN 7A
READ RESTART (RDRS)	PIN 2A

2. At PWBA locations B105-B108, adjust potentiometers K3 and K4. The timing must be set as shown in Figure 12. Potentiometer K4 is used to adjust pulse width T1, and potentiometer K3 is used to adjust pulse width T2.



T_1 = 290 ±5 NS

T_2 ON KI PROCESSOR = 75–80 NS

T_2 ON KA, KL PROCESSORS = 90–95 NS

MR-11523

Figure 12 RDRS Timing Adjustments

Memory Request Adjustment

1. Load the Ampex diagnostic program and boot the system.

2. Observe the memory system control panel and note any control errors. If control errors are detected, proceed with step 3.; otherwise, continue with normal port testing procedures.

3. Locate potentiometer K8 on the failing Sector Control card.

NOTE

This component is a 22-turn potentiometer. Make certain it is initially full CCW; once the adjustment is made on one Sector Control, go to all other sectors and set potentiometers to the same setting.

4. Turn K8 three turns clockwise.

5. Repeat steps 1 through 5 unless the clockwise turns on K8 are greater than 22. If the number of turns is greater than 22 and control errors are still occurring, excessive electrical noise is present. To correct this condition, a "glitch protect" option may be added to the memory. To install this option, refer to the Glitch Protection Option section.

Glitch Protection Option - To install this option, proceed as follows.

1. Refer to Figure 13 for locations of backpanel pins for this option.

2. Turn off memory system power.

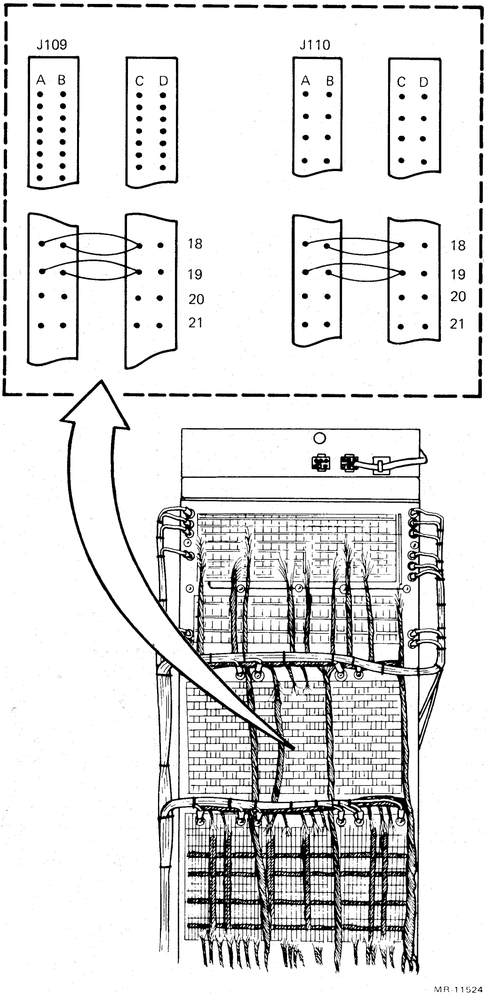


Figure 13 Glitch Protection Jumpers

3. Use wirewrap tool and 30 AWG Kaynar wire to make the following backpanel connections.

	From Conn.	Pin	To Conn.	Pin
PORT 0	BJ109	18A	BJ109	18C
PORT 1	BJ109	18B	BJ109	18C
PORT 2	BJ109	19A	BJ109	19C
PORT 3	BJ109	19B	BJ109	19C
PORT 4	BJ110	18A	BJ110	18C
PORT 5	BJ110	18B	BJ110	18C
PORT 6	BJ110	19A	BJ110	19C
PORT 7	BJ110	19B	BJ110	19C

4. Turn potentiometer K8 30 turns counterclockwise.

5. Continue with normal port testing procedures.

OPERATION

General Information

This section provides operating instructions for the ARM-10LS memory. When operating in a data processing system the memory is controlled by the CPU. Manual operation is generally limited to turning power on and off, and establishing the desired operating modes before on-line operation. Once the unit is on-line, further operator intervention should not be requried.

Controls and Indicators

Operator controls and status indicators are located on the EPO panel, blower assembly, control panel, and memory, memory timing and control, and ECC PWBAs. The following paragraphs describe these controls and indicators.

EPO Panel - Power for the memory is controlled at the EPO panel. The controls consist of switches that select remote or local power sequencing, activate and deactivate the memory, and control power supply operation. Related indicators provide a visual indication of the power status and the individual power supply voltage margins, and any other abnormal operating conditions. EPO panel controls and indicators are described in Table 10 and shown in Figure 14.

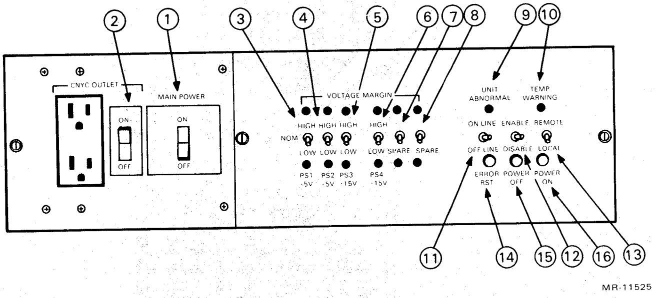


Figure 14 EPO Panel Controls and Indicators

Table 10 EPO Controls and Indicators

Fig. 14 Ref. No.	Control/Indicator Name	Type	Reference Designation	Function
1	MAIN POWER	Circuit Breaker Indicator	CB2	Controls ac input power to memory; provides automatic overload protection for the ac input circuit. Illuminates (white) when active.
2	CNVC OUTLET	Circuit Breaker Indicator	CB1	Controls ac input power to convenience outlet (J1); this outlet is intended to provide voltage for external test equipment used by service personnel. Illuminates (white) when active.
3	PS1(+5 V) VOLTAGE MARGIN	Toggle Switch (3-position)	S6	When in up position margins power supply 1 (+5 V) high. When in down position, margins power supply 1 low. Should be in center (NOM) position for normal operation.
	HIGH	LED (red) (above PS1 toggle switch)	DS1	Illuminates when PS1 is in high margin condition.
	LOW	LED (red) (below PS1 toggle switch	DS9	Illuminates when PS1 is in low margin condition.
4	PS2(-5.7 V) VOLTAGE MARGIN	Toggle Switch (3-position)	S7	Not used. Power supply 2 (-5.7 V) cannot be margined.
	HIGH	LED (red) (above PS2 toggle switch)	DS2	
	LOW	LED (red) (below PS2 toggle switch)	DS10	
5	PS3(-5.2 V) VOLTAGE MARGIN	Toggle Switch (3-position)	S8	When in up position, margins power supply 3 (-5.2 V) high. When in down position, margins power supply 3 low. Should be in center (NOM) positions for normal operation.
	HIGH	LED (red) (above PS3 toggle switch)	DS3	Illuminates when PS3 is in high margin condition.
	LOW	LED (red) (below PS3 toggle switch)	DS11	Illuminates when PS3 is in low margin condition.

Table 10 EPO Controls and Indicators (Cont)

Fig. 14 Ref. No.	Control/Indicator Name	Type	Reference Designation	Function
6	PS4 (+5 V) VOLTAGE MARGIN	Toggle Switch (3-position)	S9	When in up position, margins power supply 4 (+5 V) high. When in down position, margins power supply 4 low. Should be in center (NOM) position for normal operation.
	HIGH	LED (red) (above PS4 toggle switch)	DS4	Illuminates when PS4 is in high margin condition.
	LOW	LED (red) (below PS4 toggle switch)	DS12	Illuminates when PS4 is in low margin condition.
7	PS5(+12 V) VOLTAGE MARGIN	Toggle Switch (3-position)	S10	When in up position, margins power supply 5 (+12 V) high. When in down position, margins power supply 5 low. Should be in center (NOM) position for normal operation.
	HIGH	LED (red) (above PS5 toggle switch)	DS5	Illuminates when PS5 is in high margin condition.
	LOW	LED (red) (below PS toggle switch)	DS13	Illuminates when PS5 is in low margin condition.
8	PS6 VOLTAGE MARGIN	Toggle Switch	S11	Not Used
	HIGH	LED (red)	DS6	
	LOW	LED (red)	DS14	
9	UNIT ABNORMAL	LED (red)	DS7	Illuminates when a power supply is in high or low margin condition; also indicates that ac power is below required operating range.
10	TEMP WARNING	LED (red)	DS8	Not Used
11	ON-LINE/ OFF-LINE	Toggle Switch (2-position)	S12	Not Used
12	ENABLE/ DISABLE	Toggle Switch (2-position)	S13	Not Used
13	REMOTE/ LOCAL	Toggle Switch (2-position)	S14	Must be set to LOCAL position. NOTE If switch is in REMOTE position, memory power-on is inhibited.

Table 10 EPO Controls and Indicators (Cont)

Fig. 14 Ref. No.	Control/Indicator Name	Type	Reference Designation	Function
14	ERROR RST	Pushbutton Switch (white)	S15	Not Used
15	POWER OFF	Momentary Pushbutton Switch (red)	S16	Removes all dc power to the memory and ac power to the blower assemblies.
16	POWER ON	Momentary Pushbutton Switch (green)	S17	Applies ac power to the memory and ac power to the blowers. Should be held for 3 seconds when applying power.

Blower Assembly - A circuit breaker is located at the rear of the blower assembly, as illustrated in Figure 15. If the circuit breaker is tripped, press the center button to restore power. Note that if the breaker is tripped, power will be removed from the entire memory unit.

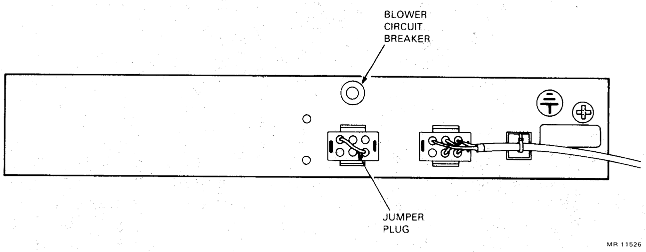


Figure 15 Blower Assembly Circuit Breaker Location

Control Panel - Controls and indicators located on the Control Panel (Figure 16) are functionally divided into Control, Error Status, and Maintenance Groups. Each group is described in the following paragraphs.

Control Group - The Control Group establishes memory operating parameters. Included are switches for enabling each of the ports, setting address boundaries, and establishing memory request type and interleave mode.

a. Port and Sector Enable switches. The three types of switches are described in Table 11.

b. Unit Starting Address. Seven switches are used to establish the memory starting address. The address may be set on 64 K boundaries, depending on the quantity of memory below the unit. Table 5 indicates the positions for this parameter.

c. Next Starting Address. Seven LED indicators display the starting address of the next unit. If the External Interleave switch is set to 1-WAY, the indicators display the ARM-10LS last address plus 1 (maximum storage size plus 1). If the External Interleave switch is set to 2-WAY, the next starting address indicators display twice the storage size plus 1. Four times the actual storage size plus 1 is displayed if the External Interleave switch is set to 4-WAY. When internal interleave (2-WAY or 4-WAY) is active, the next starting address indicators are not affected. The address range reflects system capacity only.

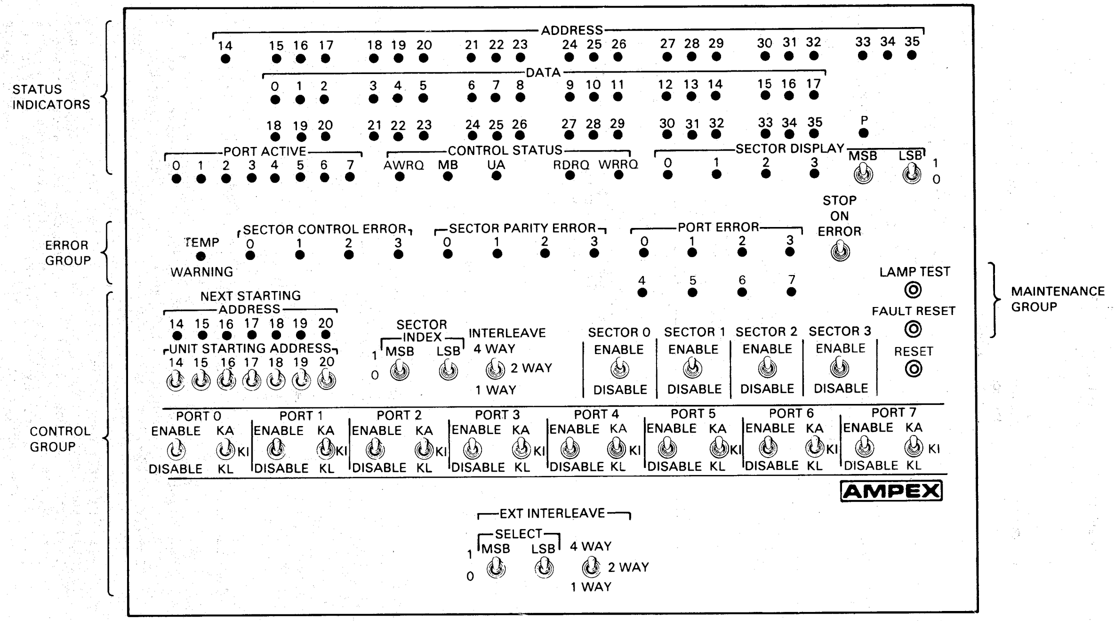


Figure 16 Main Control Panel

MR-11527

d. Sector Index. Two switches are used to determine the order in which sectors will be selected. Table 12 shows the sector index switch settings.

e. Interleave. A 3-position toggle switch is used to select the Memory Interleave mode. The 1-WAY position selects noninterleaved addressing. In the 2-WAY position, Sector 0 is interleaved with Sector 1, and Sector 2 is interleaved with Sector 3. In the 4-WAY position, all four sectors can be simultaneously addressed.

f. Reset. A single pushbutton switch that initializes the memory unit. When pressed, RESET presets control flip-flops in 6400S Memory, resets error latches in ECC PWBAs, and resets port request latches in all Sector Control PWBAs.

Every time RESET is pressed, circuitry in the Control Interface PWBA automatically configures the system memory capacity (last address) and the next unit starting address is displayed on the control panel indicators.

NOTE

Do not press reset during system operation.

Table 11 Port and Sector Enable Switches

Name	Type	Function
PORT ENABLE/DISABLE	2-position Toggle Switch	Places associated memory port on-line or off-line. In the ENABLE position, port is on-line (connected to CPU or channel); in DISABLE position, port is switched off-line.
PORT KA/KI/KL	3-position Toggle Switch	Determines the type of request to which the memory will respond. (Switches are set at the time of installation.)
SECTOR ENABLE/DISABLE	2-position Toggle Switch	Enables or disables associated sector. In ENABLE position, physical sector can be accessed by CPU. In DISABLE position, CPU access to sector is blocked.

Table 12 Sector Index Addressing

Memory Unit Sector	Sector Display Switches MSB	LSB	Sector Index Switches MSB	LSB	Sector Display Indicator ON	Electrical Address (CPU Sector Select)	Physical Address (Memory Sector)
0	0	0	0	0	0	S0	S0
	0	0	0	1	1	S1	S0
	0	0	1	0	2	S2	S0
	0	0	1	1	3	S3	S0
1	0	1	0	0	1	S1	S1
	0	1	0	1	2	S2	S1
	0	1	1	0	3	S3	S1
	0	1	1	1	0	S0	S1
2	1	0	0	0	2	S2	S2
	1	0	0	1	3	S3	S2
	1	0	1	0	0	S0	S2
	1	0	1	1	1	S1	S2
3	1	1	0	0	3	S3	S3
	1	1	0	1	0	S0	S3
	1	1	1	0	1	S1	S3
	1	1	1	1	2	S2	S3

Note that sector electrical bit assignment is a function of memory size and interleave settings.

g. External Interleave Switches. A 3-position toggle switch (S39) which sets external interleave level in multiunit installations. This switch controls SELECT switches MSB and LSB. In the 4-WAY position, both MSB and LSB are enabled. In the 2-WAY position, only MSB is enabled. In the 1-WAY position, both MSB and LSB are disabled (external interleave disabled).

h. SELECT MSB, LSB. Two 2-position toggle switches that determine the order or unit selection in a multiunit installation. The settings for these switches, in conjunction with EXT INTERLEAVE (switch S39), are defined in Table 8.

Error Group - The Error Group permits the operator to locate control and port errors.

a. STOP ON ERROR toggle switch. The 2-position STOP ON ERROR switch has the following functions.

   Up Position. If a Sector Control, Sector Parity, or Port Error is detected, the memory ceases operation; indicators display memory status at the time of error detection.

   Down Position. Memory continues operation under control of the CPU. Indicators accumulate and display errors.

b. FAULT RESET pushbutton switch. When pressed, FAULT RESET clears all control panel error displays (Sector Control, Sector Parity, and Port Error). In addition, the Fault Reset Function clears error displays on the four ECC PWBAs including UE, CE, DOPE, DIPE, and Card/Chip indicators. This switch may be pressed during system operation (also refer to the Control Group section).

c. Sector Control Error Indicators. During a Write or Read-Modify-Write cycle, if a Write Restart signal has not been received from the CPU within 25 microseconds, a control error occurs, and the Sector indicator lights. Control error indicators also light whenever an invalid request is detected. The condition also occurs when a cycle (Read or Write) has not been completed within 24 microseconds.

d. Sector Parity Error Indicators. A SECTOR PARITY ERROR indicator lights to identify the sector in which a Read or Write parity error has occurred. It should be noted that sector identification relates to physical locations, and is not influenced by interleaving or Sector Index addressing.

e. Port Error Indicators. These indicators identify I/O ports in which control or parity errors occur during a Write or Read-Modify-Write cycle. If the STOP ON ERROR switch is in the down position, accumulated errors will be displayed.

Refer to Table 13 for information on sector control error, sector parity error, and port error LED indicators.

Table 13 LED Error Indicators

Error Type	Sector Control Error	Sector Parity Error	Port Error
Read Error	LED OFF	LED ON	LED OFF
Write Error	LED OFF	LED ON	LED ON
Control Error	LED ON	LED OFF	LED ON

Status Group - The Status Group provides visual indicators relating to memory operations.

a. Sector Display. The Status Group indicators (Port Active, Control Status, and Address and Data) are shared by the four Memory Sectors. The status of only one sector can be displayed at a time. To view the status of any sector, the SECTOR DISPLAY switches must be set to the binary number representing the Sector. The corresponding SECTOR DISPLAY indicator will light.

b. Port Active. Illuminated LEDs in the port active group indicate operating ports.

c. Control Status. Status Control indicators described in Table 14 are used to monitor operating mode and status of the memory.

d. Address and Data. These status indicators provide a continuously updated display of transceiver address register and data register contents. An illuminated lamp indicates that the register contains a logical 1.

NOTE
The memory unit address is "normalized" to a number ranging from 0 to the value of the last address (256 K, 512 K, 768 K, or 1024 K). Address bits 14-17 on the control panel are not used. The selected physical sector is indicated by sector display indicators. Address bits 18, 19 always display the selected memory module.

Table 14 Control Status Indicators

Control/Indicator Name	Type	Function	Operation
AWRQ	LED	Monitors Sector Await Request Memory Status.	Lights to indicate interface is not busy and is awaiting an access request from one of the memory ports. Extinguishes when interface is busy. When sector is busy servicing a port, all other ports are locked out.
MB	LED	Monitors Memory Busy from Sector.	Lights to indicate Memory Sector is busy with a cycle. Always dimly lit due to refresh cycle.
UA	LED	Monitors Unit Available Status of Memory Sector.	Lights to indicate that sector is available to execute a cycle. Extinguishes when a sector is busy with a cycle, or has completed the Read portion of a Read-Modify-Write cycle and is awaiting WRRS (Write Restart).
RDRQ+	LED	Monitors Read Request input line.	Lights to indicate a read cycle requested.
WRRQ+	LED	Monitors Write Request.	Lights to indicate a write cycle is requested.

+If both RDRQ and WRRQ indicators are illuminated, the unit is in Read-Modify-Write mode.

Maintenance Group - The Maintenance Group consists of the LAMP TEST switch and the TEMP WARNING indicator. These are shown in Figure 16 and are described in Table 15.

Memory PWBA - Two switches and four indicators are located on each memory PWBA. These components are illustrated in Figure 17 (and Figure 10) and described in Table 16.

Table 15 Maintenance Controls and Indicators

Control/Indicator Name	Type	Function	Operation
LAMP TEST	Toggle Switch	Checks for faulty panel indicators.	Press this switch to test all indicators. Indicators that do not light are faulty.
TEMP WARNING	LED	Indicates over-temperature condition. Temperature within cabinet is 136^{o}F or greater.	Illuminates if overheating occurs. Extinguishes when temperature returns to normal.

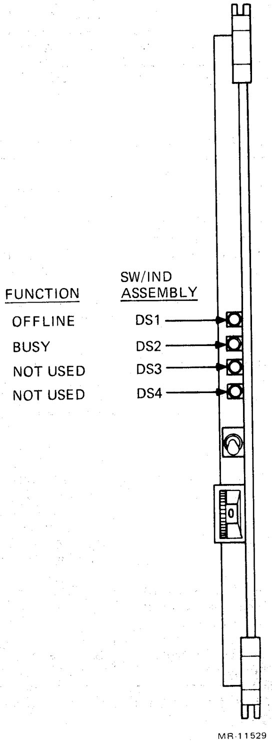


Figure 17 Memory PWBA Showing Lights

Table 16 Memory PWBA Controls and Indicators

Control/Indicator Name	Type	Reference Designation	Function
OFF LINE	LED (yellow)	DS1	Illuminates when Memory PWBA is disabled.
BUSY	LED (green)	DS2	Illuminates when Memory PWBA is performing Read or Write cycles.
MULTIBIT ERROR	LED (red)	DS3	Not Used.
SINGLE BIT ERROR	LED (red)	DS4	Not Used.
ENABLE	Toggle Switch	S1	In the down position, enables the Memory PWBA. In the up position, places the Memory PWBA off-line (disabled).
MODULE SELECT	Thumb-wheel Switch	S2	Provides four board select control settings (0-3). PWBA is selected when address line inputs match hex output of S2. Memory PWBAs in each sector must be set as follows. **Switch Setting** / **Memory PWBA Selected** 0 — 1st 64K (64K/Sector) 1 — 2nd 64K (128K/Sector) 2 — 3rd 64K (192K/Sector) 3 — 4th 64K (256K/Sector)

Memory Timing and Control PWBA - The Memory Timing and Control board shown in Figure 9 has two hexadecimal thumbwheels switches, S1 and S2. Each switch must be set to the "0" position.

ECC PWBA - Six indicators, visible through slots in the memory cover panel, reflect error conditions in the memory. ECC PWBA indicators and related controls are shown in Figure 18 and described in Table 17. Note that any detected error causes the indicators to light. From combinations of error indications (latched at the time of error), the following types of information can be determined.

1. Type of error: read data error or write data error.

2. Type of read error: single-bit (corrected) or double-bit (uncorrected).

3. Location of error: sector, card, chip group, and data bit number.

NOTE

The Chip Group is one of four physical groups of 43 memory chips located on the memory PWBA. Refer to the MEMORY CHIP FAULT ISOLATION section for the memory chip fault isolation procedure.

Power-On Procedure - Power-on sequencing is performed internally, eliminating the need for special precautions. Since power is applied to memory from the memory EPO panel, the REMOTE/LOCAL switch must be set to the LOCAL position. To power up the memory, set the MAIN POWER circuit breaker to ON and press the POWER ON pushbutton switch for approximately three seconds. The blower fans should operate, indicating that power is applied to the unit. Also check that VOLTAGE MARGIN, UNIT ABNORMAL, and TEMP WARNING indicators are extinguished.

On-Line/Off-Line Operation

For normal operation, set port ENABLE/DISABLE switches to ENABLE. To deselect a memory unit, set the port ENABLE/DISABLE switches to DISABLE.

Power-Off Procedure
Power may be removed at the memory EPO panel by pressing the POWER OFF pushbutton switch on the EPO panel.

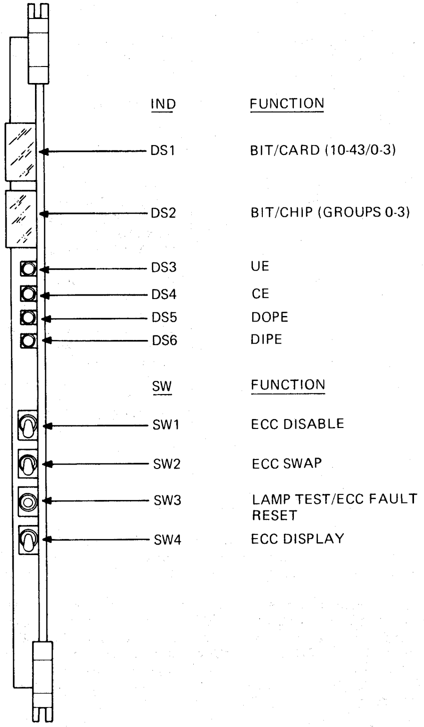


Figure 18 ECC PWBA Controls and Indicators

Table 17 ECC PWBA Controls and Indicators

Control/Indicator Name	Type	Reference Designation	Function
ECC DISABLE	Toggle switch (2-position)	SW1	Up position: Disables ECC (error checking and correction) Memory stores and checks single odd parity bit for entire data word. Down position: Enables ECC function. Memory corrects and reports single-bit data errors and reports multibit errors.
ECC SWAP	Toggle switch (2-position)	SW2	Swaps bits 29-35 with bits 36-42 so ECC bits can be checked. Switch is enabled in the up position for maintenance purposes only. This function is operational when SW1 (ECC DISABLE) is in the up position.
LAMP TEST/ ECC FAULT RESET	Push-button (momentary)	SW3	When pressed momentarily, lights all ECC LED indicators and clears ECC errors. NOTE: Switch S3 may be pressed when the system is running.
ECC Display	Toggle (2-position)	SW4	Controls hex displays DS1 and DS2. Up position: Causes DS1, DS2 to display failing Memory PWBA and failing 16 K chip group. Memory PWBA number is read on DS1 while 16 K chip group number is read on DS2. Down position: Causes DS1, DS2 to display Memory PWBA failing data bit number. MSD is read on DS1 and LSD is read on DS2.
Bit/Card	Hex Display	DS1	When SW1 is down and SW4 is up, DS1 displays the failing Memory PWBA as a decimal number 0-3. (Corresponds to module select number on Memory PWBA switch S2.)
Bit/Card	Hex Display	DS1	When SW1 is down and SW5 is down, DS1 displays the most significant decimal digit of the failing single data bit (MSD will be a number 0-4). If the failure is a multibit error, DS1 displays the hex digit "F".
Bit/Chip	Hex Display	DS2	Used in conjunction with SW1 and SW4 to display failing 16 K chip group, depending on position of SW4. When SW1 is down and SW4 is up, DS2 displays the failing 16 K group as a decimal number 0-3. When SW1 is down and SW4 is up, DS2 displays the failing 16 K group as a decimal number 0-3.

Table 17 ECC PWBA Controls and Indicators (Cont)

Control/Indicator Name	Type	Reference Designation	Function
			When SW1 is down and SW4 is down, DS2 displays the least significant decimal digit of the failing single data bit (LSD will be a number 0-9). If the failure is a multibit error, DS2 displays the hex digit "F". If the decimal point on DS2 is always ON, the indicated error is a "hard" failure. DS2 also blinks on any ECC error (both correctable and uncorrectable).
UE	LED (red)	DS3	Illuminates when a multibit (uncorrectable) error has been detected in data read from the Memory PWBA. Is cleared by pressing the Control Panel FAULT RESET or ECC TEST/FAULT RESET pushbutton. Indicator is active only when SW1 is down (ECC enabled.)
CE	LED (red)	DS4	Illuminates when a single-bit (correctable) data error has been detected in data read from the Memory PWBA. Is cleared by Control Panel FAULT RESET or ECC LAMP TEST/FAULT RESET pushbutton. Indicator is active only when SW1 is in down position (ECC enabled).
DOPE	LED (red)	DS5	Illuminates when bad parity is detected in read data (Data Out). Indicator is active only when SW1 is in the up position (ECC disabled). Indicator DS5 is cleared by pressing Control Panel FAULT RESET or ECC LAMP TEST/FAULT RESET pushbutton.
DIPE	LED	DS6	Illuminates when bad parity is detected in write data (data to memory). Indicator DS6 is cleared by pressing Control Panel FAULT RESET or ECC LAMP TEST/FAULT RESET pushbutton.

MEMORY CHIP FAULT ISOLATION

If a failure can be isolated to one data bit on a memory PWBA, the failing memory chip can be located and replaced. To isolate a failing memory chip, note the status of address bits 20 and 21 at the time of failure; then refer to Figure 19. The 172 memory chips are partitioned into four 16 K X 43 addressable blocks corresponding to the status of address bits 20 and 21.

Address Bit

20	21	
H	H	1st 16 K
H	L	2nd 16 K
L	H	3rd 16 K
L	L	4th 16 K

H - High Logic Level (Logical 0)
L - Low Logic Level (Logical 1)

Locate chip at the intersection of data bit and 16 K block; then remove and replace chip, as follows.

CAUTION
Memory ICs are MOS devices, which can be damaged by static electric charges.

1. Set PWBA on a flat surface and place one hand on PWBA.

2. Replacement memory ICs are normally packaged in a block of conductive foam; place the foam block on the PWBA.

3. Keep one hand in contact with the PWBA; remove and replace memory chip with free hand.

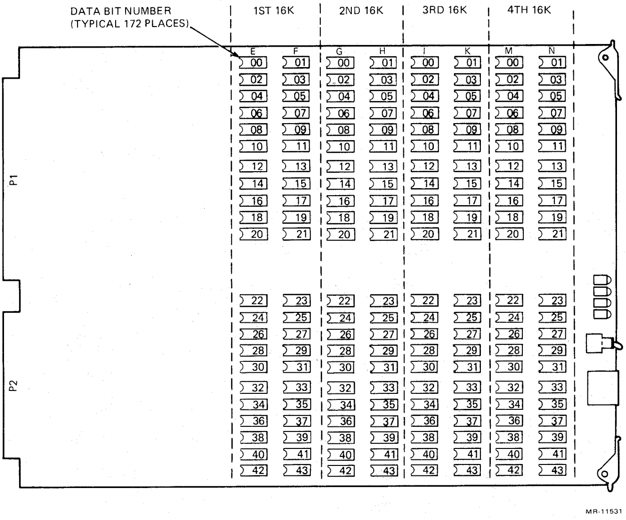


Figure 19 Memory PWBA Chip Locations

INSTALLATION REFERENCE INFORMATION
This section provides information that may be used in conjunction with memory system installation tasks.

Table 18 Memory Address Assignments

Interleave Level	Box Size	Unit Select	Sector Select	Module Select	Block Select	Intern Addr.
ONE-WAY EXTERNAL (NO INTERLEAVE)	1 M	14,15	16,17	18,19	20,21	22-35
	768 K	14,15	16,17	18,19	20,21	22-35
	512 K	14,15	16,17	18,19	20,21	22-35
	256 K	14-17	18,19	-	20,21	22-35
ONE-WAY EXTERNAL or TWO-WAY INTERNAL	1 M	14,15	16,35	17,18	20,21	22-34,19
	768 K	14,15	16,35	17,18	20,21	22-34,19
	512 K	14,16	17,35	18	20,21	22-34,19
	256 K	14-17	18,35	-	20,21	22-34,19
ONE-WAY EXTERNAL or FOUR-WAY INTERNAL	1 M	14,15	34,35	16,17	20,21	22-34,18,19
	768 K	14,15	34,35	16,17	20,21	22-33,18,19
	512 K	14-16	34,35	17	20,21	22-33,18,19
	256 K	14-17	34,35	-	20,21	22-33,18,19
TWO-WAY EXTERNAL or FOUR-WAY INTERNAL	1 M	14,34	15,35	16,17	20,21	22-33,18,19
	768 K	14,34	15,35	16,17	20,21	22-33,18,19
	512 K	14,15,34	16,35	17	20,21	22-33,18,19
	256 K	14-16,34	17,35	-	20,21	22-33,18,19
FOUR-WAY EXTERNAL or FOUR-WAY INTERNAL	1 M	34,35	14,15	16,17	20,21	22-33,18,19
	768 K	34,35	14,15	16,17	20,21	22-33,18,19
	512 K	14,34,35	15,16	17	20,21	22-33,18,19
	256 K	14,15 34,35	16,17	-	20,21	22-33,18,19

Address example: 01410010 (1 M No Interleave)

- -	15 0	18 1	21 0	24 0	27 0	30 0	33 0
- -	16 0	19 0	22 0	25 0	28 0	31 0	34 0
14 0	17 1	20 0	23 1	26 0	29 0	32 1	35 0
0	1	4	1	0	0	1	0

Bits 17, 18, 23, and 32 were on, therefore sector 1, module 2, chip 0 failed.

Table 19 Octal Storage Barrier Addresses
One Megabyte by 16 K Increments

OCT	DEC	OCT	DEC
0000000-0037777	00K- 16K	2000000-2037777	512K- 528K
0040000-0077777	16K- 32K	2040000-2077777	528K- 544K
0100000-0137777	32K- 48K	2100000-2137777	544K- 560K
0140000-0177777	48K- 64K	2140000-2177777	560K- 576K
0200000-0237777	64K- 80K	2200000-2237777	576K- 592K
0240000-0277777	80K- 96K	2240000-2277777	592K- 608K
0300000-0337777	96K-112K	2300000-2337777	608K- 624K
0340000-0377777	112K-128K	2340000-2377777	624K- 640K
0400000-0437777	128K-144K	2400000-2437777	640K- 656K
0440000-0477777	144K-160K	2440000-2477777	656K- 672K
0500000-0537777	160K-176K	2500000-2537777	672K- 688K
0540000-0577777	176K-192K	2540000-2577777	688K- 704K
0600000-0637777	192K-208K	2600000-2637777	704K- 720K
0640000-0677777	208K-224K	2640000-2677777	720K- 736K
0700000-0737777	224K-240K	2700000-2737777	736K- 752K
0740000-0777777	240K-256K	2740000-2777777	752K- 768K
1000000-1037777	256K-272K	3000000-3037777	768K- 784K
1040000-1077777	272K-288K	3040000-3077777	784K- 800K
1100000-1137777	288K-304K	3100000-3137777	800K- 816K
1140000-1177777	304K-320K	3140000-3177777	816K- 832K
1200000-1237777	320K-336K	3200000-3237777	832K- 848K
1240000-1277777	336K-352K	3240000-3277777	848K- 864K
1300000-1337777	352K-368K	3300000-3337777	864K- 880K
1340000-1377777	368K-384K	3340000-3377777	880K- 896K
1400000-1437777	384K-400K	3400000-3437777	896K- 912K
1440000-1477777	400K-416K	3440000-3477777	912K- 928K
1500000-1537777	416K-432K	3500000-3537777	928K- 944K
1540000-1577777	432K-448K	3540000-3577777	944K- 960K
1600000-1637777	448K-464K	3600000-3637777	960K- 976K
1640000-1677777	464K-400K	3640000-3677777	967K- 992K
1700000-1737777	480K-496K	3700000-3737777	992K-1008K
1740000-1777777	496K-512K	3740000-3777777	1008K-1024K

Table 20 Typical Memory PWBA Locations for Internal Interleave (One Box)

Interleave Level	Port Control PWBA Version	Box Size	S3 M3 AJ124	M2 J123	M1 J122	M0 J121	S2 M3 AJ119	M2 J118	M1 J117	M0 J116	S1 M3 AJ111	M2 J110	M1 J109	M0 J108	S0 M3 AJ106	M2 J105	M1 J104	M0 J103
Internal	3506206-01	1 M				X				X				X				X
	3506206-02	2 M			X	X			X	X			X	X			X	X
1-WAY	3506206-03	3 M					X	X	X	X	X	X	X	X	X	X	X	X
	3506206-03	4 M	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Internal	3506206-01	1 M				X				X				X				X
	3506206-02	2 M			X	X			X	X			X	X			X	X
2-WAY	3506206-03	3 M			X	X			X	X	X	X	X	X	X	X	X	X
	3506206-03	4 M	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Internal	3506206-01	1 M				X				X				X				X
	3506206-02	2 M			X	X			X	X			X	X			X	X
4-WAY	3506206-03	3 M		X	X	X		X	X	X		X	X	X		X	X	X
	3506206-03	4 M	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

NOTES: 1. BOX SIZE is memory capacity per ARM-10LS cabinet.
2. X indicates where Memory PWBA is to be installed for the corresponding BOX SIZE.
3. Memory locations for 3 M size vary depending on interleave level.

Table 21 Typical Memory PWBA Locations for External 2-Way Interleave (Two Box)

Box Number	Port Control PWBA Version	Box Size	S3	S2	S1 M3 AJ111	S1 M2 J110	S1 M1 J109	S1 M0 J108	S0 M3 AJ106	S0 M2 J105	S0 M1 J104	S0 M0 J103
BOX 0	3506206-01	512 K						X				X
	3506206-02	1 M					X	X			X	X
	3506206-03	1.5 M				X	X	X		X	X	X
	3506206-03	2 M			X	X	X	X	X	X	X	X
BOX 1	3506206-01	512 K						X				X
	3506206-02	1 M					X	X			X	X
	3506206-03	1.5 M				X	X	X		X	X	X
	3506206-03	2 M			X	X	X	X	X	X	X	X

NOTES:
1. Sectors 2 and 3 of each box not used.
2. X indicates where memory PWBA is installed for corresponding BOX SIZE.
3. 512 K and 1.5 M box sizes are nonstandard.

Table 22 Typical Memory PWBA Locations for External 4-Way Interleave (Four Box)

Box Number	Port Control PWBA Version	Box Size	S3	S2	S1	S0 M3 AJ106	M2 J105	M1 J104	M0 J103
BOX 0	3506206-01	256 K							X
	3506206-02	512 K						X	X
	3506206-03	768 K					X	X	X
	3506206-03	1 M				X	X	X	X
BOX 1	3506206-01	256 K							X
	3506206-02	512 K						X	X
	3506206-03	768 K					X	X	X
	3506206-03	1 M				X	X	X	X
BOX 2	3506206-01	256 K							X
	3506206-02	512 K						X	X
	3506206-03	768 K					X	X	X
	3506206-03	1 M				X	X	X	X
BOX 3	3506206-01	256 K							X
	3506206-02	512 K						X	X
	3506206-03	768 K					X	X	X
	3506206-03	1 M				X	X	X	X

NOTES:
1. Sectors 1, 2, and 3 of each box not used.
2. X indicates where Memory PWBA is installed for corresponding BOX SIZE.

Maintenance Software

Table of Contents

GENERAL INFORMATION

Abstract

DIACON is the executive controller which is assembled with each 11-based 10 diagnostic.

Notes

Console commands may be executed directly from DIACON command mode. If a naming conflict occurs, a period preceding the command will ensure that the console will act on it.

The sync point for scope loops is generated by DIACON at A36E1 on the CPU backplane.

Loading and Starting Procedure

DIACON is automatically loaded as part of each 11-based 10 diagnostic.

OPERATIONAL CONTROL

DIACON has two modes of operation. Normal operation (control switch 1 reset) is transparent to the user. In this mode DIACON runs the diagnostic, performs fault convergence or cataloguing, reports fault symptoms, loads and runs TIC files and/or isolation routines, and prints the most probable cause (board callout), all without operator intervention.

The second mode of operation (control switch 1 set) enables a set of commands which permit user intervention. These commands are described in Table 1. The commands described in Table 2 are only supported if they are listed by the /H command.

Table 1 DIACON Command Summary

Command	Description
.	Commands beginning with a period are passed to KLDCP for execution.
H	H<CR> Print a list of the DIACON commands currently in effect.
HE	HE<CR> Print a list of all commands supported by DIACON.
/H	/H<CR> Prints a list of special switches defined by the diagnostic programmer. User switches are supported only if the programmer specified this option. Refer to Table 2.
↑C	Control C interrupts the execution of the diagnostic. Control is returned to KLDCP.
HC	HC<CR> The HC command continues from an error halt.
$	$ES 0<CR> Altmode interrupts the execution of the diagnostic for one KLDCP command line.
TS	TS 16<CR> The TS (Test Start) command starts the diagnostic beginning at the test number specified.

Table 1 DIACON Command Summary (Cont)

Command	Description
TL	TL 14,37<CR> The TL (Test Loop) command loops between the first test number and the second test number. A carriage return instead of a second test number will cause the first test specified to be looped on. The abort switch (15) will return control to the console. The TL command can be used to report the first error in every test. This can be done by setting the print all errors switch (07) and the abort switch (15) and specifying a test range of the entire diagnostic,.
PS	PS<CR> The PS (Print Symptoms) command causes DIACON to report unreported errors or to repeat its last error report. Calling of isolation routines is also permitted. No symptom will be printed if the test has been restarted or no fault has occurred.

Table 2 DIACON User Implemented Command Summary

Command	Description
FB	FB<CR> The FB (set Function Breakpoint) command solicits a diagnostic function, bit, and polarity which, if detected, will cause a break to occur. Only one function breakpoint is permitted at a time.
FC	FC<CR> The FC (Function breakpoint Continue) command restores the PDP-11 registers and continues from the last function breakpoint.
RB	RB(CR> The RB (Remove Breakpoint) command removes the function breakpoint.
RG	RG<CR> The RG (print PDP-11 registers) command prints the contents of R0 through R7 saved at the last function breakpoint. This command is primarily for use in debugging programs.

GENERAL INFORMATION

Code	DGQDA.BIN
Title	DECSYSTEM Diagnostic Console Program
Abstract	KLDCP resides in the console front-end processor and supports KL10 based systems at the following three levels.

1. At the console level, KLDCP supports KL10 and PDP-11/40 console functions.
2. At diagnostic run time, KLDCP loads, starts, and provides subroutine services for 11-based 10 and 10-based 10 diagnostic and utility routines.
3. At the timesharing level, KLDCP provides an interface between the CTY or KLINIK terminal and the TOPS-10 or TOPS-20 monitor.

Note	The DTE20 must be in privileged mode.
Loading and Starting Procedure	Refer to the 11/10 STD module.

OPERATIONAL CONTROL

KLDCP is controlled via commands entered at the CTY or KLINIK terminal. The commands consist of two or three characters followed by one or more arguments. The conventions used to illustrate the KLDCP commands are described in Table 3. The commands supported by KLDCP are described in Table 4.

Table 3 KLDCP Command Conventions

Convention	Description
adr	An octal address
data	An octal data field
file.ext	Any legal file name from one to six characters followed by a dot and an extension of zero to three characters
<CR>	Standard command string terminator
$	When used to terminate a P command, the $ (altmode) will cause the file specified to be loaded and started.
#	An octal argument
:	Separates the address and data fields in examine and deposit commands
?	Precedes error message printouts
C	Control C aborts program; returns control to KLDCP from 10 memory.
↑T	Control T must be used as a terminator for commands that are to be interpreted by programs running in the KL10.
↑X	Control X interrupts the program running in 10 memory for one KLDCP command.
;	When a semicolon precedes local comment, the text following it is only printed on the terminal. Messages are sent between the CTY and KLINIK terminal using the semicolons.

Table 4 KLDCP Command Summary

Command	Description
General Commands	
R	R MR, EX inst, PL15<CR> Repeat commands following. Inhibit machine-state printouts.
RP	RP MR, EX inst, PL15<CR> Repeat commands following. Do not inhibit data printout.
TD #	R MR, EX inst, PL10, TD5, PL5<CR> Perform specified (#) time delay.
TF #	TF 0<CR> Set terminal fill count. 0 - 110 baud 3 - 600 baud 1 - 150 baud 4 - 1200 baud 2 - 300 baud 5 - 2400 baud
TW #	TW 132<CR> Set terminal page width (10 min. - 132 max.).
TP #	TP 60<CR> Set terminal page length.
LP	LP<CR> Select line printer for output.
KLINIK	Enable/disable KLINIK line
PDP-11 Console Commands	
ES	ES<CR> Print present 11 switch register.
ES data	ES 103452<CR> Set 11 switch register to data specified.
E36 adr	E36 5000<CR> Examine specified 11 address for a 36-bit word.
EE adr	EE 3000<CR> Examine specified 11 word address.
EB adr	EB 2001<CR> Examine specified 11 byte address.
D36 adr:data	D36 5000:252525 252525<CR> Deposit 36-bit data specified into 11 address specified.
DE adr:data	DE 3000:103452<CR> Deposit 16-bit data into 11 address specified.
DB adr:data	DB 2001:377<CR> Deposit 8-bit byte into 11 byte address.
ZE adr,adr	ZE 100, 200<CR> Clear the 11 memory from address to address.
SE adr	SE 3000<CR> Start 11 at address specified.
SED	SED<CR> Start 11 diagnostic.
SED #	SED 100<CR> Start 11 diagnostic and run number (#) of passes.
BP adr	BP 3150<CR> Set a breakpoint at 11 address specified.

Table 4 KLDCP Command Summary (Cont)

Command	Description
BC	BC<CR> Continue from breakpoint.
RB	RB<CR> Remove breakpoint.
RG	RG<CR> Print registers saved at breakpoint (R0-R7).
PA, Clock, and, Cache Commands	
PA	PA<CR> Establish a fixed core address for the KL10 communication region. A second PA will make the communications region relative to the EBR.
CS	CS<CR> Print clock source code.
CS #	CS 1<CR> Select specified (#) clock source. 0 normal clock 1 speed margin clock 2 external clock
CR	CR<CR> Print clock rate code.
CR #	CR 1<CR> Select specified (#) clock rate. 0 normal 1 divide by 2 2 divide by 4 3 divide by 8
CE	CE<CR> Print current cache enable code.
CE #	CE 10<CR> Select cache enables according to number code specified (#). 1 (0001) enable cache 3 2 (0010) enable cache 2 4 (0100) enable cache 1 10 (1000) enable cache 0 (default is 17 - all four caches)
CI	CI<CR> Executes cache invalidate instruction.
CF	CF<CR> Executes cache flush instruction.
Microcode CRAM/DRAM Commands	
MM adr	MM 150<CR> Set sync mark (bit 34) at microcode address specified.
MMA adr	MMA 150<CR> Set sync marks from address 0 up to and including address specified.
MU	MU 150<CR> Clear sync mark at microcode address specified.
MUA adr	MUA 100<CR> Clears sync marks from address 0 up to and including address specified.
SM	SM<CR> Start 10 microcode running.

Table 4 KLDCP Command Summary (Cont)

Command	Description
EC adr	EC 112<CR> Examine CRAM at address specified.
DC adr:data	DC 112:123456 123456 123456 123456 123456 12<CR> Deposit data specified at CRAM address specified.
RC adr	RC 123<CR> Examine CRAM address specified using diagnostic functions.
ED add	ED 776<CR> Examine DRAM at address specified.
DD adr:data	DD 776:7 6 1 1234<CR> Deposit data specified at DRAM address specified. NOTE The DD command will prompt for odd address data.
Diagnostic Functions	
FX FUNCT	FX 11<CR> Execute diagnostic function specified (00-37).
FW funct:data	FW 77:252525 252525<CR> Write data specified to diagnostic function address specified (40-77).
FR funct	FR 100<CR> Read and print the contents of the diagnostic function address specified (100-177).
FR functl, functX	FR 100,150<CR> Read and print the contents of each diagnostic function beginning at functl and ending with functX.
FS	FS<CR> Generate a sync pulse at 4A36E1
R/W Major Registers	
DA data	DA 123456 654321<CR> Deposit data specified into AR register.
XX	AR<CR> Read and print the contents of the register specified by XX. XX = ALL - Print all CRAM and registers AD - adders ADX - extended adders ADB - address break AR - arithmetic reg ARX - extended AR BR - buffer reg BRX - extended BR ERG - EBus reg FM - fast memory reg MQ - multiply/quotient reg PC - program counter PI - priority interrupt VMA - virtual memory address VMH - VMA held

Table 4 KLDCP Command Summary (Cont)

Command	Description
KL10 Console Commands	
RI	RI<CR> Reinitialize console program.
MR	MR<CR> Master reset.
HC	HC<CR> Continue from program halt or error.
SW	SW<CR> Print present 10 switch register.
SW data	SW 123456 654321<CR> Set the 10 switch register to the data specified.
EM adr	EM 2000<CR> Examine 10 core at address specified.
EN	EN<CR> Examine next sequential 10 address.
DM adr:data	DM 2000:123456 654321<CR> Deposit data specified into 10 address specified.
DN data	DN 123456 654321<CR> Deposit data specified into next sequential 10 address.
MZ adr,#	MZ 100,50<CR> Clear the number of address specified by # beginning at the 10 address specified.
EX inst	EX 201000 777777<CR> Execute 36-bit instruction specified.
EXP inst	EXP 201000 777777<CR> Executes instruction specified and prints machine state changes at each clock tick.
EXT inst	EXT 201000 777777<CR> Sets up the instruction specified to be executed by the TRACON or TRACE program.
SP	SP<CR> Stop 10, clear run flip-flop.
RN	RN<CR> Start 10, set run flip-flop.
SI	SI<CR> Single instruct, push continue button.
SI #	SI 5<CR> Single instruct the specified number (#) of times.
SIP	SIP<CR> Single instruct and print machine state changes.
PL	PL<CR> Pulse clock one tick.
PL #	PL 21<CR> Pulse clock specified number (#) of ticks.
BU	BU<CR> Burst clock once.
BU #	BU 3<CR> Burst clock the number (#) of times specified.

Table 4 KLDCP Command Summary (Cont)

Command	Description
KL10 CPU Setup Commands	
AC BLK	AC BLK<CR> Print current AC block number.
AC BLK #	AC BLK 7<CR> Select AC block specified (#).
PE	PE<CR> Print KL10 parity enable codes.
PE #	PE 1<CR> Enable KL10 parity options according to code specified (#). 1 (00001) field service probe 2 (00010) DRAM parity 4 (00100) CRAM parity 10 (01000) FM parity 20 (10000) AR/ARX page fail (default is 16)
PD	PD<CR> Disable all KL10 parity options.
File and Device Selection Commands	
FV	FV<CR> Select files-11 media type.
FE	FE<CR> Select secondary front-end load mode.
DL	DL<CR> Switch to DL-DN87S load mode.
AT	AT<CR> Switch to APT10 load mode.
XX #	DT 1<CR> Select specified device type (XX) and unit number (#) for input. XX = DT DECtape DX diskette RP RP04 RX floppy
KL10 Start Commands	
ST adr	ST 4000<CR> Start 10 at address specified.
ST	ST<CR> Start 10 at previously supplied address.
STD	STD<CR> Start 10 diagnostic (EPT adr = 440).
STD #	STD 100<CR> Start 10 diagnostic and run the number of passes specified (#).
EP #	EP 10<CR> Set EOP (end-of-pass) interval count.
STL	STL<CR> Start 10 loader - DIAMON, MAGMON or D20MON (EPT adr = 442).
DDT	DDT<CR> Start DDT (EPT adr = 441).

Table 4 KLDCP Command Summary (Cont)

Command	Description
STM	STM<CR> Start 10 monitor - TOPS-10 or TOPS-20 (EPT adr = 443).
MC	MC<CR> Continue 10 monitor.
RSX	RSX<CR> Boot RSX-20F from KLAD pack.
BT	BT<CR> Boot system to run diagnostics with KLDCP.
B	B<CR> Boot system and run all KL10 diagnostics.
LI	LI<CR> Log in.
LO	LO<CR> Log out.
File Load and Execute Commands	
I file.ext	I X2.CCL<CR> Execute specified indirect file.
J file.ext	J DHDIAG.CMD<CR> Execute specified double indirect file.
JR	JR<CR> Repeat last J command.
JC	JC<CR> Continue interrupted double indirect command file.
P file.ext	P DHKAA.A11<CR> Load specified file.
LE file.ext	LE TRACON.A11<CR> Load PDP-11 .A11 file.
LB file.ext	LB XTECO.BIN<CR> Load PDP-11 .BIN file.
LR file.ext	LR EBOX.RAM<CR> Load microcode .RAM file.
LT file.ext	LT DFDTE.A10<CR> Load KL10 .A10 file.
GO	GO<CR> Go start program just loaded.
File Verify, Write, and Rename Commands	
V file.ext	V DHKAA.A11<CR> Verify - compare specified file against file in core.
CD	CD CHAN.TST 700000,100000<CR> Write the contents of 11 core beginning at the first address specified and ending at the last address specified. The core contents will be written into the file specified on the selected output device. The file specified must already exist on the output device. (See KLDCPU ALLOC command.)
CDA	CDA CRASH.A11<CR> Write entire contents of 11 core to output device using file name specified. The file must already exist on the output device. (See KLDCPU ALLOC command.)

Table 4 KLDCP Command Summary (Cont)

Command	Description
WF file.ext	WF DHKBB.A11<CR> Write - copy file specified from DECtape or floppy to RP04.
RENM	RENM file.ext file1.ext<CR> Rename RP04 file from file.ext to file1.ext.
Miscellaneous Commands	
H	H<CR> Print KLDCP help file.
H file.ext	H TRACON.A11<CR> Print help file for file specified.
T	T<CR> Print time
C msg	C mount the KLAD pack <CR> Send message specified from console terminal to KLINIK terminal and vice versa.

KLDCP ERROR MESSAGE SUMMARY

The following are standard KLDCP error messages listed in alphanumerical order. The notes are referenced in the text.

NOTES

1. This error message could have occurred because of a faulty deposit or examine command. Try a TRACON deposit or examining command. TRACON does not use the PI system.

2. These error messages include the value of the PC at the time of error. The PC allows the field engineer to look up the failing code in the KLDCP listing and determine what combination of instructions caused the fault to occur.

3. These error messages are associated with the APT10 and should never occur in a system installed in the field. The APT10 is an automatic processor tester used by manufacturing to check out the KL10 CPU.

4. These error messages apply to the internal format of the program being loaded. Most likely, these errors will occur as a result of a bad copy of the program or a faulty I/O device.

? ADR - An improper address parameter was used with the command. It may be that the address is nonexistent or inappropriate for the device being addressed. For example, an odd starting address supplied with an SE command would cause an ? ADR error. Check the address parameter of the command.

? APT10 - An APT10 command was issued but no APT10 was selected. See Note 3.

? APT10 ENQ - KLDCP made a service request to the APT10 but the APT10 was unable to perform that service. See Note 3.

? BLK # FLOPPY ERR - A nonexistent block number (#) was used in addressing the floppy disk.

? BP ERR - KLDCP supports the insertion of up to eight breakpoints. Should this number be exceeded, the message ?BP ERR will be printed.

? BUS TIMEOUT - This error occurs as a result of a Unibus timeout condition. That is, a slave sync pulse has not occurred within 15 microseconds after a master sync pulse is issued. The cause of this error depends on the I/O device being serviced at that time. In most cases, however, the cause will turn out to be either the eleven memory or the DTE20. Note that the 15 microseconds timeout delay may vary depending on the characteristics of the I/O devices connected to the Unibus. Some require the delay to be extended.

? CKSUM ERR: ECT - Load line checksum error occurred. See Note 4.

? CLK ERR AT # - The clock logic in the KL10 will not respond to single pulsing. See Note 2.

? COMM ERR # CODE - This is a general APT10 error message. More specific information can be found by looking up the error (#) code. See Note 3.

? CAN'T LOAD - Indicates that the error retry count has been exceeded and the requested file cannot be loaded.

? CAN'T CONT - This message is associated with breakpoints. The breakpoint function uses the stack (R7) to store the return address. If the contents of the stack are changed after a breakpoint has occurred and the operator attempts to continue from the breakpoint by typing BC, the message ? CAN'T CONT will be printed. If it is important to restart the program try an SE command using the address of the breakpoint plus 1.

? CKSUM ERROR - The binary file just read had a data checksum error. The problem could be due to a bad copy of the binary file or a faulty I/O device. See Note 4.

? DF ERR - A diagnostic function parameter error has been detected. For example, a FX120 would cause this message because 120 is not within the acceptable range for a diagnostic function execute. Check the parameters of the diagnostic function.

? DF TIMEOUT AT # - A diagnostic function was executed but there was no response from either the KL10 or DTE20 within a reasonable period of time (a few microseconds). Check power to the DTE20 and the clock in the KL10 - the DTE20 diagnostic should catch this problem. See Note 2.

? DIAMON XFER - DIAMON was unable to transfer a file or part of file to the KL10.

? DM ERR AT # - KLDCP is unable to deposit in the KL10. Try the TRACON deposit command because it does not use the PI system. See Note 2.

? EB PAR - An EBus parity error has occurred. Check the source and direction of the EBus transfer.

? EM ERR AT # - This message occurs as a result of an incomplete examine operation (i.e., the KL10 or DTE20 did not respond properly to the command). Try the TRACON examine command. TRACON does not use the PI system. See Note 2.

? EOF - An unexpected End of File was detected. See Note 4.

? F11 FIND - The specified file cannot be found in the files-11 directory. The directory may have been destroyed. Always write-protect the KLAD Pack.

? F11 LOG BLK - The logical block number given to address files-11 formatted media is nonexistent.

? FATAL - This error message does not pertain to KLDCP. It is a condition reported to KLDCP by a program (usually a diagnostic), running in conjunction with KLDCP. This message occurs when such a program encounters an error condition it was not designed to handle. For example, if, while running an MBox Diagnostic, the EBox fails and the MBox diagnostic cannot recover on its own, it will request KLDCP to print the message "? FATAL." There are several ways to approach this problem.

1. Check for outstanding MCOs.
2. Try a different copy of the program.
3. Load it from a different I/O device.
4. Review the diagnostic hierarchies to determine if preliminary programs should be run.

? FATAL INTR - Fatal Vector Interrupt. KLDCP uses FLAG MODE to keep track of I/O devices. KLDCP does not use the PDP-11 priority interrupt system. Therefore, any vector interrupt is unexpected and considered fatal. Should this error occur, run the PDP-11 priority interrupt diagnostics.

? FORMAT ERR: ECT - The format of the load line is incorrect. See Note 4.

? HARD DTA ERROR - A hard (nonrecoverable) error has occurred in the DECtape subsystem. This is generally a controller- or transport-type problem.

? HARD FLOPPY ERROR - A hard (nonrecoverable) error has occurred in the floppy subsystem. A problem of this type usually indicates a controller or device error.

J CMN - The "common area" of the DRAM data does not match. Check the common area and retype the command.

? J SIZE - The size of the DRAM J field is too large. Check the size and retype the command.

? KL10 CLOCK ERROR STOP - This message occurs as a result of an error stop condition - FM parity, CRAM parity, DRAM parity, or FS probe. The reason for the error stop is reported along with the error message.

? KL10 HALTED - The KL10 executed a halt instruction. Check the KL10 PC and refer to the program listing.

? KL10 RUNNING ECT - Certain console commands cannot be executed while the KL10 is running. They are as follows.

Diagnostic functions
Internal EBox resistor reads
Pulsing the clock
Clock rate source changes
Microcode mark and unmark functions
CRAM and DRAM deposits and examines
Cache invalidate and flush
Clearing KL memory
AC block selection

Before executing any of these commands, stop the KL10 by typing the SP command.

KLDCP CKSUM - The KLDCP code has been changed since the last checksum operation was performed. If the code was not deliberately modified to patch around a problem or execute a slightly different operation, this could mean any of the following.

1. The last command executed somehow, inadvertently, changed the code.
2. The console front-end system has developed a problem.
3. The DTE may have a fault that caused data to be written into the wrong area of 11 core.

? LINE TOO LONG - The internal file data line is too long (in excess of 132 characters). See Note 4.

? LOAD CHR ERR: ECT - The load line identification character is invalid. See Note 4.

? LP ERR - KLDCP has detected an error status coming from the line printer.

LPT OFF - The line printer appears to be off-line.

? MZ ERR AT # - This message occurs as a result of an incomplete MZ deposit operation. Try the TRACON deposit command. It does not use the PI system. See Note 2.

? NAME EXT - An invalid file name or file extension was used.

? NO LPT - There is no detectable line printer.

? NO MASTER DTE - KLDCP will not run with the DTE20 in restricted mode. This is because a restricted DTE20 will not allow the execution of the diagnostic functions. If this error message is printed, check the switch on the DTE20. Other possibilities are that the DTE will not respond to the Unibus address, or the DTE has lost power.

NON-EX FILE - KLDCP could not find the file as specified. Try a directory command DI.

? PARAM - As soon as a command is entered, KLDCP checks to assure that the typed-in parameters of the command fall within acceptable boundaries. If the parameters are outside the boundaries for that command, the error message "? PARAM" is printed. For example, a nine (9) entered in an octal field would cause a ? PARAM error message to be printed. Check the parameters of the line and retype the command.

? RES INST - There are certain PDP-11 operation codes that are not implemented by the hardware. These are referred to as reserved instructions and should never be executed. Execution of a reserved instruction will cause a trap to address 10 and KLDCP will print "? RES INST." This type of error usually indicates that some portion of the core was destroyed. Try reloading. If that does not correct the problem, run the PDP-11 processor and memory diagnostics, including the diagnostic that checks the reserved instructions.

? RESPONSE - The APT10 has failed to respond within a reasonable amount of time. See Note 3.

? REV DTA ERROR - KLDCP allows for three reversals in tape motion during a search. If that number is exceeded, the error message ? REV DTA ERROR is printed.

? RP04 ERROR # CODE - This error message occurs as a result of an RP04 error. The number code corresponds to one of the following:

1. Unit number incorrect
2. Drive not available
3. Drive unit error 1
4. Drive unit error 2
5. Drive unit error 3
6. Home block read error
7. Not home block
10. Incorrect file system name
11. No index file
13. Reading past EOF
14. Blk size position error
15. Read error
16. Attempt to change allocation
17. Buffer size
20. Current position
21. Insufficient allocation for write
22. Directory rewrite error
23. Data block write failure
24. End of file
25. Rad50 conversion error

? SEL ERR - KLDCP cannot select the requested AC block. KLDCP uses the AR data path and diagnostic functions to select the AC block. A select error usually indicates a faulty data path or diagnostic function.

? SOFT DTA ERROR - This message indicates that a soft (recoverable) data error occurred on the DECtape. This is usually a media problem.

? ST UNFLO - Stack underflow. This error occurs any time the software attempts to POP more entries off the stack than were originally pushed onto it. This error indicates that the KLDCP code was destroyed. Reload KLDCP. If that doesn't correct the problem, run the PDP-11 Processor and Memory Diagnostics.

? UCODE HUNG - The microcode is not in the halt loop. This may indicate that the KL10 is not set up properly to execute the command (i.e., the ucode is not loaded) or that the ucode did not return to the halt loop. It may be hung up waiting for a memory response.

? - KLDCP does not recognize the command as typed. Check for proper format and retype the command.

? 10 CLK OP - The KL10 uses the clock in the PDP-11 to keep track of time. This error message indicates that the PDP-11 cannot notify the KL10 that a clock tick has occurred.

? 10 CMD ERR - The program running in the KL10 has issued an illegal command to KLDCP.

? 10 SW - KLDCP is unable to notify the KL10 of a change in the data switches. See Note 1.

? 10 TTI - KLDCP is unable to send a teletype character to the KL10. See Note 1.

? 11 PARITY - An 11 parity error has been detected. Run 11 memory NPR device and DTE20 diagnostics.

KLDCPU

GENERAL INFORMATION

Code	DGQDB.A11
Title	DECSYSTEM Diagnostic Console Utility Program
Abstract	KLDCPU is a console utility program which resides in the lower half of 11/40 core and extends the KLDCP command set to include file maintenance service. The utility is capable of performing operations on DECtape, floppy and KLAD packs. The utility has single file manipulations capability and also facilities for handling groups of files.

Notes

1. KLDCP will perform a validity check of the utility portion and will request that the operator load KLDCPU. All if it is not resident when any utility command is performed.

2. Any command which is not one of the utility commands is automatically passed to KLDCP for processing. This allows all the KLDCP commands to be performed from the utility command process.

3. The SAVRSX and KLADBT commands are used to change the hardware boot on the disk so that KLDCP is booted when the disk button is pushed.

   RSX-20F must have been installed on the KLAD-10 disk so that the proper exchange takes place.

   RSX-20F is then booted when required by the KLDCP RSX command. This command reads the RSX-20F boot block from the RSXBT.ZRO file; installs it in memory starting at zero; and starts it as though a switch register disk boot were done.

4. Wild characters - the asterisk (*) and question mark (?) - may be used in file name construction.

Loading and Starting Procedure	Standard (Refer to the 11/10 STD module.)
Control Switches	None

OPERATIONAL CONTROL

KLDCPU commmands may be entered either directly via the CTY or KLINIK link or indirectly via a control file.

The conventions used to illustrate KLDCPU commands are described in Table 5. KLDCPU switches which may be used to modify the commands are described in Table 6. The commands supported by KLDCPU are summarized in Table 7.

Table 5 KLDCPU Command Conventions

Convention	Description
↑C	Control C returns to command mode, aborting the operation in progress.
↑Z	Control Z exits TEXT mode.
:	Delimits device specification.
=	Delimits input and output file specifications.
DTn:	DECtape unit n.
RXn:	Floppy unit n.
DXn:	Floppy unit n.
RPn:	RP04/06 unit n. The RP04/RP06 disk is a read-only device, as the file structure is maintained via the TOPS-10 or TOPS-20 systems. The disk may be either the KLAD-10 or the KLAD-20 format; selection of the disk will automatically select the proper processing operations.

Table 6 KLDCPU Software Switch Summary

Switch	Description
/F	DIR DT0:/F<CR> Print the directory in abbreviated format.
/N	FILE DT0:DT1:FILE.EXT/N<CR> Do not list each file name as it is transferred.
/H	/H<CR> Print the help message.

Table 7 KLDCPU Command Summary

Command	Description	Cross Ref.
REMOTE	REMOTE<CR> Select remote terminal.	1
RI	RI<CR> Reinitialize console (KLDCP).	2
BOOT	BOOT RX0:<CR> Load and start the bootstrap loader from the device specified.	3
SVBOOT	SVBOOT DT0:=RP0:KLDTBT.BIN<CR> Create the specified boot file and write it to the boot block of the specified output device.	4
KLADBT	KLADBT<CR> Write KLADBT.ZRO to the boot block of the KLAD-10 pack.	5
SAVRSX	SAVRSX<CR> Transfer the boot block of a KLAD-10 disk to the file RSXBT.ZRO.	6
DIR	DIR DT0:<CR> Print the directory for the specified device.	7

Table 7 KLDCPU Command Summary (Cont)

Command	Description	Cross Ref.
FID	FID RP0:DGMMA.*<CR> Print the specified files, file identification line.	8
RENAME	RENAME RX0:DGKAA.A11=RX0:DGKAA.OLD<CR> Rename the specified file to a new name.	9
DEL	DEL RP0:DHKAA.A11<CR> Delete the specified file from the device specified.	10
ZERO	ZERO DT1:<CR> Clear the directory of the device specified.	11
ASG	ASG RP0:=MASTER:<CR> Assign the specified logical name to the physical device specified. Acceptable logical names are: IN, OUT, MASTER and NEW.	12
DATE	DATE: 31-OCT-77<CR> Change the date used by KLDCPU format = DD-MMM-YY.	13
ALLOC	ALLOC DT0:CRASH.DMP/100<CR> Allocate an empty file (having 100 blocks) for future use on DECtape or floppy disk.	14
PIP	PIP RX0:file.IN=DT0:file.OUT<CR> Transfer the file specified from the input device to the output device.	15
FILE	FILE DT0:=DT1:*.A11<CR> Perform bulk file transfers between the input and output devices.	16
FILET	FILET DT0:*.*<CR> Test files specified for error.	17
DTCOPY	DTCOPY<CR> Copy all the files from one DECtape to a second DECtape.	18
RXCOPY	RXCOPY<CR> Copy all the files from one floppy disk to a second floppy disk.	19
TAPT	TAPT DHKCA.A11<CR> Transfer the file specified from the APT10 to the KLAD-10 disk pack.	20
TEXT	TEXT RP0:CPU.CCL<CR> Build an ASCII command file and write it to the device specified using the filename and extension specified.	21
DO	DO CPU.CCL<CR> Execute the specified command file.	22

COMMAND DESCRIPTIONS
This section describes the commands summarized in Table 7.

1. REMOTE<CR> - The REMOTE command selects remote terminal operation over the APT10 communication link.

2. RI<CR> - The RI command is passed to KLDCP where it reinitializes the diagnostic console and returns control to KLDCP.

3. BOOT RX0:<CR> - The BOOT command causes block 0 of the device to be read into memory starting at location 0. Block 0 is assumed to contain a bootstrap loader. The utility then transfers control to the boot just read in at location 0.

4. SVBOOT DT0:=RP0:KLDTBT.BIN<CR> - The SVBOOT command reads the specified binary file (KLDTBT.BIN) and writes it out to the specified output device (DT0) in the boot block and to the core image boot blocks. (The file specified must have a bootstrap format.)

5. KLADBT<CR> - The KLADBT command copies the KLDCP bootstrap loader file (KLADBT.BR0) to block 0, cylinder 0 of the KLAD-10 pack.

6. SAVRSX<CR> - The SAVRSX command copies block 0 cylinder 0 of the KLAD-10 to the file RSXBT.ZRO.

7. DIR DT0:<CR> - The DIR command gives a directory of the requested device (DT0). This command will give the entire directory or a partial directory of requested files via use of the wild character or asterisk constructions.

DIR RP0:<CR>	Prints a full directory.
DIR RP0:*.BIN<CR>	Prints a directory of all files with a BIN extension.
DIR RP0:A7????.*<CR>	Prints a directory of all files. Those first 2 characters are A7.

8. FID RP0:DGMMA.*<CR> - The FID command prints a file identification line directory of the requested device (RP0). The file identification line is the first line of an ASCIIzed file which provides internal file identification (i.e., file name, file version and creation date). The FID command provides the same wild character and asterisk constructions as does the DIR command.

9. RENAME RX0:file.new=RX0:file.old<CR> - The RENAME command renames an old file (file.old) to a new file name (file.new).

10. DEL DT0:DHKAA.A11<CR> - The DEL command causes the file specified (DHKAA.A11) to be deleted from the directory of the device specified (DT0).

11. ZERO DT1:<CR> - The ZERO command clears the directory of the device specified (DT1).

12. ASG RP0:=MASTER:<CR> - The ASG command allows the use of logical names in command files. Allowed logical names are: IN, OUT, MASTER, and NEW. A command file may use a logical name such as "MASTER" instead of specifying a physical device. Then, before executing the command file the user can assign the desired physical device to the logical name. This permits the use of any available unit.

13. DATE: 31-OCT-77<CR> - The DATE command allows changing the date used by the utility operations. Type the date according to the following format.

    DATE: DD-MMM-YY

    DD is the day of the month.
    MMM is the month: JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC.
    YY is the year.

    When transferring files to a new medium the original file's date is used unless the input device is the disk, in which case the typed-in date is used.

14. ALLOC DT0:CRASH.DMP/100<CR> - The ALLOC command allows an empty file (CRASH.DMP) to be allocated on either the DECtape or floppy for subsequent use by the CORE DUMP KLDCP file generation command.

    The size is the number of blocks required. If the "/SIZE" is not given in the command, the size will be specifically asked for.

15. PIP RX0:file.IN=DT0:file.out<CR> - The PIP command is used to transfer a file (file.out) from one device (DT0) to another device (RX0). The device types may be different and the file name may be changed; however, asterisk and wild character constructions may not be used. The output file name must not exist on the output device.

16. FILE DT0:=DT1:*.All<CR> - The FILE command is used to do bulk transfers (i.e., all files on DT1 with an .All extension) from one device to another device. The FILE command is similar to the PIP command except that it can utilize the asterisk and wild character constructions. If a file of the same name already exists on the output device, the file command will delete the old file.

17. FILET DT0:*.*<CR> - The FILET command tests all files named by reading them into a buffer to make certain that no device errors occur. Any device errors are listed as they occur.

18. DTCOPY - To be supplied.

19. RXCOPY - To be supplied.

20. TAPT - To be supplied.

21. TEXT RP0:CPU.CCL<CR> - The utility includes the facility to execute a sequence of commands contained in ASCII text file. This text file may be created via the TEXT command.

    When the TEXT command is issued the named output file is opened for output and the operator is prompted with a quotation mark (") to indicate readiness to accept text. Any normal ASCII command character may be placed into the file.

    RUBOUT can be used to delete characters on the current line (but not on preceding lines).

    CONTROL C (↑C) will abort the text operation.

    CONTROL Z (↑Z) is the standard terminator for input. It will close out the text file and return to command mode.

22. DO CPU.CCL<CR> - The DO command is used to cause execution of a control file. The file is executed line by line and may contain either utility commands or KLDCP commands. Executable files are created via the TEXT command or via any of the text editors.

KLDCPU ERROR MESSAGE SUMMARY
The following is an alphabetical listing of KLDCPU error messages.

DELERR - A bit map error occurred during a delete operation.

DELOLD - Delete the old file before issuing a command which would create a file with the same name.

DEVERR - A device error occurred on either the input or output device. Check that the output device is write-enabled.

DEVFUL - The output device is full. There is no more file storage room available.

DIRERR - An invalid file name exists in the device directory.

INVCMD - The command issued is invalid. Examine the command for proper format and retype.

INVDEV - The device specified in a command is invalid. Check the command for proper device mnemonic and retype. If the error occurred as a result of a command file, check for logical device assignments.

INVNAM - Invalid name. No special characters are allowed (A through Z and 0 through 9 only). This error will also occur if asterisks or wild character constructions are used with a command which does not support them (i.e., PIP). Check the command file name field.

INVSW - An invalid switch was used in the command string. Refer to Table 6.

NEXFIL - The file specified in the command string does not exist. Check the directory of the device.

GENERAL INFORMATION

Code DGQFB.A11

Title KL10 Diagnostic Memory Boot Utility

Abstract This program provides all of the functions necessary to configure the KL10 memory system when running in the front-end resident, KL10 diagnostic environment. This program runs in the PDP-11 under KLDCP. KL10 memory types handled include external core memory (DMA20), internal core memory (MA20/MB20), and MOS memory (MF20). All reasonable mixtures of these devices can be handled together.

The memory boot procedure goes in three basic steps.

1. **Determining physical resources - RESDET -** Determining physical resources, or "RESDET" for short, is the longest and most involved part of the memory boot procedure. Different procedures occur for different memory types, but basically the program determines what physical memory it has to work with. Listed below are the things the program must do for each memory class.

   Internal Core Memory - MA20 and MB20 Controllers
   Find out which controllers, if any, exist. Find out which storage modules exist on each controller. Determine the set of legal starting addresses and the interleave mode for a controller or controller pair.

   External Core Memory - DMA20 Controller
   Find out if the DMA20 exists. Determine its address response(s) and the size of the response(s). Determine the legal interleave modes available. The address response(s) of external memory are fixed and the program must work around whatever it is.

   MOS Memory - MF20 Controllers
   This is very different from the core memories. In addition to finding out what exists the program must also find out the state of the controllers. Because MOS RAMs fail on a regular basis there is a lot of hardware in the controllers to compensate for these failures. The software closely controls the hardware and it is therefore important that the program knows what has already been done.

   If the controller is already configured (it is at software state 2 or 3), then the program treats it as if the address response could not be changed. In this sense it is treated like external core. However, if the program finds some bad hardware, that hardware is eliminated.

   If the controller is not configured but is otherwise initialized (it is at software state 1), the program merely records what storage it has to work with.

   If the controller has not been initialized at all (is at software state 0), then the program has a considerable amount of initialization to do. The double bit error (DBE) scan is by far the most time consuming part of the memory boot process, taking about 22 seconds per 256K of MOS RAM. Fortunately, once this is done the controller is at software state 1 and the DBE scan does not have to be done again until the next power fail. MOS storage blocks found to be irreparably bad are eliminated.

2. **Determination of Logical Configuration - FITMEM** - In this phase the program determines which configurable resources (MA20, MB20, and software state 1 MF20) will go where in the holes in the address space. Hole locations and sizes are determined by the response of the external core memory, preconfigured (software state 2) MF20 memory, and the absolute bounds of the memory space. This process does not involve the hardware at all; it is purely computational.

   The philosophy behind this algorithm is to maximize storage even at the cost of some interleave factor. No memory is ever thrown away except for certain impossible-to-configure conditions which might arise with MA20s or MB20s.

3. **Configuration of the Memory - CONFIG** - Here the program takes the logical configuration tables and sets up the hardware to match. After this phase is completed, the KL10 memory system is ready for use.

Notes

Memory controllers are assumed to have passed their respective diagnostics (DGKBB/DHKBB, DHKBF, and/or DHKBG).

It is assumed that the KL10 processor is working and that some valid microcode is already loaded. There must be master oscillator if MOS memory exists.

When MEMCON is started it will do a start microcode in order to make sure that microcode is loaded and running. Because of this, any special state which may have existed in the CPU will be lost.

Loading and Starting Procedure

Standard (Refer to the 11/10 STD module.)

Control Switches

None

OPERATIONAL CONTROL
Once started, MEMCON will prompt with a > (TAB). The user may then enter commands. There are two classes of commands: those involved with configuring the memory system (Table 8) and those which perform functions ancillary to using the memory system (Table 9). KLDCP commands may be entered directly if no naming conflict occurs. Preceding a command with a period ensures that KLDCP will process it. Example: ".RP0" selects RP04/RP06 drive 0 for KLDCP, whereas "RP0" says to report the physical resources.

ERROR MESSAGE SUMMARY
There are no error messages unique to MEMCON.

Table 8 MEMCON Memory Configuration Command Summary

Command	Description
CM	CM<CR> or CMF<CR> etc. Determine, report, and set the configuration. Then clear the memory boot. Memory is now configured and ready for use. All physical resource data has been cleared out. This command will automatically do the physical resource determination if it has not already been done, and is therefore the only essential command for configuring memory. See switches.
DL	DL<CR> Determine logical configuration. Report it; but do not set it. This command is useful for seeing what the configuration would be if it were set. See switches.

Table 8 MEMCON Memory Configuration Command Summary

Command	Description
DP	DP<CR> Determine the physical resources and report them. This forces the memory boot to start from scratch. Time already spent on MF20/MG20 DBE scan is not lost providing the previous scan ran to completion. See switches.
Switches	The switches are the same for DP, DL, and CM commands. Typing no switch will use the switches typed for the previous DP, DL, or CM command. If there was no previous DP, DL, or CM command, then the defaults are as shown below. The switches may be in any order.
0,1,2, or 4	Force MA20/MB20 interleave unless memory loss would result. Force DMA20 bus mode if legal. 0 (default) gives optimal results.
F	Force MF20/MG20 address reconfigure. In this mode preconfigured MF20/MG20s are always deconfigured before the memory resource fit is done. This is not the default. While the "F" parameter to the CM command is not the default, most of the time while in the diagnostic environment the user will want to use it. The recommended minimum command is therefore "CMF".
K	Keep bad MF20/MG20 blocks. Normally MF20/MG20 blocks which are irreparably bad to the memory boot can still be used partially by monitor if it marks certain pages as unusable. After a brief power fail, monitor should still have this bad page data intact; therefore it is safe to tell the memory boot to keep bad MF20/MG20 blocks. This is not the default. Ignored if "F" switch given.
R	Reverse configuration where possible. This is useful for shuffling memory around for diagnostic reasons. It is not normally used otherwise. The "F" switch should be used if this switch is given.
Sn	Substitute MF20/MG20 spare bits for bit n (decimal) in all MF20/MG20s. This is useful for fixing MOS array boards. The number n must be followed by a space or <CR>. No parameter says to force no swaps. The value of n is 0-35 for data bits, and is 36-42 for ECC 32, 16, 8, 4, 2, 1, and parity.

Table 9 MEMCON Ancillary Command Summary

Command	Description
↑C	Exit back to KLDCP.
↑Z	Exit back to KLDCP.
C0	C0<CR> Clear all function 0 error flags. Use before first DP, DL, or CM command and after diagnostics which intentionally cause memory errors.
DA	DA<CR> Dump PC, VMA, previous and current AC block numbers, and the contents of AC blocks 0-6. Very useful data to accompany a diagnostic bug report. The code to do this command resides in the overlay DBGOVL.A11.

Table 9 MEMCON Ancillary Command Summary

Command	Description
DR	DRx n<CR> Dump the content of the MF20/MG20 logic control RAM "x" to the console terminal. The meaning of x is "A" for address response RAM, "B" for bit substitution RAM, "E" for fixed value logic RAM, or "T" for the timing RAM. Note that if refresh is running it may interfere slightly with a timing RAM dump. The value of n is the MF20/MG20 controller number in the range 10-17.
IC	IC n<CR> Force an initialization of the specified MF20/MG20 n. This performs the minimum initialization required to talk to the MF20/MG20. The address response RAM is set up so that address bits 18-21 determine which block is being used.
KP	KP c1, c2 s1, s2<CR> Kill physical resources s1 through s2 in memory controllers c1 through c2. This command is used after the DP command. Its purpose is to get rid of storage resources that are not to be used; (i.e., they do not work). s1 and s2 are storage module numbers for MA20/MG20s and MB20s, and octal block numbers (0-13) for MF20/MG20s.
MO	MO n<CR> Select master oscillator frequency source, where n: =3 for normal (30 MHz); n = 2 for slow (25 MHz, which is for extending a board); n = 1 for fast (31 MHz, for margining the system); n = 0 is external oscillator. Do not use 0 unless a running VFO has been physically attached to the external oscillator input. Meaningless if there is no master oscillator.
PD	PD<CR> Enter a program patching dialogue where the address and content of that address are typed, and then the value the user types in goes to that address. Typing <CR> causes the data to remain the same. Typing <ESC> causes the patcher to ask for a new address. Typing <ESC> to the address enquiry causes exit. The first address used by the patcher is the first free location. The first free pointer is automatically updated as required. The code to do this command resides in the overlay DBGOVL.A11.
RI	RI<CR> Reinitialize the memory boot. The various switches and control flags are put back the way they were when the program was first loaded.
RP	RP<CR> Report physical resources. This command does not do anything other than report the content of the physical resource tables. It is useful after using the KP command to find out if an error has been made.
SD	SD w<CR> Take the 36-bit word "w" and use it as the "to MEM" word of an SBUS DIAG cycle and type the word sent back "from MEM". If the SBDIAG instruction fails then nothing is typed.
SR	SR<CR> Do an SBUS reset.
TC	TC<CR> Test configuration. This must only be done after the CM command. The test consists of reading words 20-23 on every 16K boundary. The response of all NXMs or no NXMs is then compared to what the program thinks it should have at a given address.

GENERAL INFORMATION

Code	DGQFA.A11(KL10-PA) and DHQFA.A11(KL10-PV)
Title	TRACON-KL10 Diagnostic Console Signal Tracer
Abstract	TRACON resides in the lower half of the 11/40 core. It extends the console command set of KLDCP and aids in troubleshooting KL10 central processor, channel and memory faults. TRACON commands primarily control the CPU clock, and detect and display changes in registers and control signals.
Notes	1. TRACON commands prompt for missing arguments. Responding to a prompt with an altmode ($) will abort the command. 2. KLDCP commands may be executed from TRACON by preceding the command with a period (.). 3. System standard or diagnostic microcode must be loaded in the KL10.
Loading and Starting Procedure	Standard (Refer to the 11/10 STD module.)
Control Switches	None

OPERATIONAL CONTROL
TRACON commands may be entered directly from the CTY or KLINIK link or indirectly from a control file.

TRACON commands are divided into two groups: control functions which are described in Tables 10 and 11, and extension commands which are described in Table 12.

Control functions affect TRACON's mode of operation and should not be used in control files. Extension commands are intended for general use and may be included in control files.

ERROR MESSAGE SUMMARY
There are no error messages unique to TRACON.

Table 10 TRACON Control Function Summary

Command	Description	Cross Ref.
A	A<CR> Auto insert - automatically builds an internal command file as commands are typed.	1
E	E<CR> Edit or create a command buffer. Refer to Table 2.	2
ML	ML<CR> Mark loop "starting point"	3
FB	FB 162,31,1<CR> Set function breakpoint at the diagnostic function, bit, and polarity specified.	4
FC	FC<CR> Function break continue	4
CB	CB<CR> Clear function breakpoint	4
RG	RG<CR> Print function breakpoint registers (R0 through R7)	4
KA	KA<CR> Kill (terminate) auto insert; also resets loop marker to line 1	5
T	T<CR> Type contents of command buffer	6
X	X<CR> Execute command buffer	7
L	L<CR> Loop on command file	8
M	M<CR> Multi-burst, step, and trace the command buffer	9
DC	DC CHAN.TST<CR> Write command buffer to an existing file	10
LC	LC CHAN.TST<CR> Load specified control file	11
K	K<CR> Kill command buffer (confirm with a K)	12
H	H<CR> Print command summary	13
/	/<CR> Enter switch dialogue	14

Table 11 TRACON Edit Command Summary

Command	Description	Cross Ref.
E	E<CR> Enter lines into buffer.	15
D #	D 5<CR> Delete specified line (#) from command buffer.	16
I # text	I 7 SET CHAN 3<CR> Insert text before specified line number (#).	17
R # text	R 14 SC 2, START<CR> Replace text at specified line number.	18
K	K<CR> Kill the command buffer (confirm with a K).	19
T	T<CR> Type the contents of the buffer.	20
↑C	↑C CTRL C - Exit from edit mode; return to TRACON command mode.	21

Table 12 TRACON Extension Command Summary

Command	Description	Cross Ref.
SET mode	SET EBR 3,CHAN 1<CR> Set: CACHE EN, PMA, EBR # and/or CHAN #.	22
CLR mode	CLR CACHE,ERB<CR> Clear: CACHE EN, PMA, and/or CHAN #.	23
RM	RM<CR> Reset MBox (force halt loop and set cache look and load if cache is enabled).	24
CE chan,ccw	CE 2,100<CR> Configure EPT for channel specified.	25
SC chan,cmd	SC 1,STA,RES<CR> Simulate CBus command for channel and command or for EBus data specified. Commands are: START, RESET, CTOM, DONE, STORE and SLOW.	26
QC chan,cmd	QC 1,STA,RES<CR> Queue CBus command for memory trace.	27
T1	T1<CR> Trace and print one memory reference (used with the QC command).	28
TM	TM<CR> Trace and print all memory references (used with the QC command).	29
CH	CH<CR> Print default channel number.	30
NC	NC<CR> Next channel (increment the default channel number by 1).	31
CU	CU<CR> Cache refill load (standard).	32
C #	C 3<CR> Cache refill load (use only Cache specified: 0, 1, 2, or 3).	33
IC	IC<CR> Invalidate Cache (use after refill load).	34
VC	VC<CR> Validate core from cache.	35
I	I<CR> Initialize the tick counter to 0.	36
B #	B 29<CR> Burst specified number (#) of clock ticks and report change.	37
C	C<CR> Continue advancing clock.	38
F #	F 14<CR> Find the clock tick # specified.	39
G	G<CR> Go - reset tick counter, stop the clock and print machine state changes.	40
S	S<CR> Single-pulse the clock and report machine state changes.	41

Table 10 TRACON Control Function Summary (Cont)

Command	Description	Cross Ref.
P	P<CR> Print EBus activity summary since last P or D command.	42
R	R<CR> Read and print machine state changes since they were last reported.	43
D	D<CR> Print the current state of the machine.	44
W filename	W CRASH<CR> Write a crash dump - must specify an existing file.	45
D filename	D CRASH<CR> Print the machine state saved by the W command.	46
EM addr	EM 2000<CR> Examine KL10 address (does not use PI system and the KL10 must be halted).	47
EN or EM	EN<CR> EM:<CR> Examine next sequential KL10 address.	47
D addr:data	D 2000:254000,020000<CR> Deposit data into KL10 address (does not use PI system and the KL10 must be halted).	47
DN:data	DN:254000,001472<CR> Deposit data into next sequential KL10 address.	47
EX inst.	EX 201000,777777<CR> Execute KL10 instruction.	48

TRACON COMMAND DESCRIPTION

This section describes in detail each of the commands summarized in Table 10, Table 11, and Table 12.

1. A<CR> - The A command opens the command buffer for input. All commands typed following an A command are entered into the buffer until a KA command is typed. The commands in the buffer are executed via the X, L or M command. The buffer may be saved for future reference with the DC command.

   NOTES

   1. KLDCP commands may be used in the command buffer.

   2. Commands are automatically parsed before they are entered in the command buffer. For this reason it may be necessary to reconstruct a command for inspection.

2. E<CR> - The E command enters edit mode. The editor may be used to create or edit the command buffer. Edit commands are summarized in Table 2.

3. ML<CR> - The Mark Loop command requests a line number for use with the L command.

4. **Function Breakpoint Command** - A function breakpoint is a mechanism which permits detection of an event (signal) in the KL10. The clock will be stopped when the leading edge of the event is detected. The event is specified by entering a diagnostic function code, a bit number and a 1 or 0 to select the polarity desired. Once set, the KL10 clock will be stopped and the user notified each time the signal specified transitions to the state selected. Only one function breakpoint may be set at a time. Since this mechanism depends on single-pulsing the clock through the function being performed, only extension commands are affected.

   FB 166,30,1<CR> - Set a breakpoint for diagnostic function 166 bit 30 on a 1 (MEMRQ 1 H). The clock will be stopped on the leading edge of MEMRQ 1 H. Other commands can now be used to read the state of the machine.

   FC<CR> - Continue the command execution until either the next detection of the break condition, the end of the current extension command, or the end of the command buffer.

   CB<CR> - Clear the function break condition set by the FB command.
   RG<CR> - Print the contents of the function break registers R0 through R7.

5. **KA<CR>** - The KA command performs two functions: it terminates auto insert (A), and it resets the loop marker (LM) to line 1.

6. **T<CR>** - The T command prints the contents of the command buffer.

   NOTE
   The commands in the buffer are automatically parsed. Therefore the commands may be printed in a slightly different format.

7. **X<CR>** - The X command executes the contents of the command buffer once.

8. **L<CR>** - The L command repeatedly executes (loops on) the commands in the buffer. After the first execution of the buffer, execution begins at the line specified by the loop marker (ML). If no ML command has been executed the loop marker defaults to line 1.

9. **M<CR>** - The M command:

   a. clears the tick counter
   b. burst-executes the command buffer
   c. prints the state of the machine, and
   d. increments the tick counter by 1.

   Steps b through d are repeated until the user interrupts by typing an altmode ($), or until the EBox enters the halt loop, or until no change in machine state is detected for a prespecified number of ticks (refer to TRACON Switches Number 14). The M command, in effect, allows the command buffer to be executed at full speed while printing the machine state at each tick.

   NOTES
   1. The first commands in the buffer must initialize the CPU to an exact known state. Otherwise, the reported changes will be garbaged beyond usefulness.

   2. The C command may be used to continue the trace if it was stopped with an altmode ($).

10. DC CHAN.TST<CR> - The DC command writes the contents of the command buffer to the specified file (CHAN.TST). The file must already exist on the output device.

NOTE
A temporary file can be generated using the KLDCPU ALLOC command.

11. LC CHAN.TST<CR> - The LC command loads the specified control file (CHAN.TST) into the command area of core.

12. K<CR> - The K command clears the command buffer. TRACON requires the K command be confirmed by typing a second K.

13. H<CR> - The H command prints a summary of TRACON commands.

14. /<CR> - The / command allows the user to specify groups of registers and signals to be traced. Each group is divided into subgroups which may be turned on or off. The groups are as follows.

    Signals

    EBOX - PI, MCL, CLK, DIA, CTL, CON, MTR, SCD, VMA, CRA

    MBOX - CSH, CHX, MBC, MBX, MBZ

    CHAN - CCL, CH, CCW, CRC

    CYCLIC - Any signals which change frequently. The current list is as follows.

        EBUS CLK
        SBUS CLK
        EBOX SOURCE
        SYNC
        EBOX CLK
        A CHANGE COMING A
        B CHANGE COMING
        PHASE CHANGE COMING

    Registers

MICRO	DRAM ABJP, CRA LOC, CR ADR, SBR RET, CRAM NN, DISP, IR, AC, TRAP MIX
DATA/ADDR	PIH, PIO, PI GEN, VMAH OR PC, CLK BURST, FM BLOCK & ADR, AR, ARX, BR, BRX, AD, ADX, FM, MQ, SC, FE, VMA, VMAH, ADR BRK, PC, EBUS REG
METER	CACHE COUNT, EBOX COUNT, INTERVAL, PERF COUNT, PERIOD, TIME
CHAN ADDR	CCW CHA, CH BUF ADR

    The NO CHANGE LIMIT may also be altered with the / command. The limit is used to stop a trace after a specified number of clock ticks with no observed changes in machine state. The current limit is output and the user may enter a new number or a carriage return if the limit is satisfactory.

15. !E<CR> - Enters lines into buffer. The user types E<CR> and the editor outputs a line number at the left margin. After an initial load or an editor K command, the first number output will be 1. If the buffer contains information, the next free line's number is output. After each number, the user enters any extension or console command. Prompting is enabled. No validity checking occurs for console commands. To terminate entry, type an altmode following the line number output.

Example:

```
* E<CR>
!E<CR>
1       RM<CR>                  ;Reset MBox
2       SET EBR 3<CR>           ;Set executive base
                                ;register to 3
3       CE 0,200000 100<CR>     ;Condition channel 0 EPT
4       .DM 100:0<CR>           ;Put a CHLT in command list
5       QC 0.START/RESET<CR>    ;Start channel 0
6       TM<CR>                  ;Watch it fetch a halt
7       $<ALTMODE>              ;Exit-enter command mode
```

16. !D # - Deletes the line # and renumbers all the lines which follow it. (Line numbers are not "sticky;" if needed, use the T command to type all line numbers and their current contents.)

17. !I # <TEXT> - Insert text before line number (#). All lines starting from # are moved down and the text inserted in the resulting hole. As in the D command, lines are renumbered.

18. !R # <TEXT> - Replace text at line # with new text.

19. !K - Kill the buffer. Resets the line count to 0 and recovers the buffer storage space. Confirm with K.

20. !T - Type out the buffer. Types line numbers and text.

21. !↑C - Exit from editor mode to TRACON command mode.

22. SET - The set command alters the operating mode of TRACON, and modifies the performance of the RM command so that the function(s) set is repeated each time the RM command is executed.
    SET CHAN #<CR> - Sets the default channel number to (#) for the CE, SC, and QC commands. Once a channel number has been set, prompting for channel numbers will not occur.

    SET CACHE EN<CR> - Sets cache look and load.

    SET PMA<CR> - Forces the PMA (physical memory address) to the error address register.

    SET EBR #<CR> - Loads the executive base register with the number (#) specified.

    NOTE
    Channel diagnostics always set the EBR to 3.

23. CLR - The CLR command is the complement of the SET command.

    CLR CHAN #<CR> - Eliminates the default channel and reinstates channel number prompting.

    CLR CACHE EN<CR> - Disables cache look and load.

    CLR PMA<CR> - Discontinues the forcing of the PMA to the error address register.

    CLR EBR - Not implemented.

24. RM<CR> - The RM command performs a master reset; clears the diagnostic CRAM address register; and performs 35 MBox clocks. The RM command is similar to the KLDCP SM command except the clock is not left running.

    NOTE
    Functions set by the SET command are also performed each time the RM command is executed.

25. CE 2,100<CR> - The CE command deposits the "initial command word" specified (100) in the executive page table (EPT) for the channel specified (2). The location will be the executive base register (EBR) location specified by a SET command plus four times the channel number. The next location, STATUS 1 will be cleared.

26. SC 1,START<CR> - The SC command uses the diagnostic function write 70 (FW 70 DATA) to simulate a command from the RH20. The data to be used for the write function may be specified as a 36-bit word (DIAG FUNCT 70) or as the signal mnemonic.

EBus Bit	Mnemonic
06	RESET
07	START
09	DONE
10	CTOM
11	STORE
12	SLOW REQ

Channel timing is synchronized to the proper scan point as a function of the SC command.

NOTE

The SC command should not be used in conjunction with the TM command.

27. QC 1,STA,RES<CR> - The QC command sets up a list of CBus commands for later execution. The purpose of the QC command is to defer CBus activity until a T1 or TM command is executed. (The memory reference trace feature provided by this command may miss printing some memory references unless the timing of the channel scan is coordinated with the memory trace.)

NOTE

The QC command accepts the same arguments as the SC command.

28. T1<CR> - The T1 command traces and prints memory references one at a time so that timing synchronization of CBus requests may be provided.

NOTES

1. The T1 command causes the timing to revert to single-pulse mode.

2. The T1 command is normally used in conjunction with the QC command.

29. TM<CR> - The TM command traces and prints the condition of memory requests and the physical memory address at each SBus address hold time.

NOTE

Notes 1 and 2 under the T1 command apply to the TM command as well.

30. CH<CR> - The CH command prints the default channel number selected by the SET command. Prints NO DEFAULT if prompting for the channel is in effect.

31. NC<CR> - The NC command updates the default channel selected. If no channel default has been set, the default will be channel 0; otherwise, the channel will be incremented. An error indication will be typed if an attempt is made to default to a channel greater than 7.

32. CU<CR> - The CU command uses the standard cache look and load algorithm (least recently referenced data is overwritten). All four caches are loaded.

NOTE

The CU command should be immediately followed by an IC command.

33. C #<CR> - The CU command uses the standard cache look and load algorithm (least recently referenced data is overwritten). However, only the specified cache (#) is loaded.

NOTE

The C# command should be immediately followed by an IC command.

34. IC<CR> - The IC command invalidates the contents of cache (clears cache valid bit).

35. VC<CR> - The VC (validate core) command writes the contents of cache to core.

36. I<CR> - The I command sets the clock step (tick) counter to 0.

37. B 29<CR> - The B command bursts the clock the specified number of times (29) and prints the difference between the initial and final state of the machine.

38. C<CR> - The C command continues the clock and prints the machine state changes at each tick. This is accomplished by single-stepping (if the trace was initiated by a G command) or by incremental bursting (if the trace was initiated by an M command). In both cases, the initial state of the machine is assumed to be that stored from the last interrupted G or M command.

NOTE

Typing an altmode ($) during a trace printout will stop the printout at the end of the current line. Typing a C command will continue the printout.

39. F 14<CR> - The F command single-clocks the CPU the specified number of ticks (14) and prints the difference between the initial state of the machine and the state of the machine after the final (14th) tick.

40. G<CR> - The G command:
    a. resets the tick counter to 0;

    b. reads the initial state of the machine;

    c. steps the clock once;

    d. reads the new state; and

    e. compares the previous state against the new state and prints the difference.

    Steps 3 through 5 are continuously repeated until the user interrupts by typing an altmode ($), or until the EBox transitions to a halted state, or until no changes are detected within a specified number of ticks. (Refer to TRACON Switches Number 14.)

41. S<CR> - The S command single-pulses the clock and prints the machine state changes.

42. P<CR> - The P command prints an EBus bit activity summary and resets the EBus bit activity accumulator registers. Two accumulators are kept for each group of eight diagnostic read functions (i.e., 100-107, 110-117, 120-127, etc.). One accumulator maintains a logical AND for that group; the other, a logical OR. The P command prints out all these accumulators by diagnostic "read function group" plus a total accumulation for all 64 diagnostic functions. In the AND word, if a bit is a 1, then it was always high; in the OR word a 0 bit was always low. (This should be the case with any bits not assigned to a diagnostic function read group.)

43. R<CR> - The R command reads the current state of the machine, compares it against the previously stored state, and prints the difference. The R command allows the user to execute KLDCP commands and then monitor the machine state change (i.e., execute a KLDCP command followed by an R command).

44. D<CR> - The D command reads and prints the current state of the machine. It also prints the EBus bit statistics and resets the accumulators as in the P command.

45. **W CRASH<CR>** - The W command writes a crash file for later use. The file specified (CRASH) must already exist on the output device.

46. D CRASH<CR> - The D command reads in and prints the file (CRASH) saved by the W command.

47. **Examine and Deposit Commands** - Unlike the KLDCP examine deposit commands, the TRACON examine/deposit commands do not use the PI system and do require that the KL10 be halted. They are implemented by executing instructions from the AR which load the ACs and move data to and from memory.

NOTE

Because prompting is in force, a second carriage return is required to reexamine the last address used.

48. **EX instruction**<CR> - The EX command causes the instruction specified to be placed in the AR and executed by the KL10. This command is similar to the EX command supported by KLDCP; however, breakpoint function may be used with the TRACON version.

GENERAL INFORMATION

Code	DDQDC.SAV
Title	DIAMON - DECsystem-10 Diagnostic monitor

Abstract

DIAMON is the basic 10/10 diagnostic monitor. It runs in either exec or user mode. In exec mode DIAMON can load and sequence program from any of the following.

Paper tape
DECtape (either PDP-10 or PDP-11 format)
Disk pack (using a TOPS-10 file structure).

In user mode, DIAMON will run under TOPS-10 (only). The load medium is restricted to disk.

DIAMON is command-controlled and can be directed to load and run a single program or execute a control file which will direct DIAMON to run a sequence of programs. Control files enable DIAMON to be used for the following purposes.

Rapid checkout of the hardware
Acceptance testing
Reliability testing
Unattended overnight testing

DIAMON also supports KI10 margining and special user mode operations.

Hardware Required

KA10, KI10 or KL10 mainframe/32K of core (minimum)/load device: paper tape, DECtape, disk, or console load device (KL10 only).

Preliminary and Associated Programs

DIAMON assumes that the basic instructions and the selected load device are operational.

Restrictions

The diagnostic monitor may be used to call only those programs which follow the prescribed diagnostic formats.

Notes

1. If the monitor fails to operate, use the diagnostic programs individually to isolate the problem.
2. The DECsystem subroutine program and DDT are automatically loaded on system startup or device specification if they are not already resident in the PDP-10 memory.

Loading and Starting Procedure

Standard (Refer to the 10/10 STD module.)

Control Switches

The state of the control switches does not affect the operation of DIAMON unless a control file is being used. A control file lists, as part of each command line, the program to be run and the right half switches to use with that program. This allows the actual (console) right hand switches to be used to control the operation of DIAMON. The switches which affect the operation of DIAMON when a control file is in use are listed in Table 13.

OPERATIONAL CONTROL
After the diagnostic monitor is started it will type the following message:

DIAMON - DECSYSTEM DIAGNOSTIC MONITOR

DEV: T, K, D, V, P -

In user mode, the disk is automatically selected and this question is not asked.

Table 14 describes the device selection commands.

After selection of the load device DIAMON will automatically load SUBRTN and KLDDT and print:

DIAMON CMD -

Table 15 describes general DIAMON commands.
Table 16 describes program starting commands.
Table 17 lists DIAMON manual starting addresses.

Table 13 DIAMON Control Switch Summary

Switch	State	Descriptions
9	0	Reduces the iteration count in a control file by a factor of 100 to 1, thus reducing the run time for each program in the file. This is useful for a quick check of the hardware and margining operations.
	1	Each program listed in a control file is run the specified number of iterations.
15	0	Normal operation
	1	Inhibit printing the test title of each program executed by DIAMON.
18	0	Normal operation
	1	Expand the basic command set to include margining and special user mode operations. Refer to the X command, Table15.

Table 14 DIAMON Load Medium Selection Commands

Command	Description
D	D<CR> Indicates to DIAMON that a PDP-10 formatted DECtape is to be used as the load medium. DIAMON will request the DECtape unit number. Type: 0 - 7 to indicate which DECtape unit contains the program(s) to be run. S to direct DIAMON to search all mounted and selected DECtapes to find the program(s).
K	K<CR> Use the load device selected by KLDCP. This response is only valid for KL10-based systems.
P	P<CR> Selects a disk unit as the load medium. DIAMON will request the disk name and the [P,PN] project, programmer number to use as follows: DISK: [P,PN] Typing a <CR> will cause DIAMON to use the default. The default to DISK will cause a pack search from KLAD and DSKA to DSKO. The default [P,PN] is [6, 10].

Table 14 DIAMON Load Medium Selection Commands

Command	Description
T	T<CR> Selects the paper tape reader as the load medium.
V	V<CR> Indicates to DIAMON that a PDP-11 formatted DECtape is to be used as the load medium. DIAMON will request the DECtape unit number. Type: 0 - 7 to indicate which DECtape unit contains the program(s) to be run. S to direct DIAMON to search all mounted and selected DECtapes to find the program(s).

Table 15 Diamon General Command Summary

Command	Description
<CR>	Standard command terminator.
$	Altmode - a special command terminator which causes a single program to be loaded but not started.
↑Z	A control Z is used to terminate the T command.
D	D<CR> Directs DIAMON to read a control file from the load medium. DIAMON will respond by printing FILE.EXT-. Respond by typing the name of the control file.
F	F<CR> Directs DIAMON to print a directory of the load medium.
G	G<CR> Directs DIAMON to start or restart execution of the program currently loaded in core.
I	I<CR> Directs DIAMON to begin execution of the control file currently in core.
L	L<CR> Directs DIAMON to print a file stored on the load medium. DIAMON will request the name of the file to be printed by printing FILE.EXT-.
R	R<CR> Directs DIAMON to reinitialize itself. DIAMON will begin by requesting the load medium to be used.
S	S<CR> Directs DIAMON to load a single program. DIAMON will request the name of the program by printing FILE.EXT-. This question may be answered with only the file name as the extension will default to .A10 or .SAV unless the console load device is selected, and then the default is .A10.
T	T<CR> Directs DIAMON to open a buffer and begin building an internal control file. A control Z (↑Z) terminates the T command. Refer to section on building control files which follows Table 5.
X	X<CR> Directs DIAMON to run through the expanded command set dialogue.

EXPANDED COMMAND DIALOGUE

The following additional command sequences are added when either the X command is used or the XPAND switch (18) is set.

TYPE Y FOR CLOCK MARGINS -

<CR> = no margins
Y<CR> = clock margins

Clock margins speeds up the basic clock cycle by 10 percent.

TYPE Y OR A FOR SPECIAL USER MODE -

<CR> = no
A<CR> = special user mode after first pass
Y<CR> = special user mode on all passes

Special user mode is a pseudo-user mode where the diagnostic program being run is run in user mode (with paging, etc.) and the I/O is trapped back to the diagnostic monitor for processing. This provides a method of checking user mode operation with functional and reliability diagnostics without actually having to use a monitor and timesharing.

The following additional question is asked if special user mode is selected.

TYPE Y OR A FOR CONCEALED MODE -

<CR> = public mode
Y<CR> = concealed mode
A<CR> = alternate public and concealed modes

TYPE L, S, I, R FOR VOLTAGE MARGINS -

<CR> = No margins

L<CR> = Limits. Margins are done at the $\pm$.25 V settings only.

S<CR> = Sweep margins. The 5 V power supplies are varied by 1-increment steps (21 MV) up to and down to the $\pm$.25 V limits.

I<CR> = Increment. The user may specifiy the margin step per program pass. If I is typed then the following question will be asked.

SPECIFY MARGIN INCREMENT (1 TO 17) -

One increment step equals 21 millivolts (e.g., 4 would specify an 84-millivolt increment).

R<CR> = Rack. Sweep margins are run on the processor logic rack specified. The following question is asked.

SPECIFY RACK (0 to 37) -

Type rack number to be margined.

Margins provide the control necessary to operate the KI10 programmable margin system. Answer the above question(s), then:

SET 'MARGIN SELECT' OFF
SET 'MARGIN ENABLE' SWITCH
TYPE ANY CHAR WHEN READY!

Table 16 DIAMON Program Starting Commands

Command	Description
DDT	DDT<CR> Start DDT
PFSTRT	PFSTRT<CR> Power fail restart
REE	REE<CR> Reenter (user mode)
SFSTRT	SFSTRT<CR> Special features start.
START	START<CR> Start diagnostic
START#	START3<CR> Special start. Numbers range from 1 through 5.
STD	STD<CR> Start diagnostic
STL	STL<CR> Start DIAMON
STM	STM<CR> Reinitialize start

Table 17 Standard Manual Starting and Restarting Addresses

Address	Description
20000	DIAMON starting address
20001	If it is desired to abort a test currently in progress or to restart at the next sequential program, the operator may do so by starting at location 20001.
20002	If the diagnostic monitor is running in the mode where titles are not printed [SW 15(1)] and a user program fails such that it is not known which program failed, starting at location 20002 will cause the title to be printed. The computer will then halt at location 20000. The operator may at this time manually restart the user program or restart the diagnostic monitor.
20003	Program starting and restarting address.

DIAMON CONTROL FILES

A control file for DIAMON is an ASCII file consisting of a list of programs to be run. The following apply to constructing a DIAMON control file.

1. A control file can be constructed with any editor program or via the DIAMON T command.

2. A control file can have up to 50 command lines.

3. Each command line consists of five items each separated by a space or tab. The items are as follows.

   a. Program name. If the program name includes an extension, the extension must be included and separated by a period.

      NOTE

      If the special user mode routines are selected, a line that starts with a minus (-) signifies that the program will run in special user mode.

   b. Pass count. The pass count is the number of passes that the program is to run. The pass count may be in the range 0 to 777777. If 0, the program will run on each pass through the control file.

   c. Switches. This is an octal half word (6 digits) to be used by the program as the right half of the console data switches.

   d. Iterations. This is the number, in octal, of iterations the program is to be executed. The iteration count may be in the range 0 to 377777. If 0, one iteration is assumed.

   e. <CR>. A carriage return terminates the command line and opens the next line for input. If the T command was used to build the control file, a ↑Z (control Z) will close the file and return to DIAMON command mode.

   Example:

```
DEKAA.A10    10    0         1000<CR>
DEKAB.A10     1    123456    200<CR>
DEKAC.A10     0    00001     1<CR>
↑Z
```

4. If the control file is being generated via the T command the following headers will be printed. These act as a guide only and are not actually a part of the control file.
   NAME   PASSES   RH SWS   ITERATIONS

5. Typing errors may be corrected by typing a RUBOUT. The RUBOUT will print three Xs and delete the entire line.

The control file is executed via the I command. The diagnostic monitor will read in and execute the first program on the command list. The program will be iterated the requested number of times and control will then revert to the monitor. The monitor will then proceed to the next program on the list until all programs requested have been executed. When the final program on the command list has been executed, the pass count will be printed and then the monitor will restart with the first program again.

EXAMPLE:

DIAMON PASS 000001
DIAMON PASS 000002
etc.

A control file will remain in core so that if the monitor is restarted the command list does not have to be read in again unless a new control file or single program is selected.

To use the same control file type I.

DIAMON ERROR SUMMARY

CMD'S REQUIRED
The program was commanded to execute the control file, but the list is empty. Input some programs to execute.

Disk Pack Errors
Any disk pack errors will print out the reason, ERROR AT and the octal address of the error. Consult the listing for error explanation.

Margin Errors
If margins are selected and the MARGIN ENABLE switch is not set on startup, the margin setup message will be repeated.

'MARGIN ENABLE' NOT SET?
If during a margin run the MARGIN SELECT switch is reset, the above message will be printed. All subsequent programs will be run in normal mode.

MUUO ERROR
If the diagnostic program being run (in special user mode) causes an MUUO, (not trapped I/O) the above error message will be printed and the program will halt. The operator may examine the user MUUO locations (17424 and 17425) to determine the cause of the error.

PROGRAM NOT FOUND - PROG.EXT
The program requested is not on the load device.

USER TRAP ERROR
If the diagnostic program being run (in special user mode) causes a trap (PAGE FAIL, PUSHDOWN OVERFLOW or TRAP 3), the above message will be printed and the program will halt.

GENERAL INFORMATION

DDT (Dynamic Debugging Technique) is a utility program for on-line checkout, testing, and control of MACRO and FORTRAN programs. A modified version of DDT is always loaded with the 10-based 10 diagnostic routines. Many of these diagnostics use DDT for command interpretation and test dispatching (e.g., a diagnostic which uses an $G following a test identification (FRTEST$G) is actually using a DDT feature to dispatch to the starting address of the test). DDT supports many commands which are useful for controlling diagnostics during maintenance.

DDT<CR>	KLDCP and DIAMON command to start DDT
<CR><LF>	PROMPT - DDT uses a carriage return followed by a line feed to indicate it is ready to accept a command.
$G	Exit DDT - Begin execution of main (diagnostic) program.
NOTES	1. This module summarizes the most commonly used DDT commands. Refer to the Software Notebooks for a complete list of commands. 2. Use symbolic location PATCH for building special test routines or patching the main program.

DATA AND COMMAND FORMATS

DDT has two primary data formats: symbolic and halfword.

```
SYMBOLIC:  CAT+2/ MOVE 3,500
HALFWORD:  CAT+2/ 200140,,500
```

Table 18 describes the data format field delimiters.

Table 19 summarizes the DDT commands.

Table 20 summarizes DDT error messages.

Table 18 DDT Field Delimiters

Delimiter	Description
space	A space delimits the op-code field.
,	A comma delimits the AC field.
( )	Parentheses delimit the index field.
@	The @ symbol indicates indirect addressing.
,,	Double commas delimit half words.

Table 19 DDT Command Summary

Command	Description
Special Editing Commands	
rubout	The rubout key will cause the last character typed to be deleted.
↑U	(Control U) Delete line.
↑W	(Control W) Delete last word, back to delimiter.
↑R	(Control R) Retype last line.
Arithmetic Operations	
+	117+123<CR> Addition
-	51-17<CR> Subtraction
*	15*12<CR> Multiplication
'	256'16<CR> Division
Radix	
$nR	$8R Set the base radix to n.
Address Modes	
$A	Set address mode to absolute numeric.
$R	Set address mode to relative symbolic.
Printout Modes	
$H	Set printout mode to halfword.
$S	Set printout mode to symbolic.
$T	Set printout mode to ASCII text.
6$T	Set printout mode to sixbit text.
Searching	
a<b>c$W	2000<2050>MOVE$W Search for the key word "c." Begin the search at address "a" and end the search at address "b."
Symbols	
.	A period represents the symbolic value of the position pointer.
$Q	Represents the last quantity typed.
@	Represents the indirect bit.
name$:	MAIN$: Opens local symbol table for use by DDT. Name equals the name specified in the MACRO-10 title statement. For most diagnostics the title is MAIN.
sym:	CAT: Insert a new symbol in the symbol table. Use the current value of the pointer.
n<sym:	2017<CAT: Insert a symbol in the symbol table. Use the value specified by n.
sym$$K	CAT$$K Delete the specified symbol from the symbol table.

Table 19 DDT Command Summary (Cont)

Command	Description
Breakpoints	
adr$B	4000$B Set a breakpoint at the specified address. Symbolic address may be used.
$P	Proceed from the breakpoint.
n$P	5$P Set the proceed counter to n and proceed from the breakpoint.
$$P	Proceed always.
$B	Remove all breakpoints.
0$nB	0$2B Remove the breakpoint specified by n.
Instruction and Program Execution	
inst$X	MOVE 3,CAT+3$X Execute the specified instruction once.
$X	$X Execute the instruction pointed to. Print the operands and increment the pointer (PC).
n$X	4$X Repeat the $X command n times, printing the operands and incrementing the pointer each time.
n$$X	4$$X Repeat the $X command n times. The operands are printed for the last executive only.
$G	Start the program at the normal starting address (JOBSA).
adr$G	2050$G<CR> Start the program at the specified address.
Input Formats	
inst	MOVE AC4, CAT+3 Format for inputting a symbolic instruction.
#,,#	777000,,000777 Format for inputting half words.
#	14 Format for inputting octal digits.
#.	94. Format for inputting decimal digits.
#.#	273.5 Format for inputting a floating point number.
"/A/	"/THIS IS A MESSAGE/ Format for inputting ASCII text.
"A$	"Y$ Format for inputting one ASCII character.
$"/A/	$"/THIS IS A MESSAGE/ Format for inputting sixbit ASCII text.
$"A$	$"Y$ Format for inputting one sixbit ASCII character.

Table 19 DDT Command Summary (Cont)

Command	Description
Examine and Modify Locations	
adr/	CAT/<CR> Print contents at address and leave open for modification.
adr!	CAT!<CR> Open address for modification but do not print current contents.
adr[	MASK[<CR> Print contents of address as a numerical value. Leave open for modification.
adr]	Print symbolic contents of address. Leave open for modification.
^ (BACKSPACE)	Examine address location minus one
TAB	Examine location specified by address
$<	A patch is made by opening an address, typing (ALTMODE) (ANGLE-BRACKET). This saves the current contents of the address and opens the patch area for new instructions. After the new instructions are entered, the patching is closed by typing (ALTMODE) (ANGLE-BRACKET). The original contents are then placed in the patch area followed by two jump instructions which will return to the original address +1 or +2, depending on whether the last instruction in the patch skips or not. Example: ADDRESS/contents $< PATCH/new instruction PATCH +1/new instruction #> PATCH +2/contents PATCH +3/jump 1, ADR +1 PATCH +4/jump 2, ADR +2
line feed	Typing a line feed will close the current address and cause the contents of the next sequential address to be printed. The address will be left open for modification. Up arrow will cause the contents of the last address specified minus one to be printed. The address is left open for modification.
Carriage return	Typing a carriage return will clear the currently open address. If modifications were made the new contents are inserted.
Repeating Printouts in Modes Other Than Prevailing or Temporary	
=	Typing the = symbol following a symbolic printout will cause the printout to be repeated in halfword format.
-	Typing a dash (-) following a halfword printout will cause the printout to be repeated in symbolic format.
/	Typing the / symbol will print out the location pointed to but will not change the pointer.
[	Typing the [symbol will print out the location pointed to as a numeric value.
]	Typing the] symbol will print out the location pointed to as a symbolic instruction.

Clear Memory	
adr<adr$$Z	PATCH<PATCH+20$$Z Clear memory from address to address.

Table 20 DDT Error Messages

Error	Description
U	Indicates the user typed an undefined symbol which cannot be interpreted by DDT. Everything typed by the user since the last DDT printout is ignored.
?	Indicates an illegal DDT command has been typed or a location outside of the user's assigned memory area has been referenced.

D20MON

GENERAL INFORMATION

Code	DDQDH.EXE
Title	DECSYSTEM-20 Diagnostic Monitor
Abstract	D20MON is a variation of DIAMON which has been modified to handle TOPS-20 file structures. It will run in either exec or user mode. In exec mode D20MON can load and sequence programs from disk or the KLDCP load device. In user mode the load device is restricted to disk only.
Hardware Required	KL10 mainframe/32K of core (minimum)/load device: 1. KLDCP - KL10 only, use KLDCP selected device. 2. Disk pack, RP04/5/6 RM03 on RH10 or RH20.
Preliminary and Associated Programs	D20MON assumes that the basic instructions and the selected load device are operational.
Restrictions	The diagnostic monitor may be used to call only those programs which follow the prescribed diagnostic formats.
Notes	1. If the monitor fails to operate, use the diagnostic programs individually to isolate the problem. 2. The DECsystem SUBRTN package and DDT are automatically loaded on system startup or device specification if they are not already resident in the PDP-10 memory.
Loading and Starting Procedure	Via KLDCP type: P D20MON<CR> STL<CR> Via TOPS-20 type: RUN D20MON<CR>
Control Switches	The state of the control switches does not affect the operation of D20MON unless a control file is being used. A control file lists, as part of each command line, the program to be run and the right half switches to use with that program. This allows the actual (console) right-hand switches to be used to control the operation of D20MON. The switches which affect the operation of D20MON when a control file is in use are listed in Table 21.

OPERATIONAL CONTROL
After the diagnostic monitor is started it will type the following message:

D20MON - DECSYSTEM-20 DIAGNOSTIC MONITOR
DEV:

In user mode, the disk is automatically selected and this question is not asked.

Table 22 describes the device selection commands.

When the disk pack is selected as the load device the monitor operates from the DISK:<DIRECTORY> that is specified. The default disk is PS: and the default directory is <DIAGNOSTICS>. To use the defaults type a <CR>.

After selection of the load device, D20MON will automatically load SUBRTN and KLDDT and print:

D20MON CMD -

Table 23 describes general D20MON commands.
Table 24 describes program starting commands.
Table 25 lists D20MON manual starting addresses.

Table 21 D20MON Control Switch Summary

Switch	State	Description
9	0	Reduces the iteration count in a control file by a factor of 100 to 1, thus reducing the run time for each program in the file. This is useful for a quick check of the hardware.
	1	Each program listed in a control file is run the specified number of iterations.
15	0	Normal operation
	1	Inhibit printing the test title of each program executed by D20MON.
18	0	Normal operation
	1	Expand the basic command set to include margining and special user mode operations. Refer to the X command, Table 23.

Table 22 D20MON Device Selection Commands

Command	Description
K<CR>	Use the load device selected by KLDCP.
?<CR>	Will cause a list of all available disk structures to be printed.
dev:<CR>	Use the disk specified by dev as the load medium (e.g., KLAD20:<CR>).
dev:?<CR>	Will cause the master directory for the disk specified by dev: to be printed.
<CR>	Will default to the public structure (same as typing PS:<CR>).

Table 23 D20MON General Command Summary

Command	Description
<CR>	Standard command terminator.
$	Altmode - a special command terminator which causes a single program to be loaded but not started.
↑Z	A control Z is used to terminate the T command.
D	D<CR> Directs D20MON to read a control file from the load medium. D20MON will respond by printing FILE.EXT-. Respond by typing the name of the control file.
F	F<CR> Directs D20MON to print a directory of the load medium.
G	G<CR> Directs D20MON to start or restart execution of the program currently loaded in core.
I	I<CR> Directs D20MON to begin execution of the control file currently in core.
L	L<CR> Directs D20MON to print a file stored on the load medium. D20MON will request the name of the file to be printed by printing FILE.EXT-.
R	R<CR> Directs D20MON to reinitialize itself. D20MON will begin by requesting the load medium to be used.

Table 23 D20MON General Command Summary (Cont)

S	S<CR> Directs D20MON to load a single program. D20MON will request the name of the program by printing FILE.EXT-. This question may be answered with only the file name as the extension will default to .A10 or .SAV unless the console load device is selected and then the default is .A10.
T	T<CR> Directs D20MON to open a buffer and begin building an internal control file. A control Z (↑Z) terminates the T command. Refer to section on building control files which follows Table 25.
X	X<CR> Directs D20MON to run through the expanded command set

EXPANDED COMMAND DIALOGUE

The following additional command sequences are added when either the X command is used or the XPAND switch (18) is set.

TYPE Y OR A FOR SPECIAL USER MODE -

<CR> = no
A<CR> = special user mode after first pass
Y<CR> = special user mode on all passes

Special user mode is a pseudo-user mode where the diagnostic program being run is run in user mode (with paging, etc.) and the I/O is trapped back to the diagnostic monitor for processing. This provides a method of checking user mode operation with the functional and reliability diagnostics without actually having to use a monitor and timesharing.

The following additional question is asked if special user mode is selected.

TYPE Y OR A FOR CONCEALED MODE -

<CR> = public mode
Y<CR> = concealed mode
A<CR> = alternate public and concealed modes

Table 24 D20MON Program Starting Commands

Command	Description
DDT	DDT<CR> Start DDT
PFSTRT	PFSTRT<CR> Power fail restart
REE	REE<CR> Reenter
SFSTRT	SFSTRT<CR> Special features start
START	START<CR> Start diagnostic
START#	START3<CR> Special start. Numbers range from 1 through 5.
STD	STD<CR> Start diagnostic
STL	STL<CR> Start D20MON
STM	STM<CR> Reinitialize start

Table 25 Standard Manual Starting and Restarting Addresses

Address	Description
20000	D20MON starting address
20001	If it is desired to abort a test currently in progress or to restart at the next sequential program, the operator may do so by starting at location 20001.
20002	If the diagnostic monitor is running in the mode where titles are not printed [SW 15(1)] and a user program fails such that it is not known which program failed, starting at location 20002 will cause the title to be printed. The computer will then halt at location 20000. The operator may at this time manually restart the user program or restart the diagnostic monitor.
20003	Program starting and restarting address.

D20MON CONTROL FILES

A control for D20MON is an ASCII file consisting of a list of programs to be run. The following apply to constructing a D20MON control file.

1. A control file can be constructed with any editor program or via the D20MON T command.

2. A control file can have up to 50 command lines.

3. Each command line consists of five items each separated by a space or tab. The items are as follows.

   a. Program name. If the program name includes an extension, the extension must be included and separated by a period.

      NOTE

      If the special user mode routines are selected, a line that starts with a minus (-) signifies that the program will run in special user mode.

   b. Pass count. The pass count is the number of passes that the program is to run. The pass count may be in the range 0 to 777777. If 0, the program will run on each pass through the control file.

   c. Switches. This is an octal half word (6 digits) to be used by the program as the right half of the console data switches.

   d. Iterations. This is the number, in octal, of iterations the program is to be executed. The iteration count may be in the range 0 to 377777. If 0, one iteration is assumed.

   e. <CR>. A carriage return terminates the command line and opens the next line for input. If the T command was used to build the control file a ↑Z (control Z) will close the file and return to DIAMON command mode.

   Example:

   ```
   DFKAA.A10   10   0        1000<CR>
   DFKAB.A10    1   123456   200<CR>
   DFKAC.A10    0   00001    1<CR>
   ↑Z
   ```

4. If the control file is being generated via the T command, the following headers will be printed. These act as a guide only and are not actually a part of the control file.

   ```
   NAME   PASSES   RH SWS   ITERATIONS
   ```

5. Typing errors may be corrected by typing a RUBOUT. The RUBOUT will print three Xs and delete the entire line.

The control file is executed via the I command. The diagnostic monitor will read in and execute the first program on the command list. The program will be iterated the requested number of times and control will then revert to the monitor. The monitor will then proceed to the next program on the list until all programs requested have been executed. When the final program on the command list has been executed, the pass count will be printed and then the monitor will restart with the first program again.

Example:

D20MON PASS 000001
D20MON PASS 000002
etc.

A control file will remain in core so that if the monitor is restarted the command list does not have to be read in again unless a new command list or single program is selected.

To use the same control file, type I.

D20MON ERROR SUMMARY

CMD'S REQUIRED
The program was commanded to execute the control file, but the list is empty. Input some programs to execute.

DISK PACK ERRORS
Any disk pack errors will print out the reason, ERROR AT, and the octal address of the error. Consult the listing for error explanation. Disk ECC errors are automatically corrected.

MUUO ERROR
If the diagnostic program being run (in special user mode) causes an MUUO, (not trapped I/O) the above error message will be printed and the program will halt. The operator may examine the user MUUO locations (17424 and 17425) to determine the cause of the error.

PROGRAM NOT FOUND - PROG.EXT
The program requested is not on the load device.

USER TRAP ERROR
If the diagnostic program being run (in special user mode) causes a trap (PAGE FAIL, PUSHDOWN OVERFLOW or TRAP 3), the above message will be printed and the program will halt.

Table of Contents

SYSTEM SOFTWARE
System software is one of the three major categories of software. Refer to Figure 1.

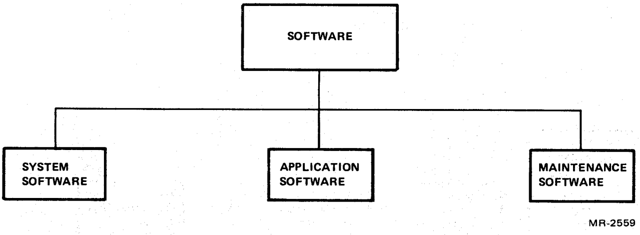


Figure 1 Three Major Categories of Software

System software consists of an operating system or monitor and a library of Commonly Used System Programs (CUSPs). Refer to Figure 2.

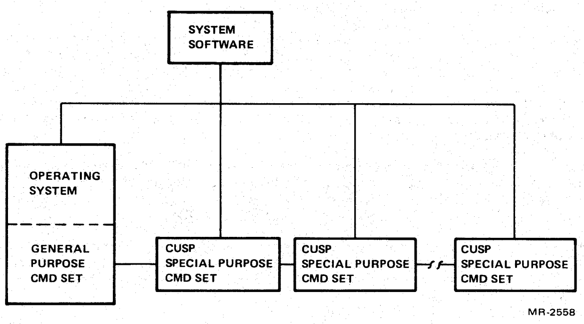


Figure 2 Component Parts of System Software

The operating system directs and monitors the overall performance of the system and supports a general purpose command set. The CUSPs, in effect, extend the general purpose command set by supporting individual special purpose command sets.

SYSTEM MONITORING - Directing and monitoring the overall performance of the system is the most complex aspect of an operating system. It involves tasks such as scheduling jobs for execution, directing I/O operation, handling interrupts, and managing system resources. Although field maintenance personnel should have an overall understanding of this aspect of operating systems, an in-depth knowledge is not generally required.

COMMONLY USED SYSTEM PROGRAMS (CUSPs) - The number and type of CUSPs associated with a given system program library depends largely on the intended use of the system. Regardless of the intended use of the system, however, the relationship between the operating system and the CUSPs in the corresponding system program library will remain the same. That is, the operating system will support a set of general purpose commands and each CUSP will support a unique set of special purpose commands. Refer to Figure 3.

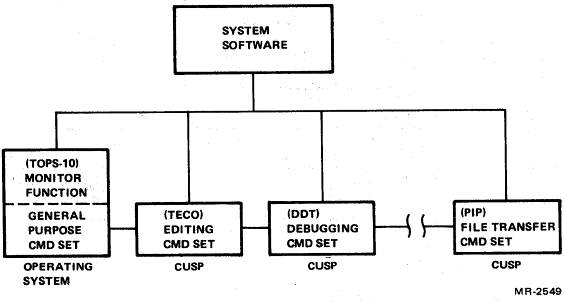


Figure 3 Typical Operating System and CUSP Relationship

Figure 3 uses the TOPS-10 operating system and three CUSPs from the TOPS-10 system program library to illustrate the relationship between operating systems and CUSPs.

The general purpose command set supported by the operating system enables system programmers, operators and users to perform the following functions: gain access to the system, run existing system and application software, communicate with system operators or other users on the system, request system resources and operator services as needed, and gather information concerning job and system performance.

Three of the CUSPs which extend or supplement the TOPS-10 general purpose command set are described below. Note that the CUSP command set is selected for use via one of the general purpose commands, usually GET or RUN (e.g., RUN TECO<CR>).

The Text Editor and COrrector (TECO) supports commands which enable the user to build and edit an ASCII text file. Later, this file may be transformed into a usable program via an assembler or compiler-type CUSP.

The Dynamic Debugging Technique (DDT) supports a command set which allows the user to test and debug his program on-line before putting it into operation.

The Peripheral Interchange Program (PIP) supports commands which enable a user to copy or transfer files between standard peripheral devices.

For field maintenance personnel, command sets are the simplest and most important aspect of system software. Some skill and proficiency in using system software is essential to field maintenance personnel because system software must be used to maintain on-line file storage areas, run on-line (user mode) utility and diagnostic programs, and compile and print system error logs.

System Software Command Format

Operating systems and system library programs use a command format similar to the one illustrated in Figure 4.

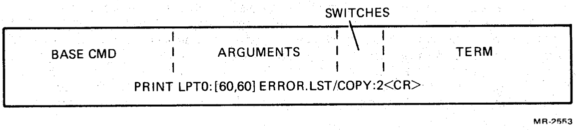


Figure 4 Typical System Software Command Format

BASE CMD - The base command is usually a verb which describes the task that the command will accomplish (e.g., GET, RUN, PRINT, etc.).

ARGUMENTS - The arguments specify the base command parameters. For example, the arguments supplied to the PRINT command illustrated in Figure 4 specify LPT0: as the output device, [60,60] as the project programmer numbers, and ERROR.LST as the file to be printed.

SWITCHES - Switches cause a minor modification to the basic action of the command. For example, the COPY:2 switch illustrated in Figure 4 will cause two copies of the file ERROR.LST to be printed instead of one, which is the default. For example, DIRECT [60,60]/FAST<CR>. The FAST switch associated with the DIRECTORY command will cause an abbreviated form of the directory area to be printed.

TERM - The command terminator, usually a carriage return <CR>, line feed <LF> or altmode <$>, directs the operating system or CUSP to execute the command. As a result of executing the command illustrated by Figure 4, line printer 0 will print two copies of the file ERROR.LST, which is stored in the [60,60] project programmer area of the default input device (in this case the system disk).

Although some system software commands do not require all of the command elements described above, and some will prompt for missing arguments, the basic format (BASE CMD ARGUMENTS SWITCHES TERM) will generally remain the same for all system software. Thus, learning to use system software is a relatively easy task.

Tips on Learning to Use System Software

The following are some tips you may find helpful when learning to use new system software.

1. Study the file structure and organization used by the operating system. This is important because many system software commands are related to file generation, modification and manipulation.

2. Think of system software in terms of command sets. Do not become overly concerned with the monitoring function.

3. Think of each command individually in terms of what task it will accomplish. Do not become overly concerned with how the command achieves the task.

4. Review the general purpose command set supported by the operating system. Become familiar with the type of commands that are available.

5. Review the abstract and command set associated with each CUSP in the system program library. Determine which CUSPs you are most likely to use on a regular basis.

6. Design some exercises which will help you develop skill and proficiency in using the system software. Remember perfect practice makes perfect.

7. Finally, and most important - DON'T be intimidated by system software. It is designed to be easy to use and there are a lot of people using it that know far less about computers than you do.

Table of Contents

RSX-20F SYSTEM PROGRAM LIBRARY

The RSX-20F System Program Library consists of four kinds of files.

Microcode files. These files are for the KL10 and are listed and described in Table 1.

Boot files. These files are used by the front-end subsystem to boot the KL10. They are listed and described in Table 2.

Automatic task files. These files are used by RSX-20F for various housekeeping tasks and are not normally loaded by the user. They are listed and described (for reference purposes) in Table 3.

User task files. These files are listed and described in Table 4.

Table 1 RSX-20F System Program Library Microcode Files

Task	Description
KLA.MCB	Microcode file for KL10 model PAs.
KLX.MCB	Microcode file for KL10 model PVs.

Table 2 RSX-20F System Program Library Boot Files

Task	Description
BOOT.EXB	Boot Boots KL10 monitor system image into KL's core from RIGID disk; is written in executable binary KL code.
MTBOOT.EXB	Magtape Boot Allows transfer of a program's core image from magtape into KL10's core; is written in executable binary KL code.

Table 3 RSX-20F System Program Library Auto Tasks

Task	Description
F11ACP.TSK	Files-11 Ancillary Control Processor File handler for front-end disk files (performs file access, management, and control functions).
KLE.TSK	KL Error Error processing of KL10 errors. Uses diagnostic DTE functions. Produces "snapshot" of KL10 error conditions for troubleshooting. Calls KLINIT when done.
KLI.TSK	KL Initialization Initializes the KL10 processor (produces installation dialogue, loads microcode, runs bootstrap, etc.). Called whenever system comes up.
KLR.TSK	KLINIK Request Checks KLINIK time window and KLINIK password when KLINIK line rings. If they are correct, it then enables KLINIK.

Table 3 RSX-20F System Program Library Auto Tasks (Cont)

Task	Description
KLX.TSK	KL Transfer Transfers KLEER.SNP to SYSERR file in KL10. (Not to be confused with KLX.MCB, which is the filename of the KL10-PV microcode.)
MIDNIT.TSK	Midnight Roll over time of day at midnight.
SETSPD.TSK	Set Speed Sets line speed table for -10 after restart and sets the time in the -10. NOTE **Do not confuse this with the TOPS-20 program SETSPD.EXE. SETSPD.TSK is a front-end task and it does not access CNFG.CMD.**
TKTN.TSK	Task Termination Program Outputs task termination notification and provides orderly termination for front-end tasks. Interfaces between KLINIT and KLERR (lets KLE call KLI).
T20APC.TSK	TOPS-20 Ancillary Control Processor File handler for files to be transferred to and from the KL10's disk area. Interacts with TOPS-20 area in terms compatible with FILES-11 operations.
UFD.TSK	User File Directory Sets up directories in FILES-11 area. Directories are "named" by a UIC (user identification code) and enclosed in brackets: [X, Y].

Table 4 RSX-20F System Program Library User Tasks

Task	Description
COP.TSK	Copy Floppy disk copy utility. Also allows verification of physical state of the disk, as well as verification of successful copying.
DMO.TSK	Dismount Removes a device from the front-end system's knowledge, making its contents inaccessible to the user.
FEDDT.TSK	Front-End DDT Symbolic debuger for RSX-20F. Permits user to read and print selected portions of front-end crashes.
INT.TSK	Initialize Initializes FILES-11 devices to be recognizable FILES-11 "VOLUMES". Sets up master directory space, index and home blocks, etc.
MOU.TSK	Mount Makes a device known to the system so that it can be accessed by a given user.
PARSER.TSK	Command Parser Primary means of access to front-end programs. Provides access to KL10's memory for diagnostic functions, as well as debugging tools. Will interface with KLINIK in future versions.
PIP.TSK	Peripheral Interchange Program Performs general file transfer and some maintenance functions among FILES-11 devices and other peripherals (e.g., floppy-to-disk file transfers, file deletions, typing directories at console, etc.).
RED.TSK	Redirect Changes front-end system's "home" from one FILES-11 device to another, and tells system where it resides presently.
SAV.TSK	Save Saves core image of front-end on RIGID disk in FILES-11 area.
ZAP.TSK	Zap Permits direct examination and modification of files on a FILES-11 volume. Patch task images and data files in an interactive environment.

GENERAL INFORMATION
The command PARSER runs as a task under the RSX-20F executive. Its primary function is to receive ASCII command strings, usually from the console terminal, and perform console functions on the KL10 or PDP-11 computer.

^\	Control Backslash - Command to RSX-20F to load and run PARSER
PAR>	Prompt - Indicates PARSER is ready to accept commands, and the KL10 clock and run flip-flop are on
PAR%	Prompt - Indicates PARSER is ready to accept commands, the KL10 run flip-flop is off, and the KL10 clock is on
PAR#	Prompt - Indicates PARSER is ready to accept commands, and the KL10 clock is off. This may indicate an error condition
QUIT or ^Z or SET CON/USER	Exit PARSER - Return to RSX-20F command mode. The CTY is connected to the program running in the KL10

Note

1. Commands and arguments may be abbreviated to the simplest form that uniquely identifies them; e.g., the EXAMINE command may be typed as E since no other commands begin with E.
2. The maximum number of characters in a command line is 280.
3. Numeric arguments default to decimal unless they are address or data arguments. Then they default to octal.

COMMAND CONVENTIONS
The command conventions and special characters used by PARSER are described in Table 1.

COMMAND SUMMARY
The command PARSER has four modes of operation. The mode is set by the SET CONSOLE command.

Maintenance Mode - Enables the commands described in Table 2.

User Mode - Connects the console to the program running in the KL10. No PARSER commands are in effect.

For a description of the commands listed in Table 3 and Table 4, refer to Table 2.

Table 1 Command PARSER Special Characters

Character	Meaning
?	PAR>?<CR> or PAR>SET?<CR> A question mark typed at PARSER command, subcommand, or argument level will cause a brief help message to be displayed.
;	PAR>E PC;E 20;SH<CR> Used to separate individual commands within a command line.
!	PAR>REP 5;E PC! SEE IF CPU IS IN HALT LOOP<CR> Indicates a comment line.
<CR>	PAR>SH<CR> Command line terminator - causes the command line to be executed.
-<CR>	PAR>ST M0-<CR> Nullifies the <CR> terminator - allows the command line to be continued on the next line. The continuation line will prompt with another dash.

Table 1 Command PARSER Special Characters (Cont)

Character	Meaning
^C	PAR<DE T 100:^C5<CR> Digits preceded by an up arrow and a C are interpreted as 1's complement.
^D	PAR>DE E 200:^D5<CR> Digits preceded by an up arrow and a D are interpreted as decimal.
^B	PAR>DE T 200:^B1010<CR> Digits preceded by an up arrow and a B are interpreted as binary.
^O	PAR>DE T 200:^O5252<CR> Digits preceded by an up arrow and an O are interpreted as octal (default).
^O	A control O can also be used to suppress printouts.
^Z	A control Z causes PARSER to exit. The console is connected to the program running in the KL10.
'	PAR>E E 34'<CR> A single quote adds the current value of the relocation switch to the number. See SET OFFSET.
"	PAR>E E 34"CR> A double quote subtracts the current value of the relocation switch from the number. See SET OFFSET.
-	PAR>DE T 30:-1<CR> A string of digits preceded by a hyphen (minus sign) is interpreted as the 2's complement of the value of the string.
+-*/	Two numeric expressions separated by plus, minus, asterisk, or slash are evaluated by applying the operations of addition, subtraction, multiplication or division, respectively.
_	Two numeric expressions separated by underscore are evaluated by shifting the first left by the second. Example: 1_3 is 10 octal.
(2*8)/4	Parentheses may be used to enclose expressions. Thus parentheses can be used to change the implicit order of arithmetic operations.

Table 2 PARSER Maintenance Mode Command Summary

Command	Description	Cross Ref.
ABORT	PAR>A<CR> Force the KL10 into the HALT loop. See HALT.	1
CLEAR	PAR>CL arg<CR> The CLEAR command accepts the following arguments. See SET commands. CLOCK e.g., PAR>CL CL CON<CR> The CLEAR CLOCK command accepts the following arguments. CONTROL e.g., PAR>CL CL CON<CR> Disable the control logic clock. CRAM e.g., PAR>CL CL CR<CR> Disable the CRAM clock. DATA-PATH e.g., PAR>CL CL D<CR> Disable the data path clock. EXTERNAL e.g., PAR>CL CL E<CR> Select the internal KL10 clock source. Same as SET CLOCK INTERNAL.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	FULL e.g., PAR>CL CL F<CR> Set the KL10 clock rate to full speed. Same as SET CLOCK FULL.	
	HALF e.g., PAR>CL CL H<CR> Set the KL10 clock rate to full speed. Same as SET CLOCK FULL.	
	INTERNAL e.g., PAR>CL CL I<CR> Select the internal KL10 clock source. Same as SET CLOCK INTERNAL.	
	MARGIN e.g., PAR>CL CL M<CR> Select the internal KL10 clock source. Same as SET CLOCK INTERNAL.	
	NORMAL e.g., PAR>CL CL N<CR> Set the KL10 clock parameters to internal source and full rate with the CRAM, DATA-PATH and CONTROL clocks enabled.	
	QUARTER e.g., PAR>CL CL Q<CR> Set the KL10 clock rate to full speed. Same as SET CLOCK FULL.	
	SLOW e.g., PAR>CL CL S<CR> Set the KL10 clock rate to full speed. Same as SET CLOCK FULL.	
	CONSOLE e.g., PAR>CL C<CR> Put the console front end into operator mode. Equivalent to SET CONSOLE OPERATOR.	
	DATE e.g., PAR>CL D<CR> Clear the date validity bit and prompt for a new date and time. This command is invalid if RSX-20F is in primary protocol; i.e., if the public structure (PS) is mounted. See SET DATE.	
	FS-STOP e.g., PAR>CL FS<CR> Disable the field service clock error stop feature. Same as CLEAR PARITY-STOP FS-STOP.	
	INCREMENT e.g., PAR>CL I<CR> Set the KL10 increment factor to 0. See SET INCREMENT.	
	KLINIK e.g., PAR>CL K<CR> Clear KLINIK parameters (only).	16
	MEMORY e.g., PAR>CL M<CR> Make KL10 memory the default for deposits and examines. Not to be confused with zeroing memory. See SET MEMORY and ZERO.	
	NOT e.g., PAR>CL NO REL<CR> Used with CLEAR to negate the clear function. It is equivalent to SET.	
	OFFSET e.g., PAR>CL O<CR> Set the value of the PDP-11 relocation counter to 0. See SET OFFSET.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	PARITY-STOP e.g., PAR>CL P ALL<CR> The CLEAR PARITY-STOP command accepts the following arguments.	
	ALL e.g., PAR>CL P ALL<CR> Disable all parity stop features.	
	AR e.g., PAR>CL P AR<CR> Disable the AR and ARX parity stop features.	
	CRAM e.g., PAR>CL P C<CR> Disable the CRAM parity stop feature.	
	DRAM e.g., PAR>CL P D<CR> Disable the DRAM parity stop feature.	
	ENABLE e.g., PAR>CL P E<CR> Clear all parity stop enables. Same as CLEAR PARITY-STOP ALL<CR>	
	FM e.g., PAR>CL P FM<CR> Disable the fast memory (FM) parity stop feature.	
	FS-STOP e.g., PAR>CL P FS<CR> Disable the field service clock error feature. Same as CLEAR FS-STOP.	
	RELOAD e.g., PAR>CL REL<CR> Disable the automatic reloading of the KL10 following a fatal error condition.	
	REPEAT e.g., PAR>CL REP<CR> Set the repeat counter to 0. All subsequent command lines will be repeated once. See SET REPEAT.	
	RETRY e.g., PAR>CL RET<CR> Clear the PARSER RETRY flag. Every KEEP-ALIVE-CEASED error will cause a KLERR snapshot before reloading the KL10.	
	TRACKS e.g., PAR>CL T<CR> Clear the KL10 tracking function. See SET TRACKS.	10
CONTINUE	PAR>CO<CR> Continue the KL10 running if it is continuable (i.e., the KL10 has not been reset). See START.	2
DEPOSIT	PAR>DE T N:500<CR> The DEPOSIT command accepts the following arguments. Default: see SET MEMORY. The previous contents of the location or argument specified will be displayed.	
	AR e.g., PAR>DE A:777777777777<CR> Load data (777777777777) into the AR.	
	ELEVEN e.g., PAR>DE E 2000:500<CR> Deposit data (500) into PDP-11 location specified (2000).	
	DEPOSIT ELEVEN accepts the following arguments. Default: THIS.	
	DECREMENT e.g., PAR>DE E D:500<CR> Deposit data (500) into the last PDP-11 location referenced minus two (-2).	
	INCREMENT e.g., PAR>DE E I:500<CR> Deposit data (500) into the last PDP-11 location referenced plus two (+2).	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	NEXT e.g., PAR>DE E N:500<CR> Same as DE E I:500<CR> (INCREMENT) PREVIOUS e.g., PAR>DE E P:500<CR> Same as DE E D:500<CR> (DECREMENT) THIS e.g., PAR>DE E T:500<CR> Deposit data (500) into the last PDP-11 location referenced. THIS is the default. TEN e.g., PAR>DE T 30000:500<CR> Deposit data (500) into PDP-10 location specified (30000). All references are to a physical address. Paged (user) deposits are not supported by PARSER. DEPOSIT TEN accepts the following arguments. Default: THIS DECREMENT e.g., PAR>DE T D:500<CR> Deposit data (500) into the last PDP-10 location referenced minus the increment value. See SET INCREMENT. INCREMENT e.g., PAR>DE T I:500<CR> Deposit data (500) into the last PDP-10 location referenced plus the increment value. See SET INCREMENT. NEXT e.g., PAR>DE T N:500<CR> Deposit data (500) into the last PDP-10 location referenced plus one (+1). PREVIOUS e.g., PAR>DE T P:500<CR> Deposit data (500) into the last PDP-10 location referenced minus one (-1) THIS e.g., PAR>DE T T:500<CR> Deposit data (500) into the last PDP-10 location referenced. THIS is the default.	
DISCONNECT	PAR>DI<CR> Disconnect the KLINIK link by running KLDISC.TSK. The existing KLINIK parameters are not affected. See CLEAR KLINIK.	
EXAMINE	PAR>EX T 3000<CR. The EXAMINE command accepts the following arguments. Default: see SET MEMORY. ELEVEN e.g., PAR>EX EL 3000<CR> Display the contents of the PDP-11 location specified (3000). EXAMINE ELEVEN accepts the following arguments. Default: THIS. DECREMENT e.g., PAR>EX EL D<CR> Display the contents of the last PDP-11 location referenced minus two (-2). INCREMENT e.g., PAR>EX EL I<CR> Display the contents of the last PDP-11 location referenced plus two (+2). NEXT e.g., PAR>EX EL N<CR> Same as EX EL I<CR> (INCREMENT) PREVIOUS e.g., PAR>EX EL P<CR> Same as EX E D<CR> (DECREMENT)	

PARSER

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	THIS e.g., PAR>EX EL T<CR> Display the contents of the last PDP-11 location referenced. THIS is the default.	
	TEN e.g., PAR>EX T 30000<CR> Display the contents of the PDP-10 location specified (30000). All references are to a physical address. Paged (user) examines are not supported by PARSER. EXAMINE TEN accepts the following arguments. Default: THIS.	
	DECREMENT e.g., PAR>EX T D<CR> Display the contents of the last PDP-10 location referenced minus the increment value. See SET INCREMENT.	
	INCREMENT e.g., PAR>EX T I<CR> Display the contents of the last PDP-10 location referenced plus the increment value. See SET INCREMENT.	
	NEXT e.g., PAR>EX T N<CR> Display the contents of the last PDP-10 location referenced plus one (+1).	
	PREVIOUS e.g., PAR>EX T P<CR> Display the contents of the last PDP-10 location referenced minus one (-1).	
	THIS e.g., PAR>EX T T<CR> Display the contents of the last PDP-10 location referenced. THIS is the default.	
	AB e.g., PAR>EX AB<CR> Display the contents of the Address Break register.	
	AD e.g., PAR>EX AD<CR> Display the state of the ADder.	
	ADX e.g., PAR>EX ADX<CR> Display the state of the ADder Extended	
	AR e.g., PAR>EX AR<CR> Display the contents of the Arithmetic Register.	
	ARX e.g., PAR>EX ARX<CR> Display the contents of the Arithmetic Register eXtended.	
	BR e.g., PAR>EX BR<CR> Display the contents of the Buffer Register.	
	BRX e.g., PAR>EX BRX<CR> Display the contents of the Buffer Register eXtended.	
	CRADDR e.g., PAR>EX CRADDR<CR> Display the contents of the Cram ADDRess register.	
	CRLOC e.g., PAR>EX CRLOC<CR> Display the contents of the CRAM LOCation register.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	DRADDR e.g., PAR>EX DRADDR<CR> Display the contents of the DRAM ADDRess register.	
	DTE-20 e.g., PAR>EX DTE<CR> Display the contents of the three DIAG registers and the status register in the DTE20.	
	EBUS e.g., PAR>EX EBUS<CR> Display the contents of the EBus.	
	FE e.g., PAR>EX FE<CR> Display the contents of the Floating Exponent register.	
	FLAGS e.g., PAR>EX FLAGS<CR> Display the state of the flag bits (00-12) in the left half of the PC: OVF, CY0, CY1, FOV, BIS, USR, UIO, LIP, AFI, AT1, AT0, FUF and NOV.	
	FM e.g., PAR>EX FM<CR> Display the contents of the Fast Memory register.	
	KL e.g., PAR>EX KL<CR> Perform, in order, an EX PC, EX VMA, EX PI and EX FLAGS.	
	MQ e.g., PAR>EX MQ<CR> Display the contents of the Multiplier Quotient register.	
	PC e.g., PAR>EX PC<CR> Display the contents of Program Counter.	
	PI e.g., PAR>EX PI<CR> Display the state of the Priority Interrupt system.	
	REGISTERS e.g., PAR>EX REG<CR> Display the contents of the following registers: AD, ADX, AR, ARX, BR, BRX, EBUS, FM, MQ, and PC.	
	SBR e.g., PAR>EX SBR<CR> Display the contents of the Subroutine Return register.	
	SC e.g., PAR>EX SC<CR> Display the contents of the Shift Count register.	
	VMA e.g., PAR>EX VMA<CR> Display the contents of the Virtual Memory Address register.	
	VMAH e.g., PAR>EX VMAH<CR> Display the contents of the Virtual Memory Address Held register.	
FREAD	PAR>FR 110<CR> Display the result of a diagnostic function read using the function code specified (110). The function code must be in the range of 100 to 177.	
FWRITE	PAR>FW 77:252525777777<CR> Perform a diagnostic function write using the function code (77) and data (252525777777) specified. The function code must be in the range of 40 to 77.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
FXCT	PAR>FX 0<CR> Perform a diagnostic function execute using the function code specified (0). The function code must be in the range of 00 to 37.	4
HALT	PAR>H<CR> Halt the KL10. See ABORT and SHUTDOWN.	5
INITIALIZE	PAR>I<CR> Check the state of the KL10 clock, run flip-flop and opcode enable.	6
JUMP	PAR>J 30000<CR> Start the KL10 at the address specified (30000) and exit. The address is in the executive space and the processor mode is not affected. See START TEN.	
MCR	PAR>M BOOT<CR> Load and start the specified task file (BOOT.TSK). Same as RUN.	
QUIT	PAR>Q<CR> Exit from PARSER. Same as SET CONSOLE USER<CR> or ^Z.	
REPEAT n	PAR>REP 2;EX T N<CR> Cause the command(s) in the remainder of the line to be repeated n(2) times.	7
RESET	PAR>RES ALL<CR> The RESET command accepts the following arguments. Default: <CR>.	
	<CR> e.g., PAR>RES<CR> Cause a master reset of the KL10. The state of the clock enables and parity stops are not affected. This is the default.	
	ALL e.g., PAR>RES AL<CR> Perform a RES APR, RES DTE-20, RES PAG and RES PI command. The KL10 must be halted.	
	APR e.g., PAR>RES AP<CR> Execute a CONO APR,267760. The KL10 must be halted.	
	DTE-20 e.g., PAR>RES D<CR> Reset the DTE20.	8
	ERROR e.g., PAR>RES E<CR> Execute a CONO APR,27760 clearing the error flags in the Arithmetic Process Register (APR).	
	INITIALIZE e.g., PAR>RES IN<CR> Perform a KL10 master reset and return clock enables and parity stops to their default. The KL10 must be halted.	
	IO e.g., PAR>RES IO<CR> Execute a CONO APR,200000 which causes an I/O reset.	
	PAGE e.g., PAR>RES PAG<CR> Execute a CONO PAG,0 followed by a DATAO PAG,X (where the contents of X = 100). This will reset the KL10 paging box.	
	PI e.g., PAR>RES PI<CR> Execute a CONO PI,10000 which resets the Priority Interrupt system.	

Table 2   PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
RUN	PAR>RU PIP<CR> Load and run the specified task file (PIP.TSK). Same as MCR.	
SET	PAR>SET MEM TEN<CR> The SET command accepts the following arguments.	
	CLOCK e.g., PAR>SET CL N<CR> The SET CLOCK command accepts the following arguments.	
	CONTROL e.g., PAR>SET CL CON<CR> Enable the control logic clock.	
	CRAM e.g., PAR>SET CL CR<CR> Enable the CRAM clock.	
	DATA-PATH e.g., PAR>SET CL D<CR> Enable the data path clock.	
	EXTERNAL e.g., PAR>SET CL E<CR> Set (select) the KL10 external clock source. PARSER will request confirmation.	9
	HALF e.g., PAR>SET CL H<CR> Set the KL10 clock rate to one half of the standard (divide by 2).	
	INTERNAL e.g., PAR>SET CL I<CR> Set (select) the KL10 internal clock source.	
	MARGIN e.g., PAR>SET CL M<CR> Set (select) KL10 clock margins.	
	NORMAL e.g., PAR>SET CL N<CR> Set the KL10 clock rate to the standard (internal source, full rate with CRAM, data-path and control logic clocks enabled).	
	QUARTER e.g., PAR>SET CL Q<CR> Set the KL10 clock rate to one quarter of the standard (divide by 4).	
	SLOW e.g., PAR>SET CL S<CR> Set the KL10 clock rate to one eighth of the standard (divide by 8).	
	CONSOLE e.g., PAR>SET CON M<CR> The SET CONSOLE command accepts the following arguments.	
	MAINTENANCE e.g., PAR>SET CON M<CR> Set the console to maintenance mode. The command set is unrestricted. Refer to Table 2.	
	OPERATOR e.g., PAR>SET CON O<CR> Set the console to operator mode. The command set is restricted to those listed in Table 3.	
	PROGRAMMER e.g., PAR>SET CON P<CR> Set the console to programmer mode. The command set is restricted to those listed in Table 4.	
	USER e.g., PAR>SET CON U<CR> Exit PARSER. Leave the CTY connected to the program running in the KL10.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	DATE e.g., PAR>SET D<CR> Set the date and time to be used by the front-end executive, RSX-20F. This command is illegal if RSX-20F already has a valid date from a previous SET DATE command or a reload of the KL10.	
	FS-STOP e.g., PAR>SET F<CR> Enable the Field Service Clock Error Stop feature in the KL10. This requires backplane jumper wires to be meaningful. Same as SET PARITY-STOP FS-STOP.	
	INCREMENT e.g., PAR>SET I 10<CR> Set the increment and decrement value for KL10 deposit and examine commands to the value specified (10).	
	KLINIK e.g., PAR>SET K<CR> Set the KLINIK link for remote console operation.	15
	MEMORY e.g., PAR>SET M T<CR> The SET MEMORY command accepts the following arguments. ELEVEN e.g., PAR>SET M E<CR> Set the PDP-11 as the default memory for deposits and examines. TEN e.g., PAR>SET M T<CR> Set the KL10 as the default memory for deposits and examines.	
	NOT e.g., PAR>SET NO RELOAD<CR> Used with SET to negate the SET function. It is equivalent to CLEAR.	
	OFFSET e.g., PAR>SET O 101204<CR> Set the PDP-11 relocation counter to the value specified (101204). The relocation counter is initially set to the address of the PARSER root overlay.	
	PARITY-STOP e.g., PAR>SET P ALL<CR> The SET PARITY-STOP command accepts the following arguments. ALL e.g., PAR>SET P ALL<CR> Set the parity stop enable to on and enable the following parity stop features. AR, CRAM, DRAM, FM and FS-STOP. AR e.g., PAR>SET P AR<CR> Add stop on AR and ARX parity error to the parity stop features. CRAM e.g., PAR>SET P C<CR> Add stop on CRAM parity error to the parity stop conditions. DRAM e.g., PAR>SET P D<CR> Add stop on DRAM parity error to the parity stop conditions. ENABLE e.g., PAR>SET P E<CR> Enable (turn on) the selected PARITY-STOP features. FM e.g., PAR>SET P FM<CR> Add stop on a fast memory (FM) parity error to the parity stop conditions. FS-STOP e.g., PAR>SET P FS<CR> Enable the Field Service Clock Error Stop feature in the KL10. This requires backplane jumper wires. Same as SET FS-STOP.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	RELOAD e.g., PAR>SET REL<CR> Enable the automatic reload of the KL10 by the PDP-11 front end. This is the default. See CLEAR RELOAD.	
	REPEAT e.g., PAR>SET REP 5<CR> Set the repeat counter to the decimal value specified. All subsequent command lines will be repeated that number of times. The value will also be used as a multiplier by the REPEAT command.	7
	RETRY e.g., PAR>SET RET<CR> Set the PARSER RETRY flag. See CLEAR RETRY.	17
	TRACKS e.g., PAR>SET T<CR> Display all FR, FW, FX, Examine, Deposit, and DTE-20 operations.	10
SHUTDOWN	PAR>SH<CR> Gracefully shut down the TOPS-10 or TOPS-20 operating system. This is done by depositing a minus 1 in location 30. Timesharing ceases.	11
START	PAR>ST M 0<CR> or PAR>ST T 2000<CR> The START command accepts the following arguments. START with no arguments or an argument of 0 is illegal. If neither TEN nor MICROCODE is specified, TEN is assumed.	
	MICROCODE e.g., PAR>ST M 0<CR> Start the microcode at the address specified (0).	12
	TEN e.g., PAR>ST T 3000<CR> Start the KL10 at the address specified (3000). See CONTINUE and JUMP.	13
WHAT	PAR>W CL<CR> The WHAT command accepts the following arguments.	
	CLOCK e.g., PAR>W CL<CR> Display the current clock state. See SET CLOCK.	
	CONSOLE e.g., PAR>W CON<CR> Display the current console mode. See SET CONSOLE.	
	DATE e.g., PAR>W D<CR> Display the state of the validity flag and the current date and time held by RSX-20F.	
	INCREMENT e.g., PAR>W I<CR> Display the current increment/decrement value. See SET INCREMENT.	
	KLINIK e.g., PAR>W K<CR> Display the current status of the KLINIK link. See SET KLINIK.	15
	MEMORY e.g., PAR>W M<CR> Display the current default memory. See SET MEMORY.	
	OFFSET e.g., PAR>W O<CR> Display the current value of the PDP-11 relocation counter. See SET OFFSET.	

Table 2 PARSER Maintenance Mode Command Summary (Cont)

Command	Description	Cross Ref.
	PARITY-STOP e.g., PAR>W P<CR> Display the current state of the parity stop feature. See SET PARITY-STOP.	
	RELOAD e.g., PAR>W REL Display the current state of the KL10 automatic reload feature (ON or OFF). See SET RELOAD.	
	REPEAT e.g., PAR>W REP<CR> Display the current value of the repeat counter. See SET REPEAT.	
	RETRY e.g., PAR>W RET<CR> Display the state of the PARSER RETRY flag. See SET RETRY.	17
	TRACKS e.g., PAR>W T<CR> Display the current state of the trace enable feature (ON or OFF). See SET TRACKS.	10
	VERSION e.g., PAR>W V<CR> Display the current version of PARSER and RSX-20F.	
XCT	PAR>X 254200000000<CR> Execute the argument (245200000000) as a PDP-10 instruction. The KL10 must be in executive mode.	14
ZERO	PAR>Z 200>277<CR> Zero PDP-10 physical memory from first argument (200) through second argument (277). Note: depending on the amount of memory this may take a while.	

Table 3 PARSER Operator Mode Command Summary

Command	Description
ABORT	PAR>A<CR>
CLEAR	PAR>CL C<CR> or PAR>CL R<CR> etc. The CLEAR command accepts the following arguments. CONSOLE KLINIK NOT INCREMENT MEMORY REPEAT
DISCONNECT	PAR>DI<CR>
EXAMINE	KL e.g., PAR>EX KL<CR>
	PC e.g., PAR> EX PC<CR>
	ELEVEN e.g., PAR>EX EL adr<CR>
	DECREMENT e.g., PAR>EX EL D<CR>
	INCREMENT e.g., PAR>EX EL I<CR>
	NEXT e.g., PAR> EX EL N<CR>
	PREVIOUS e.g., PAR> EX EL P<CR>
	THIS e.g., PAR> EX EL T<CR>
	TEN e.g., PHR>EX T adr<CR>
	DECREMENT e.g., PAR>EX T D<CR>
	INCREMENT e.g., PAR>EX T I<CR>
	NEXT e.g., PAR>EX T N<CR>
	PREVIOUS e.g., PAR>EX T P<CR>
	THIS e.g., PAR>EX T T<CR>

Table 3 PARSER Operator Mode Command Summary (Cont)

Command	Description
JUMP	PAR>J 30000<CR>
MCR	PAR>MCR BOOT<CR>
QUIT	PAR>Q<CR>
REPEAT	PAR>REP 2:EX T N<CR>
RUN	RU PIP<CR>
SET	CONSOLE e.g., PAR>SET CON M<CR>
	The SET console command accepts the following four arguments: USER, OPERATOR, PROGRAMMER and MAINTENANCE.
	INCREMENT e.g., PAR>SET I 10<CR>
	KLINIK e.g., PAR>SET K<CR>
	MEMORY e.g., PAR>SET M E<CR> or PAR>SET M T<CR>

Table 4 PARSER Programmer Mode Command Summary

Command	Description
ABORT	PAR>A<CR>
CLEAR	PAR>CL C<CR> or PAR>CL T<CR> etc.
	The CLEAR command accepts the following arguments.
	CONSOLE MEMORY REPEAT DATE NOT RETRY INCREMENT OFFSET TRACKS KLINIK RELOAD
CONTINUE	PAR>CO<CR>
DEPOSIT	AR e.g., PAR>DE A:data<CR>
	ELEVEN e.g., PAR>DE E adr: data<CR>
	DECREMENT e.g., PAR>DE E D:data<CR>
	INCREMENT e.g., PAR>DE E I:data<CR>
	NEXT e.g., PAR>DE E N:data<CR>
	PREVIOUS e.g., PAR>DE E P:data<CR>
	THIS e.g., PAR>DE E T:data<CR>
	TEN e.g., PAR>DET adr:data<CR>
	DECREMENT e.g., PAR>DE T D:data<CR>
	INCREMENT e.g., PAR>DE T I:data<CR>
	NEXT e.g., PAR>DE T N:data<CR>
	PREVIOUS e.g., PAR>DE T P:data<CR>
	THIS e.g., PAR>DE T T:data<CR>
DISCONNECT	PAR>DI<CR>

Table 4 PARSER Programmer Mode Command Summary (Cont)

Command	Description
EXAMINE	PAR>EX AB<CR> or PAR>EX PC<CR> etc.
	The EXAMINE command accepts any of the following arguments.
	AB CRLOC MQ AD DRADDR PC ADX DTE-20 PI AR EBUS REGISTERS ARX FE SBR BR FLAGS SC BRX FM VMA CRADDR KL VMAH
	ELEVEN e.g., PAR>EX EL adr<CR>
	DECREMENT e.g., PAR>EX EL D<CR>
	INCREMENT e.g., PAR>EX EL I<CR>
	NEXT e.g., PAR>EX EL N<CR>
	PREVIOUS e.g., PAR>EX EL P<CR>
	THIS e.g., PAR>EX EL T<CR>
	TEN e.g., PAR>EX T adr<CR>
	DECREMENT e.g., PAR>EX T D<CR>
	INCREMENT e.g., PAR>EX T I<CR>
	NEXT e.g., PAR>EX T N<CR>
	PREVIOUS e.g., PAR>EX T P<CR>
	THIS e.g., PAR>EX T T<CR>
HALT	PAR>H<CR>
INITIALIZE	PAR>I<CR>
JUMP	PAR>J 30000<CR>
MCR	PAR>MCR BOOT<CR>
QUIT	PAR>Q<CR>
REPEAT	PAR>REP 2;EX T N<CR>
RESET	PAR>RES ALL<CR> or PAR>PAG<CR> etc
	The RESET command accepts the following arguments.
	ALL ERROR PAG APR INITIALIZE PI DTE-20 I/O
RUN	PAR>RU PIP<CR>
SET	CONSOLE e.g., PAR>SET CON M<CR>
	The SET CONSOLE command accepts four arguments; USER, OPERATOR, PROGRAMMER and MAINTENANCE.
	DATE e.g., PAR>SET D<CR>
	INCREMENT e.g., PAR>SET I 10<CR>
	KLINIK e.g., PAR>SET K<CR>
	MEMORY e.g., PAR>SET M E<CR> or PAR>SET M T<CR>
	The SET MEMORY command accepts two arguments: ELEVEN and TEN.
	NOT e.g., PAR>SET NO arg<CR>
	OFFSET e.g., PAR>SET O 101204<CR>

Table 4 PARSER Programmer Mode Command Summary (Cont)

Command	Description
	RELOAD e.g., PAR>SET REL<CR>
	REPEAT e.g., PAR>SET REP 5<CR>
	RETRY e.g., PAR>SET RET<CR>
	TRACKS e.g., PAR>SET T<CR>
SHUTDOWN	PAR>SH<CR>
START	PAR>ST M<CR> or PAR>ST T 3000<CR>
	The START command accepts two arguments: MICROCODE and TEN.
WHAT	PAR>W CL<CR> or PAR>W V<CR> etc.
	The WHAT command accepts the following arguments.
	CLOCK, MEMORY, RETRY CONSOLE, OFFSET, TRACKS DATE, PARITY-STOP, VERSION INCREMENT, RELOAD KLINIK, REPEAT
XCT	PAR>X 254200000000<CR>
ZERO	PARZ 200>277<CR>

COMMAND DESCRIPTION
This section describes in detail the commands listed in Table 2.

1 A<CR> - The ABORT command stops the KL10 by trying to force it into the HALT loop. If this fails after a reasonable number of EBox clock ticks, the command tries to START MICROCODE at CRAM address 0, which implies a master reset of the KL10 processor.

NOTE
This is the best way to get the KL10 into a known state when the previous state left it hung.

2 CO<CR> - The CONTINUE command takes the KL10 out of the HALT loop, causing it to execute the instruction pointed to by the PC. If single instruction mode was not set, the KL10 should continue running. If single instruction mode was set via the FXCT 12 function, the instruction is executed, and the KL10 is returned to the HALT loop.

3 FLAGS<CR> - The PC flag mnemonics displayed are defined as follows.

AFI - Address Failure Inhibit (bit 08)
AT0 - Trap 1 (bit 10)
AT1 - Trap 2 (bit 09)
BIS - First Part Done (bit 04)
CY0 - Carry 0 (bit 01)
CY1 - Carry 1 (bit 02)
FOV - Floating Overflow (bit 03)
FUF - Floating Underflow (bit 11)
LIP - Public (bit 07)
NDV - No Divide (bit 12)
OVF - Overflow/Previous Context Public (bit 00)
UIO - User In-Out/Previous Context User (bit 00)
USR - User (bit 05)

4 FX<CR> - The FXCT command accepts a number as a function write code, performs the function write, and displays the result. Useful values are 0 (stops the KL10 clock), and 1 (starts the KL10 clock). Random use of FXCT can cause false CRAM parity errors. (Use the HALT or ABORT commands first.)

5 H<CR> - The HALT command tries to put the KL10 into the HALT loop by clearing RUN, and waiting. If the KL10 is unable to go into the HALT loop, the HALT command tries to force it in by using BURST mode. If this does not work, an error message is displayed.

6 I<CR> - The INITIALIZE command (re)initializes PARSER, and checks the state of the KL10, sets up the KL10 state flag word with default values and restarts the KL10 based on those values. The following KL10 conditions are checked: clock running, run flip-flop set, and opcode enabled. INITIALIZE also checks to see if this PDP-11 is running on a privileged DTE20.

7 REP 2:EX T P<CR> - The REPEAT n command causes the command(s) in the remainder of the line to be repeated n (2) times if the SET REPEAT value is set to 1. See SET REPEAT. If the SET REPEAT value is greater than 1 then it is multiplied by the REPEAT n value and the commands are repeated that many times.

8 RES D<CR> - The RESET DTE-20 command resets the DTE20 by depositing a 1 in bit 6 of DIAG WORD 2 in the DTE20. Then bit 0 in DIAG WORD 1 of the DTE20 is set to 1 indicating word mode transfers.

9 SET CL E<CR> - The SET CLOCK EXTERNAL command selects the external clock source for the KL10. If no external clock source is connected, the KL10 is stuck and can only be reset by powering the system down and then up again.

10 SET T<CR> - The SET TRACKS command causes changes in the internal state of the KL10 to be displayed after each clock tick. This is done via diagnostic reads and is primarily used for debugging hardware or front-end software. This will result in a lot of wasted paper if you are not careful.

11 SH<CR> - The SHUTDOWN command deposits a -1 (minus one) into KL10 executive virtual location 30 (octal). It is used to gracefully bring down the KL10 timesharing systems. It will cause PARSER to exit if the deposit was successful, which will cause the console terminal to be connected to either EDDT (if loaded), or to the dead KL10. If EDDT is not loaded, the KL10 will execute a HALT instruction (TOPS-20 only) as soon as the clock interrupt is serviced.

12 ST M 0<CR> - The START MICROCODE command performs a KL10 master reset and starts the microcode at the microcode address specified. Starting the MICROCODE at addresses other than 0 is probably not helpful for most users.

13 ST T 30000<CR> - The START TEN command starts the KL10 at the address requested using an algorithm determined by the version of the microcode. It puts the KL10 into the HALT loop, loads the address onto the AR, and does a function CONTINUE, causing the KL10 to start at the address requested in EXEC KERNAL mode. To start the KL10 without losing the old processor mode, use the JUMP command, which will accept an address, EXECUTE a JRST (opcode 254) to that address (in EXEC Virtual Space), and continue in whatever mode the processor was in.

14 X 254200000000<CR> - The XCT command takes a 36-bit octal argument and executes it as a KL10 instruction.

NOTE

Executing an instruction with an opcode of zero may cause random results because the microcode uses op-code zero coming out of the HALT loop for START and CONTINUE.

15 SET KLINIK<CR> - The RSX-20F KLINIK link is enabled by issuing a SET KLINIK command to PARSER from the local console (CTY). PARSER will then request and validate the following parameters.

PARSER will request the KLINIK mode desired with the following prompt.

KLINIK MODE:

The acceptable response to this prompt is either USER or REMOTE.

USER indicates that the KLINIK link is to be used as a timesharing terminal line (only). See SET CONSOLE USER.

REMOTE indicates that the KLINIK link is to be used as a remote console line in either Maintenance, Operator or Programmer mode. See SET CONSOLE.

There is no default response to this prompt. If any other response is supplied, the command will abort and the local operator will receive one of the following error messages:

PAR [SET] NSK NO SUCH KEYWORD "XXX"
PAR [SET] ILC ILLEGAL CHARACTER "C"

where "XXX" and "C" are the offending keyword and character, respectively.

Next PARSER will request the KLINIK ACCESS WINDOW parameters by printing the following prompts and accepting responses in sequence.

ACCESS WINDOW OPEN DATE:
ACCESS WINDOW OPEN TIME:
ACCESS WINDOW CLOSE DATE:
ACCESS WINDOW CLOSE TIME:

The possible date formats are as follows.

DD-MMM-YY
DD-MMM-YYYY
DD MMM YY
DD MMM YYYY

DD is the decimal day, MMM is the alphabetic representation of the month, and YY or YYYY is the decimal year in which the KLINIK WINDOW is to open or close. The default response to a date prompt is a <carriage return>. This will set the Window Open Date to TODAY, and the Window Close Date to TODAY + 1. TODAY is the current date obtained from RSX-20F. See WHAT DATE.

The day specified must be within the range of 1-31. Date for months having less than 31 days will be validated. This includes a special check for February in a leap year. The month MMM is composed of the first three letters of the month to be entered. The year may be specified as either a Gregorian year, 19XX, or as a year relative to 1900, (00 through 99) where the first two digits are assumed to be the first two digits of the current century. Failure to adhere to this syntax will cause the command to abort, and one of the following error messages to be printed.

PAR [SET] DOR DAY OUT OF RANGE - If the day specified does not exist in the month specified.

PAR [SET] NSK NO SUCH KEYWORD "XXX" - If the keyword specified for the month cannot be matched.

PAR [SET] AMB AMBIGUOUS KEYWORD "XXX" - If that keyword is ambiguous. "XXX" is the offending keyword.

PAR [SET] YOR YEAR OUT OF RANGE - If the year has been improperly specified.

PAR [SET] DBT DATE BEFORE TODAY - If the entire window open or close date is prior to TODAY.

The Window Open Time and Window Close Time may be specified in either of the following formats.

HHMM
HH:MM

HHMM is a representation of the hour and minute. In both formats, HH is the hour and must be within the range of 00 to 23, and MM is the minute and must be within the range of 00 to 60. The default response is a <carriage return>. This will set the Window Open Time and the Window Close Time to NOW. NOW is the current time of day obtained from RSX-20F. See WHAT DATE.

Specifying a time which does not conform to this syntax will cause the command to abort and the following error message to be printed.

PAR [SET] TOR TIME OUT OF RANGE

Finally, when the complete specifications for both the Window Open and Window Close times and dates have been specified, the Window Open time and date will be checked to ensure that it does precede the Window Close time and date. If this is not the case, the command will abort and the following error message will be printed.

PAR [SET] KWE KLINIK WINDOW ERROR

If the KLINIK mode specified was USER, the dialogue will terminate at this point, as all necessary parameters have been input. If the specified KLINIK mode was REMOTE, two more parameters will be solicited from the operator. PARSER will first request a password with the following prompt.

PASSWORD:

The local operator must communicate this password to the remote KLINIK user in order that he be allowed access to the KLINIK link.

The password must be at least one and not more than six numeric or uppercase alphabetic characters, with no imbedded or trailing blanks. There are no default responses. The operator's response to this prompt will be echoed on the local console (CTY).

Failure to provide a password in this form will cause the command to abort and one of the following messages to be printed.

PAR [SET] NPI NULL PASSWORD ILLEGAL - If no password was specified.

PAR [SET] PTL PASSWORD TOO LONG - If more than six characters were typed.

PAR [SET] IPC ILLEGAL PASSWORD CHARACTER "C" - If a nonalphanumeric character was typed as a password character. "C" is the offending character.

PARSER will next request that the operator specify the highest PARSER console mode to be allowed while the KLINIK link is active with the following prompt.

HIGHEST CONSOLE MODE:

The acceptable responses to this prompt areas follows (See SET CONSOLE).

MAINTENANCE
OPERATOR
PROGRAMMER

While the KLINIK link is active, PARSER will not allow the remote or the local console to raise the command PARSER console mode, to a level higher than that specified in response to this prompt. There is no default response to this prompt.

Failure to provide the proper response to this prompt will cause the command to abort and the following error message to be printed:

PAR [SET] NSK NO SUCH KEYWORD "XXX"

where "XXX" is the offending keyword.

If all parameters have been properly input and validated, PARSER will return to command level after displaying the KLINIK enable parameters in the following format.

KLINIK [<ACTIVE> <INACTIVE> <DISABLED>]
ACCESS WINDOW OPEN: DD-MMM-YY HH:MM
ACCESS WINDOW CLOSED: DD-MM-YYY HH:MM
KLINIK MODE: [<REMOTE> <USER>]

ACTIVE indicates that the KLINIK user is connected to the RSX-20F KLINIK link.

INACTIVE indicates that the KLINIK parameters have been set, but access has not yet been allowed (i.e., the WINDOW is not open yet).

DISABLED indicates that no KLINIK parameters have been set.

If the KLINIK mode is REMOTE, one additional line will be displayed describing the highest PARSER console mode to be allowed.

CONSOLE MODE LIMIT: [<MAINTENANCE> <OPERATOR> <PROGRAMMER>]

Upon receipt of these parameters RSX-20F will log the SET KLINIK command and the parameters that were accepted. Further, RSX-20F will pass these parameters to the KL10 operating system (TOPS-20 or TOPS-20), to facilitate KLINIK recovery from a PDP-11 reboot.

16 CLEAR KLINIK<CR> - The RSX-20F KLINIK link is disabled via the CLEAR KLINIK command. This command does not accept arguments, it simply clears the KLINIK WINDOW. If the KLINIK link is active, the CLEAR KLINIK command will cause the following message to be printed on both the local and the remote consoles.

KLD KLINIK ACCESS TERMINATED BY OPERATOR

The current KLINIK enable parameters will be reset and passed to the KL10 operating system (TOPS-10 or TOPS-20). The KLINIK ACCESS WINDOW will close and RSX-20F will log the KLINIK mode termination on the CTY. The modem will not be hung up; however, all input from and output to the remote console will be ignored and all subsequent calls made to the KLINIK LINK will be acknowledged and rejected until such time as a new KLINIK WINDOW is set by the local operator. The rejection message will be in the following format.

KLR--KLINIK RING KLINIK-WINDOW CLOSED

This rejection message will appear on both the local and remote consoles.

17 CL RET<CR> - When the RETRY flag is set, the occurrence of a KEEP-ALIVE-CEASED error will result in the execution of the instruction in location 71. The instruction typically branches to a routine that will cause the KL10 operating system (TOPS-10 or TOPS-20) to dump memory and request a reload. If this can not be accomplished before the end of the keep-alive period (5 seconds), then RSX-20F assumes that the KL10 is incapacitated. In this case KLERR is called to take a KL10 hardware snapshot and then reload the KL10.

If the RETRY flag is clear (CLEAR RETRY command) every occurrence of a KEEP-ALIVE-CEASED error will result in a KLERR snapshot and reload of the KL10.

RSX-20F STOP CODES AND I/O ERROR CODES
This appendix contains two lists of error codes. The first list contains RSX-20F stop codes. Associated with each code is the name of the module that issued the stop code, a short explanation of the error, and a possible cause of the error. The second is a list of I/O error codes that are produced by the device handlers and file control primitives. These error codes have associated messages that are listed along with them; however, due to the many different situations in which these errors can arise, no attempt is made to describe recovery algorithms for these errors.

Code	Module	Meaning
BO3	SCOMM	BUFFER OVERFLOW 3 The PDP-11 was not able to obtain the buffer space necessary for data it wanted to send to the KL. Possible Cause: Buffer pool space became exhausted or highly fragmented. R1 contains the node (buffer) size requested. FREPL points to the list of free nodes. .FREPL+2 contains the number of free bytes in the pool. Nodes are linked together in the forward direction through the first word of the node. The second word of each node contains the node size.
BO5	TTYDRR	BUFFER OVERFLOW 5 The Front-End does not have the buffer space to send an XON or an XOFF to a line.
CBR	PF	CROBAR ERROR DTE-20 power did not return after a power-fail restart. RSX-20F allows it 30 seconds to reappear. Possible Cause: Malfunctioning hardware in the KL.
DTB	QPRDTE	TO-11 DTE TRANSFER FAILURE A TO-11-done interrupt has occurred, but the TO-11 address in the DTE TO11AD register (register 22) did not have the expected value. Since TO11AD is incremented for each byte transferred, it should point to the first word following the buffer into which the TO-11 data was written. Possible Cause: The PDP-11 received the wrong byte count or, more likely, the DTE has a hardware malfunction. TO11BC contains the negative count of data that was actually transferred. TO11AS contains address of data node. R1 contains expected termination address and CR$DTB-2 contains the actual termination address for transfer.
DTD	COMTRP	UNIBUS TIMEOUT Reference to the DTE-20 caused a UNIBUS timeout. Possible Cause: Malfunction of the hardware in the KL.
DTF	QPRDTE	TO-10 DTE TRANSFER FAILURE A TO-10-done interrupt has occurred but the TO-10 address in the DTE TO10AD register (register 20) did not have the expected value. Since TO10AD gets incremented for each byte transferred, it should point to the first word following the packet that was sent to the KL. Possible Cause: The PDP-11 gave the KL the wrong byte count or, more likely, the DTE has a hardware malfunction. TO10SZ contains the size of the transfer and TO10AS the start address. The expected termination address is in R4.

RSX-20F STOP CODES AND I/O ERROR CODES (Cont)

Code	Module	Meaning
ETE	QPRDTE	TO-11 TRANSFER ERROR A DTE interrupt occurred with the TO11ER bit set in the DTE status register (register 34). Possible Cause: Hardware malfunction along the data path between the KL and PDP-11 (MBOX, EBOX, EBUS, DTE-20, through to 11-memory).
FTA	LC	FILES-11 TASK ABORTED A task occupying F11TPD partition has aborted and the task termination notification task (TKTN) cannot be started since it too runs in the F11TPD partition. Possible Cause: .TKTN may have aborted. R5 and .CRTSK point to the Active Task List (ATL) node of the aborted task.
IAS	SCH	UNKNOWN SIGNIFICANT EVENT An unused bit in .SERFG has been set. Possible Cause: PDP-11 hardware malfunction or corrupted software in PDP-11. .SERFG has the bit set.
ILF	QPRDTE	ILLEGAL PROTOCOL FUNCTION The function code in a TO-11 protocol header specified a function that is outside the legal range or that is currently unimplemented. Possible Cause: KL software is corrupted or hardware malfunction along data path between KL and PDP-11. R1 contains the function code times two. R4 contains the address of the protocol header.
ILQ	QPRDTE	ILLEGAL QUEUE COUNT The KL and the PDP-11 disagree on the number of direct transfers that have thus far taken place from the KL to the PDP-11. You should take into account that indirect headers are sent across the DTE-20 as direct packets. Possible Cause: The PDP-11 is missing TO-11 doorbell interrupts, or the software of either the KL or the PDP-11 is corrupted. STATI+0 to STATI+2 contain the KL's TO-11 status word as read by RSX-20F at the last examine. STATI+4 is the count the KL expects, and TO10QC is the count the PDP-11 expects.
LRF	SCH	LOAD REQUEST FAILURE An attempt to load a nonresident monitor routine into the F11TPD partition failed. Possible Cause: The Files-11 system is incomplete or damaged.
MPE	LC	MEMORY PARITY ERROR A memory parity error has occurred in the PDP-11 (trap to location 114). The memory status registers are stored starting at location PARSAVE. (Refer to the PDP-11 Processor Handbook for details.)

RSX-20F STOP CODES AND I/O ERROR CODES (Cont)

Code	Module	Meaning
PT1	QPRDTE	PROTOCOL BROKEN An illegal protocol device number was specified in TO-11 request. The number was found to be greater than the maximum allowed device number .DQPSZ (currently 10). Possible Cause: KL software is corrupted or hardware malfunction along the data path between the KL and PDP-11. The device number from the protocol header is in TO11DV.
PT2	QPRDTE	PROTOCOL ERROR 2 An illegal protocol function was specified in a TO-11 request. The function was found to be greater than the allowed maximum BC.FNM (currently 34). Possible Cause: Same at PT1 above. The function code from the protocol header is in TO11FN.
PT3	QPRDTE	PROTOCOL ERROR 3 The PDP-11 has received a doorbell interrupt from the KL. The indirect bit in the KL's TO-11 status word indicates that an indirect transfer is to be initiated. The function code, however, sent in the last protocol header, does not indicate that an indirect request is in progress (the most significant bit of the function code was not set). Possible Cause: Same as PT1 above. TO11FN contains the function code and STATI contains the TO-11 protocol status word.
PT4	QPRDTE	PROTOCOL ERROR 4 The KL wants to send a packet to the PDP-11, but the packet size is greater than the maximum allowed size of 100. Possible Cause: Same as PT1 above. The size in in EQSZ.
RED	RED	REDIRECT ERROR A fatal error has occurred during an MCR REDIRECT command. The file control service is corrupted. Call your Software Support Specialist.
RES	LC	RESERVED INSTRUCTION TRAP This is the PDP-11 trap to location 10. An attempt was made to execute an illegal or reserved instruction. Refer to the PDP-11 Processor Handbook for further details. Possible Cause: PDP-11 software is corrupted,or a PDP-11 hardware malfunction occurred.
TBT	LC	T-BIT TRAP This PDP-11 trap to location 14 occurs when the BPT instruction (not used by RSX-20F) is executed or when the T-bit is set. (See the PDP-11 Processor Handbook for further details.)

RSX-20F STOP CODES AND I/O ERROR CODES (Cont)

Code	Module	Meaning
		Possible Cause: Corrupted PDP-11 software or PDP-11 hardware malfunction.
TET	QPRDTE	TO-10 TRANSFER ERROR A DTE-20 interrupt has occurred with either TO10ER (TO-10 error) or MPE11 (PDP-11 parity error) bit set in the DTE-20 status register (register 34). Possible Cause: DTE-20 hardware error, PDP-11 memory parity error, or hardware malfunction along the data path between the PDP-11 and KL.
T04	LC	TRAP AT LOCATION 4 The PDP-11 traps to location 4 when it makes a word reference to an odd address or when a bus timeout occurs. (See the PDP-11 Processor Handbook for further details.) Possible Cause: PDP-11 software is corrupted, or a PDP-11 peripheral device is malfunctioning or has gone away.
UIE	QPRDTE	UNIMPLEMENTED PROTOCOL FUNCTION The KL uses bits 0-2 of its TO-11 status word in the communications region to inform the front end of any disaster occurring in the KL. These bits are read by the front end on receipt of a TO-11 doorbell. The currently implemented functions are KL-RELOAD REQUEST and KL POWER FAIL. Any other bits that are set cause this halt. Possible Cause: Corrupted KL software, a KL hardware malfunction or any hardware malfunction along the data path between KL and PDP-11 could be the cause of this error.

The following is a list of possible I/O error codes. Since these codes are returned by the device handlers and file control primitives in RSX-20F, they are global in the sense that they can come from any utility in the system. That is, a code of -33 means the same thing when it comes from PIP that it means when it comes from SAV. Because of the global nature of the error codes, it is not possible to describe the exact problem; the situation is different with different utilities. Therefore, the following list does not attempt to explain the error code other than to list the message associated with it.

Note that there are two messages associated with the code -2. This is legitimate; a message code of -2 is produced in two types of situations.

Code	Message
-1	Bad parameters
-2	Invalid function code
-2	EBOX stopped
-3	Device not ready
-4	Parity error on device
-5	Hardware option not present
-6	Illegal user buffer
-7	Device not attached
-8	Device already attached
-9	Device not attachable

Code	Message
-10	End of file detected
-11	End of volume detected
-12	Write attempted to locked unit
-13	Data overrun
-14	Send/receive failure
-15	Request terminated
-16	Privilege violation
-17	Sharable resource in use
-18	Illegal overlay request
-19	Odd byte count or virtual address
-20	Logical block number too large
-21	Invalid UDC module
-22	UDC connect error
-23	Caller's nodes exhausted
-24	Device full
-25	Index file full
-26	No such file
-27	Locked from write access
-28	File header full
-29	Accessed for write
-30	File header checksum failure
-31	Attribute control list format error
-32	File processor device read error
-33	File processor device write error
-34	File already accessed on LUN
-35	File ID, file number check
-36	File ID, sequence number check
-37	No file accessed on LUN
-38	File was not properly closed
-39	Open - no buffer space available for file
-40	Illegal record size
-41	File exceeds space allocated, no blocks
-42	Illegal operation on file descriptor block
-43	Bad record type
-44	Illegal record access bits set
-45	Illegal record attributes bits set
-46	Illegal record number - too large
-47	Multiple block read/write - not implemented
-48	Rename - two different devices
-49	Rename - new file name already in use
-50	Bad directory file
-51	Cannot rename old file system
-52	Bad directory syntax
-53	File already open
-54	Bad file name
-55	Bad device name
-56	Bad block or device
-57	Enter duplicate entry in directory
-58	Not enough stack space (FCS or FCP)
-59	Fatal hardware error on device
-60	File ID was not specified
-61	Illegal sequential operation
-62	End of tape detected
-63	Bad version number
-64	Bad file header
-65	Device off-line
-66	File expiration date not reached
-67	Bad tape format
-68	Not ANSI "D" format byte count

Table of Contents

GENERAL INFORMATION

PIP (Peripheral Interchange Program) is a utility program which is used to transfer files between standard peripheral devices. PIP can also perform editing and magtape control functions during file transfers.

R PIP <CR>	Monitor commmand to load and start PIP
*	Prompt - indicates PIP is ready to accept commands
↑C	Exit PIP - return to monitor command mode

Notes

1. This module is a summary of PIP intended for use by field engineers. Refer to the Software Notebooks for a complete description.
2. Wild characters, the asterisk (*) and question mark (?) may be used in filename and extension construction.
3. Octal constants may be used in filenames and extensions. The octal constant must be preceded by a pound sign (#) and delimited by a nonoctal digit or a character.
4. Including the "/X" switch in a command string will cause PIP to transfer each file separately (file by file) to the destination device.
5. Excluding the "/X" switch from the command string will cause PIP to combine (concatenate) the specified source files into one large file on the destination device.

COMMAND CONVENTIONS AND SWITCHES

PIP command conventions and switches are described in the following tables.

Table 1 PIP Command Conventions
Table 2 PIP Command String Delimiters
Table 3 PIP Acceptable Device Mnemonics
Table 4 File Protection Codes
Table 5 UFD and SFD Protection Codes
Table 6 PIP Control Switch Summary
Table 7 PIP Magtape Switch Summary

PIP Command String Format

A PIP command string consists of two fields separated by an equal sign (=) and terminated by a carriage return <CR>.

A PIP command string which is used to transfer files between I/O devices has the following format:

DESTINATION = SOURCE <CR>

A PIP command string which does not transfer files (i.e., move magtape) has the following format:

DESTINATION = <CR>

MTA3:(MU)=<CR>

The equal sign delimiter and a terminator are still required in commands formatted in this manner despite the fact that only the DESTINATION portion of the command is used.

The DESTINATION portion of a PIP command describes the device and file(s) which is to receive the transferred data. This portion of a command consists of one file specification.

The SOURCE side of the command describes the device from which the transferred data is to be taken. This portion of a command may contain one or more file specifications.

PIP command strings may be of any length; both upper and lower case characters may be used. PIP commands are normally terminated and the requested operation initiated by a carriage return. However, an ALTMODE, ESC, line feed, vertical TAB, or form feed can also be used as a command terminator.

Table 1 PIP Command Conventions

Convention	Description
dev:	Either a physical or a logical device name. Refer to Table 3.
[directory]	The identifier of a specific directory (i.e., UFD or MFD) within the system. This identifier may consist of a project, programmer number pair and Sub File Directory (SFD) names.
.ext	A 1- to 3-character alphanumeric extension assigned to the name of a file either by the user or by the system.
file	A 1- to 6-character alphanumeric identification which is either to be assigned to a new file (when on the destination side of the command) or which identifies an existing file (when on the source side of the command).
↑ident↑	A 1- to 6-character name which is to be given to the contents of a DECtape reel mounted on a specified DECtape unit.
<nnn>	A 3-digit protection code which is to be assigned to either one or more destination files or to a specified User File Directory. Refer to Table 4 and Table 5 respectively.
/s	Switches which affect the transfer. All switches in a PIP command string must be preceded by a slash - e.g., /sw/sw - or enclosed in parentheses - e.g., (sw/sw). Refer to Table 6 for a summary of PIP switches.

Table 2 PIP Command String Delimiters

Delimiter	Use and Description
:	The colon delimiter follows and identifies a device name. For example, the device DTA1 is specified as DTA1: in PIP commands.
[]	Square brackets are used to enclose the user DIRECTORY numbers and SFD names (if SFDs are used). For example [40,633] or [40,633,SFD1,SFD2...SFDn] represent the manner in which DIRECTORY numbers can be written.
<>	Angle brackets must be used to enclose a protection code (e.g. <057>) which is to be assigned to either a file or a user file directory (UFD).
,	Commas are used to separate user project and programmer numbers, and file specification groups. For example: dev:[40,633]=dev:file.ext,file.ext<CR>
↑↑	A name to be assigned as an identifier to a DECtape is enclosed within a set of up-arrows (e.g. ↑MACFLS↑).
.	A period delimiter must be the first character of a filename extension. The form on an extension is (.ext).
#	A number symbol is used as a flag to indicate the presence of an octal constant in a filename or a filename extension.
!	An exclamation symbol may be used to delimit a file specification. When used, the ! symbol causes control to be returned to the monitor from PIP and the specified file (or program) to be loaded and run. This function is provided as a user convenience to eliminate the need for several control entries.

Table 2 PIP Command String Delimiters (Cont)

Delimiter	Use and Description
=	The equal sign must be used to separate the destination and source portions of a PIP command.
()	Parentheses are used to enclose magnetic tape options, PIP control switches, and one or more PIP function switches. The form of a command employing parentheses to enclose a series of switches is: dev:file.ext(sw1sw2..swn)=...<CR>

Table 3 PIP Acceptable Device Mnemonics

Mnemonic	Device
CDP	Card Punch
CDR	Card Reader
CTY	Console TTY
DTA	DECtape
DSK	Disk
DPx	Packs
FXx	Fixed-Head
DIS	Display
LPT	Line Printer
MTA	Magnetic Tape
OPR	Operator Terminal
PTP	Paper Tape Punch
PTR	Paper Tape Reader
PLT	Plotter
PTY	Pseudo-TTY
SYS	System Library
TTY	Terminal
TMP	Pseudo-device TMPCO

Table 4 File Protection Codes

Code	Permitted Operations
0	Change protection, rename, write, update, append, read, execute.
1	Rename, write, update, append, read, execute.
2	Write, update, append, read, execute.
3	Update, append, read, execute.
4	Append, read, execute.
5	Read, execute.
6	Execute only.
7	No access privileges. File may be looked up if the UFD permits.

Table 5 UFD and SFD Protection Codes

Code	Permitted Operations
0	Access not permitted.
1	The directory may be read as a file.
2	CREATEs are permitted.
3	The directory may be read as a file and CREATEs are permitted.
4	LOOKUPs are permitted.
5	The directory may be read as a file and LOOKUPs are permitted.
6	CREATEs and LOOKUPs are both permitted.
7	The directory may be read as a file and both CREATEs and LOOKUPs are permited.

Table 6 PIP Control Switch Summary

Switch	Description
/DX	Copy all but specified files
/F	List disk or DTA directory (filenames and ext. only).
/G	Ignore I/O errors.
/H	Image binary processing (mode)
/I	Image processing (mode)
/J	Punch cards in ASCII (output device must be CDP) or convert control characters on terminal output.
/L	List directory.
/N	Delete sequence numbers.
/O	Same as /S switch, except increment is by 1.
/P	FORTRAN output conversion assumed. Convert format control character for line printer listing. /B/P FORTRAN binary.
/Q	Print (this) list of switches and meanings.
/R	Rename file.
/S	Resequence, or add sequence number to file; increment is by 10.
/T	Suppress trailing spaces only.
/U	Copy block 0 (DTA).
/V	Match and count angle brackets (<>).
/W	Convert TABs to multiple spaces.
/X	Copy specified files. (The DX switch tells PIP to copy all but specified files.)
/Y	DECtape to paper tape - If extension is: RMT - A RIM10B paper tape (with terminating transfer word) is produced RTB - A RIM10B paper tape (with RIM loader and terminating transfer word) is produced SAV - A RIM10B paper tape is produced (with neither RIM loader nor terminating transfer word)
/Z	Zero out directory

Table 7 PIP Magtape Switch Summary

Switch	Description
(M2)	Select 200 BPI density.
(M5)	Select 556 BPI density.
(M8)	Select 800 BPI density.
(MA)	Advance MTA one file.
(M#nA)	Advance MTA n files.
(MB)	Backspace MTA one file.
(M#nB)	Backspace MTA n files.
(MD)	Advance MTA one record.
(M#nD)	Advance MTA n records.
(ME)	Select Even Parity.
(MF)	Mark EOF.
(MP)	Backspace MTA one record.
(M#nP)	Backspace MTA n records.
(MT)	Skip to logical EOT.
(MU)	Rewind and unload MTA or DTA.
(MW)	Rewind MTA or DTA.

Examples

The following are examples of commonly used PIP command strings:

EX1 - PIPing an ASCII file from the DISK to the line printer

```
LPT:=DSK:ERROR.SYS<CR>
```

EX2 - Combines two files on disk into one file on DECtape:

```
DTA1:FILCOM.MAC=DSK:FILA.MAC,FILB.MAC<CR>
```

EX3 - Copies a paper tape

```
PTP:=PTR:<CR>
```

EX4 - Specifies that the DECtape on DTA3 be given the identifier "MYFILE" and receive a copy of each file on DTA1.

```
DTA3:↑MYFILE↑/X=DTA1:*.*<CR>
```

TOPS-10 SYSTEM PROGRAM LIBRARY

The programs in the TOPS-10 System Program Library are listed and described in Table 8.

Table 8 TOPS-10 System Program Library

Program	Description
AID	Algebraic Interpretive Dialogue. Each command occupies one line and can be executed immediately or stored as part of a routine for later execution. This interpreter requires no previous programming experience.
ALCFIL	A program used for allocating space for a new file or reallocating space for an existing file in one contiguous region on the disk.
ALGOL	ALGOrithmic Language. A scientifically oriented language that contains a complete syntax for describing computational algorithms.
BACKUP	A program used to save disk files on magnetic tape, and later to restore any or all of these files to disk. Magnetic tape is the medium used for backup storage of disk files and for transporting files between sites.
BASIC	Beginner's All-purpose Symbolic Instruction Code. A time-sharing computer programming language that is used for direct communication between terminal units and computer centers. The language was developed at Dartmouth College.
BATCON	The Batch controller. This program reads a job's control file, starts the job, and controls the job by passing commands and data to it.
BLISS	A programming language that enables users to write programs consisting only of declarations, which establish structure, and expressions, which compute values. It is specifically designed for implementing system software.
BOOTS	A bootstrap program whose main functions are to load a program into core from a SAVE file on a disk unit and/or to dump core as a SAVE file for later analysis.
CHKPNT	A program used to gather the information on the utilization of the DECsystem-10 for accounting and billing purposes.
COBDDT	The COBOL Dynamic Debugging Technique. With COBDDT the user can: 1. Change data-name contents, 2. Set breakpoints, 3. Continue the program, 4. Display the contents of a data-name, and 5. Trace paragraphs and sections.
COBOL	COmmon Business Oriented Language. A programming language used in programming data processing applications.
COMPIL	A utility program that allows the user to type a short, concise command string in order to cause a series of operations to be performed. COMPIL deciphers the command and constructs new command strings for the system program that actually processes the command. Several of the commands that invoke COMPIL are EDIT, COMPILE, CREF, and EXECUTE.
CREF	A program which produces a sequence-numbered assembly listing followed by tables showing cross references for all operand-type symbols, all user-defined operators, and/or all operation codes and pseudo-op codes.
DAEMON	A program for writing all or parts of a job's core area and associated monitor tables onto disk.
DATDMP	A program for dumping the core data base.

Table 8 TOPS-10 System Program Library (Cont)

Program	Description
DDT	The Dynamic Debugging Technique program used for on-line checkout, testing, examination, modification, and program composition of object programs.
DIRECT	A program for producing directory listings of disks and DECtapes.
DSKLST	A program which gives status and statistics of all user disk files at a given time.
DSKRAT	A damage assessment program that scans a file structure and reports any inconsistencies detected.
DTBOOT	A bootstrap program used to save and restore core images on DECtape or magnetic tape. It operates only in executive mode.
DUMP	A program that outputs selected portions of a file in one of the various formats that can be specified by the user.
EDDT	Executive DDT (Dynamic Debugging Technique). A version of DDT used for debugging programs, such as the monitor, in executive mode.
EDIT	A program used to build and edit ASCII text files.
FAILSAFE	A program used to save the contents of the disk on magnetic tape and later restore the saved contents back onto disk.
FILDDT	File DDT (Dynamic Debugging Technique). A version of DDT used for examining and changing a file on disk instead of in core memory. This program is used to examine a monitor for debugging purposes.
FILEX	A general file transfer program used to convert between various core image formats and to read and write various DECtape directory formats and standard disk files.
FORTRAN	FORmula TRANslator. A procedure-oriented programming language designed for solving scientific-type problems by expressing the procedure for their solution as arithmetic formulas. The language is widely used in many areas of engineering, mathematics, physics, chemistry, biology, psychology, industry, military, and business.
FUDGE 2	A program used to update libraries containing one or more relocatable binary modules and to manipulate modules within these libraries.
GLOB	A program used to read collections of relocatable binary modules which have been loaded together (from both library files and separate files) in order to generate an alphabetical cross-referenced list of all the global symbols encountered. When a program is composed of many modules which communicate via global symbols, it is useful to have an alphabetical list of all global symbols with the names and modules in which they are defined and referenced.
GRIPE	A program that accepts text from the user and records it in a disk file for later examination by the operations staff.
INITIA	A program for performing standard system initialization for a particular terminal. It is used to initiate specific programs, such as the spooling programs, on the designated terminal.
LINK	A program that provides automatic loading and relocation of binary programs, producing an optional storage map, and performs loading and library searching. Also, the program loads and links relocatable binary programs and generates a symbol table in core for execution under DDT.

Table 8 TOPS-10 System Program Library (Cont)

Program	Description
LINKER	A program that combines many input modules into a single module for loading purposes. Thus, it allows for independent compilations of modules. Typically, it satisfies global references and may combine control sections.
LINKING LOADER	A program that provides automatic loading, relocation, and linking of compiler- and assembler-generated object modules.
LOGIN	The system program by which the system users gain access to the computing system.
LOOKFL	A program for typing the characteristics of a single disk file, such as creation date and number of words written, on the terminal.
MONEY	A program for reading the system's time accounting file and assigning a monetary charge for each user according to the time and resources that he has used on the system.
MONGEN	The monitor generator dialogue program that enables the system programmer to define the hardware configuration of his individual installation and the set of software options that he wishes to select for his system.
OMOUNT	A program that interfaces with the operator in order to handle requests concerning removable media.
OPSER	The OPerator SERvice program that facilitates multiple job control from a single terminal by allowing the operator or user to initiate several jobs from his terminal.
PIP	The Peripheral Interchange Program which transfers data files from one standard I/O device to another and performs simple editing functions, such as sequencing, trailing blank suppression, and compressing blanks into tabs, and magnetic tape control functions.
PLEASE	A program that provides the user with two-way communication with the operator via an operator's terminal that is reserved for PLEASE commands and the user's terminal.
QMANGER	The Batch queue manager. QMANGR is called by BATCON to schedule jobs by computing and dynamically revising job priorities.
QUEUE	The system program that allows users to add, delete, list, or modify queue entries in the various system queues.
QUOLST	A program that prints the user's quotas for each file structure in his search list and the number of free blocks available in each file structure.
REACT	A program for maintaining administrative control files. It can be used to create, modify, delete or list entries in a file.
RUNOFF	A program that facilitates the preparation of typed or printed manuscripts by performing formatting, case shifting, line justification, page numbering, titling, and indexing.
SCRIPT	A program that sends predetermined sequences of characters over multiple pseudoterminals in order to simulate a load on the system for testing, measurement, and analysis.
SETSRC	A program that allows the user to list or change his search list.
SOUP	The SOftware Updating Package that consists of a set of programs for facilitating the updating of system or user source files.

Table 8 TOPS-10 System Program Library (Cont)

Program	Description
SPRINT	The Batch input stacker. SPRINT reads any sequential input stream, sets up the job's control file and data files, and enters the job into the Batch input queue.
SYSDPY	A variation of the SYSTAT program which runs on a keyboard display terminal (at up to 2400 baud). SYSDPY maintains a dynamic display of system status by periodically altering lines of the display to replace old information with the latest information.
SYSERR	SYSERR is the report generating portion of the DECsystem-10 and DECSYSTEM-20 error detection, recovery, and reporting system. As an error is detected by the monitor, various pieces of information describing pertinent hardware and software status are gathered and appended to a disk file. SYSERR is a user-mode program which lists the contents of this file at the direction of the command string.
SYSTAT	A program that outputs to the user's terminal status information on the system as a whole, on selected aspects of the system, or on a selected job or set of jobs.
TECO	A sophisticated Text Editor and Corrector program that allows simple editing requests, character string searches, complex program editing, command repetition, and text block movement. TECO editing is performed on files consisting of ASCII characters.
UMOUNT	A program for user interfacing for the handling of requests concerning removable media.

TOPS-10 COMMAND LANGUAGE

The TOPS-10 Operating System supports approximately 96 commands. The conventions used to illustrate these commands are described in Table 1. The individual commands are arranged in alphabetical order in Table 2.

Note that the complete command format has been shown for the commands. Depending on the circumstances, only part of this format may be required. Refer to the DECsystem-10 Operating System Commands manual to determine the arguments required for a particular task. In addition, the commands can be abbreviated as long as the abbreviation does not conflict with any other command abbreviation.

Many command strings allow wild-card characters to be used in place of alphanumeric characters. These characters permit more than one file or directory to be referenced by a single specification. Two such wild-card characters are available:

1. * - The asterisk is a wild card for an entire field. When positioned in the appropriate context, it means:

		Examples	
a.	any filename or extension	*.EXT	FILNAM.*
b.	any project number or programmer number (also, any subfile directory)	[*,1164]	[27,*]

   Note that *.* and [*,*] are also possible.

2. ? - The question mark is a wild card for a single character. It can be used in any field mentioned above, provided the * does not share the field. It means: any character.

   Examples:

   .EX? FI???.EX? ?ILNAM. [27,116?] [*,11??]

In addition, the directory name can be specified with the project number, the programmer number, or both numbers missing.

ERROR MESSAGES

TOPS-10 operating systems use four types of stop codes.

DEBUG - If a priority interrupt is in progress, the condition is not immediately harmful to the system or any job. The monitor types out a message on the console terminal and continues. If no priority interrupt is in progress, a DEBUG stopcode acts the same as a JOB stopcode.

JOB - If no priority interrupt is in progress, the condition jeopardizes the integrity of the current job. The monitor sends a message to both the console terminal and the user's terminal and aborts the job. If a priority interrupt is in progress, then a JOB stopcode acts like a STOP stopcode.

STOP - This condition jeopardizes the integrity of the entire system. The monitor sends a message to the console terminal, aborts all jobs, and reloads the system.

HALT - This condition is so serious that the monitor is not going to do anything that might affect stored data. The system executes a HALT instruction and waits for the operator to initiate a reload.

Table 11 lists and describes the STOP CODES associated with a TOPS-10 operating system (6.03 release).

Table 9 TOPS-10 Command Conventions

Convention	Description
adr	An octal address.
arg	A letter or word specifying the desired function of the command.
control file	The name of the control file for the Batch System.
core	Decimal number of blocks (n or nK) or pages (nP) of core.
dev:	Any physical (or logical, normally) device name (e.g., MTA:). The colon must be included.
devn:	Any physical device name of three characters followed by a unit number of one to three numerals (e.g., DTA3:). The colon must be included.
devSnn:	Any physical device name of three characters followed by the letter S and a station number (e.g., LPTS2:). The colon must be included.
[directory]	A designation identifying a particular disk area. This designation can be in the form [proj,prog] which identifies a UFD or [proj,prog,sfd,sfd, ...] which identifies a sub-file directory path branching from a UFD. The square brackets are required.
drives	The physical drives on which a unit is to be mounted.
file.ext	Any legal filename from one to six characters followed by a dot and an extension of zero to three characters.
file structure	The name of a particular disk. This name is usually in the form DSKA, DSKB, etc.
input specifications	File specifications for the disk files to be processed.
jobn	A user's job number assigned by the system.
jobname	A name of up to six characters of the job being entered into one of the system queues.
lh	Left half of a 36-bit word.
logdev:	Any logical device name from one to six alphanumeric characters. The colon should be included.
log file	The name to be given to the log file created by the Batch system.
n or m	A number.
x	A letter.
<nnn>	A three-digit octal code indicating the protection of a file. This code can appear only on the output side of the command string and must be enclosed in angle brackets.
prog	A program name of six or fewer characters.
rh	Right half of a 36-bit word.
/S	One or more switches used to modify the command string.
↑tape id↑	A one to six character identifying name recorded on a DECtape.
text	A message to be sent to the designated user or terminal.

Table 9 TOPS-10 Command Conventions (Cont)

Convention	Description
[user number]	A numeric identification assigned to the user for the purpose of gaining access to the system. It is usually two numbers separated by a comma.
=	An equal sign used in command strings to separate the output specification (left of the equal sign) from the input specification (right of the equal sign).

Table 10 TOPS-10 Command Summary

Command	Description
ALCFIL	R ALCFIL<CR> Allocates space for a new file or reallocates space for an existing file in one contiguous region on the disk.
ASSIGN	ASSIGN dev:logdev:<CR> ASSIGN devSnn:logdev:<CR> ASSIGN devn:logdev:<CR> Allocates an I/O device to the user's job without operator intervention.
ATTACH	ATTACH jobn [user number]<CR> Detaches the current job and connects the terminal to the specified detached job.
BACKSPACE	BACKSPACE MTAn:m FILES<CR> BACKSPACE MTAn:m RECORDS<CR> Spaces a magnetic tape backward the specified number of files or records.
CCONTINUE	CCONTINUE<CR> Continues the program from the point at which it was interrupted, but leaves the terminal in monitor mode.
CLOSE	CLOSE dev:<CR> Terminates I/O currently in progress on the specified device, performs the CLOSE UUO, but does not release the device.
COMPILE	COMPILE dev:file.ext [directory]/S,...<CR> Produces relocatable binary files (.REL files) for the specified source files.
CONTINUE	CONTINUE<CR> Continues the program from the point at which it was interrupted.
COPY	COPY dev: ↑tape id↑ file.ext [directory] <nnn> = dev:file.ext [directory], file.ext [directory], ...<CR> Transfers files from one I/O device to another.
CORE	CORE core<CR> Types or modifies the amount of core assigned to the user's job.
CPUNCH	CPUNCH jobname = dev:file.ext [directory]/s, ...<CR> Places entries into the card punch output spooling queue.

Table 10 TOPS-10 Command Summary (Cont)

Command	Description
CREATE	CREATE file.ext<CR> Opens a new file on disk for creation with LINED.
CREF	CREF<CR> Lists on LPT: any cross-referenced listing files generated by a previous COMPILE, LOAD, EXECUTE, or DEBUG command.
CSTART	CSTART adr<CR> Begins execution of a program that was either loaded with a GET command or interrupted, but leaves the terminal in monitor mode.
D(eposit)	D lh rh adr<CR> Deposits information in the user's core area.
DAYTIME	DAYTIME<CR> Types the current date followed by the time of day.
DCORE	DCORE dev:file.ext [directory]<CR> Writes a core image file of the user's core area.
DDT	DDT<CR> Copies the saved program counter and starts the program at the beginning address of DDT if DDT was loaded with the program (automatic in 6.01).
DEASSIGN	DEASSIGN dev:<CR> Returns devices assigned to the user's job to the monitor's pool of available devices and clears logical names.
DEBUG	DEBUG dev:file.ext [directory]/s, ...<CR> Produces relocatable binary files (.REL files) for the specified source files, loads the .REL files along with an appropriate system debugging program, and prepares for debugging.
DELETE	DELETE dev:file.ext [directory], ...<CR> Deletes files from DECtape or disk.
DETACH	DETACH<CR> Disconnects the terminal from the current job without affecting the status of the job.
DIRECT	DIRECT dev:file.ext [directory] = dev:file.ext [directory]/s, ...<CR> Lists the directory entries for the specified arguments.
DISMOUNT	DISMOUNT dev:/s, ...<CR> Returns, via the operator, devices assigned to the user's job to the monitor's pool of available devices.
DSK	DSK jobn<CR> Types disk usage for the combined structures of the specified job.
DTCOPY	R DTCOPY<CR> Copies contents of one DECtape to another, clears the blocks on a DECtape and clears the directory, compares two DECtapes, and/or loads and writes a bootstrap loader.

Table 10 TOPS-10 Command Summary (Cont)

Command	Description
DUMP	DUMP/S ...<CR> Writes a core image file, analyzes the file written, and provides printed output.
DUMP	R DUMP<CR> Provides printable output of data files in specified forms and modes.
E(xamine)	E adr<CR> Examines the specified core location in the user's area.
EDIT	EDIT file.ext<CR> Opens the specified file already existing on disk for editing with LINED.
EOF	EOF MTAn:<CR> Writes an end-of-file mark on the specified magnetic tape.
EXECUTE	EXECUTE dev:file.ext [directory]/s, ...<CR> Produces relocatable binary files (.REL files) for the specified source files, loads the .REL files, and begins execution.
FAILSAFE	R FAILSAFE<CR> Saves and restores disk files.
FILCOM	R FILCOM Compares two versions of a file and outputs any differences.
FILE	FILE arg, ↑tape id↑, file.ext, file.ext, ...<CR> Provides remote control, via the operator, of DECtape-to-disk and disk-to-DECtape transfers.
FILEX	R FILEX<CR> Converts between various core image formats, and reads and writes various directory formats.
FINISH	FINISH dev:<CR> Terminates I/O in progress on the specified device and performs the RELEASE UUO and DEASSIGN command.
FUDGE	FUDGE<CR> Creates a library REL file by reading a temporary file generated by a previous COMPILE, LOAD, EXECUTE, or DEBUG command containing the /FUDGE switch.
FUDGE2	R FUDGE2<CR> Updates files containing relocatable binary programs, and manipulates the programs within these files.
GET	GET dev:file.ext [directory] core<CR> Loads a core image from the specified device, but does not begin execution.
GLOB	R GLOB<CR> Reads multiple binary files to produce an alphabetical cross-referenced listing of all global symbols encountered.

Table 10 TOPS-10 Command Summary (Cont)

Command	Description
GRIPE	R GRIPE<CR> Accepts text from a user and records it in a disk file for the operations staff.
HALT	HALT<CR> or ↑C Stops the job and stores the program counter in the job data area. Control C can be used at user level as well as at monitor level.
HELP	HELP dev:prog<CR> or HELP dev:*<CR> Outputs useful documentation on various system features.
INITIA	INITIA<CR> Performs standard system initialization for the terminal issuing the command.
JCONT	JCONT jobn<CR> Continues the specified job if it was in a ↑C state because of a call to the device error message routine (HNGSTP).
KJOB	KJOB logfile = file structures/s<CR> Gives up access to the system.
LABEL	LABEL DEV: ↑tape id↑<CR> Writes an identifier onto a DECtape.
LIST	LIST dev:file.ext [directory]/s, ...<CR> Lists the specified files on the line printer.
LOAD	LOAD dev:file.ext [directory]/s, ...<CR> Produces relocatable binary files (.REL files) for the specified files and loads the .REL files generated.
LOCATE	LOCATE nn<CR> Establishes, logically, the user's job at a specified station.
LOGIN	LOGIN user number/s ...<CR> Provides access to the system.
MAKE	MAKE dev:file.ext [directory]<CR> Opens a new file on disk for creation with TECO.
MOUNT	MOUNT dev:logdev:/s drives<CR> Allocates an I/O device to the user's job via the operator.
OPSER	R OPSER<CR> Provides multiple job control from a single terminal.
PJOB	PJOB<CR> Outputs the job number to which the terminal is currently attached.
PLEASE	PLEASE dev:prog! text<CR> Provides two-way communication between the user and the operator.
PLOT	PLOT jobname = dev:file.ext [directory]/s, ...<CR> Places entries into the plotter output spooling queue.

Table 10 **TOPS-10 Command Summary (Cont)**

Command	Description
PRESERVE	PRESERVE file.ext, file.ext, ...<CR> Renames the specified files with the standard protection inclusively ORed with 100.
PRINT	PRINT jobname = dev:file.ext [directory]/s, ...<CR> Pleaces entries into the line printer output spooling queue.
PROTECT	PROTECT file.ext<nnn>, file.ext<nnn>, ...<CR> Sets the specified files to the requested protections.
PUNCH	PUNCH jobname = dev:file.ext [directory]/s, ...<CR> Places entries into the paper tape punch output spooling queue.
QUEUE	QUEUE queue name:jobname = input specifications<CR> Enters items into the specified system queue.
QUOLST	R QUOLST<CR> Types the used, loggin-in quota, and logged-out quota for each file structure to which the user has access, followed by the number of free blocks left on that structure.
R	R file.ext core<CR> Loads a core image from the system device (SYS:) and starts it at the location specified within the file.
REASSIGN	REASSIGN dev:jobn<CR> Gives the specified device to the designated job.
REATTA	R REATTA<CR> Transfers the job from the current terminal to the specified terminal.
REENTER	REENTER<CR> Starts the program at an alternate entry point specified by the program.
RENAME	RENAME new = old, new = old, ...<CR> Changes the name and protection of one or more files on DECtape or disk.
RESOURCES	RESOURCES<CR> Outputs the names of all available devices (except for terminals and PTYs), all file structures, and all physical units not in file structures.
REWIND	REWIND dev:<CR> Rewinds a magnetic tape or DECtape.
RUN	RUN dev:file.ext [directory] core<CR> Loads a core image from the specified device and starts it at the location specified within the file.
SAVE	SAVE dev:file.ext [directory] core<CR> Writes a core image of the user's core area on the specified device.
SCHED	SCHED<CR> Outputs the schedule bits set by the last SET SCHED command.

Table 10 TOPS-10 Command Summary (Cont)

Command	Description
SEND	SEND dev:text<CR> SEND jobn text<CR> Provides a one-way interconsole line of communication.
SET BLOCKSIZE	SET BLOCKSIZE dev:nnnn<CR> Sets the default blocksize for the specified magnetic tape.
SET BREAK	SET BREAK AT adr ON arg, ...<CR> SET BREAK NO arg, ...<CR> SET BREAK NONE<CR> Sets address break in program according to specified conditions used with KI10 processors only.
SET CDR	SET CDR file<CR> Sets the filename for the next card-reader spooling intercept.
SET CPU	SET CPU CPxn<CR> SET CPU NO CPxn<CR> SET CPU ALL<CR> SET CPU ONLY CPxn<CR> Sets the CPU specification for the job. This command is only available on multiprocessor systems (1055, 1077) and requires certain bits be set in the privilege word.
SET DENSITY	SET DENSITY dev:nnn<CR> Sets the default density for the specified magnetic tape.
SET DSKFUL	SET DSKFUL ERROR<CR> SET DSKFUL PAUSE<CR> Controls the job when the user has exhausted his disk space.
SET DSKPRI	SET DSKPRI n<CR> Sets the priority for the job's disk operations (data transfers and head positionings). Requires certain bits to be set in the privilege word.
SET HPQ	SET HPQ n<CR> Sets the high priority scheduler run queue for the job. Requires certain bits to be set in the privilege word.
SET PHYSICAL	SET PHYSICAL LIMIT core<CR> SET PHYSICAL GUIDELINE CORE<CR> Specifies when the job will go virtual and specifies a guideline for the page fault handler if GUIDELINE is designated. Used with KI10 processors only.
SET SPOOL	SET SPOOL dev:, dev:, ...<CR> SET SPOOL ALL<CR> SET SPOOL NONE<CR> SET SPOOL NO dev:, dev:, ...<CR> Adds devices to or deletes devices from the list of spooled devices for this job.
SETSRC	R SETSRC<CR> Manipulates the job's search list or system's search list.
SET TIME	SET TIME n<CR> Sets the central processor time limit for the job.

Table 10 TOPS-10 Command Summary (Cont)

Command	Description
SET TTY	SET TTY NO arg<CR> SET TTY arg Sets properties to be associated with the terminal.
SET VIRTUAL LIMIT	SET VIRTUAL LIMIT core<CR> Specifies the limit on the virtual memory for a job. Used with KI10 processors only.
SET WATCH	SET WATCH arg, arg, ...<CR> SET WATCH ALL<CR> SET WATCH NONE<CR> SET WATCH NO arg, arg, ...<CR> Sets the output of incremental job statistics.
SKIP	SKIP MTAn:m FILES<CR> SKIP MTAn:m RECORDS<CR> SKIP MTAn:EOT<CR> Moves the specified magnetic tape forward the designated number of files or records or to the logical end of tape.
SSAVE	SSAVE dev:file.ext [directory] core<CR> Writes a core image of the user's core area on the specified device. When it is loaded with a GET (or RUN) command, the high segment will be sharable.
START	START adr<CR> Begins execution of a program either previously loaded with the GET command or interrupted while running.
SUBMIT	SUBMIT jobname = control file, log file/s<CR> Places entries into the Batch input queue.
SYSTAT u	SYSTAT/S<CR> Prints information about the current status of the system.
TECO	TECO dev:file.ext [directory]<CR> Opens the specified file for editing with TECO.
TIME	TIME jobn<CR> Outputs the running time for the specified job.
TPUNCH ?u	TPUNCH jobname = dev:file.ext [directory]/s, ...<CR> Places entries into the paper tape punch output spooling queue.
TTY	TTY NO arg<CR> TTY arg<CR> Sets properties to be associated with the terminal.
TYPE	TYPE dev:file.ext [directory]/s, ...<CR> Types the specified files on the user's terminal.
UNLOAD	UNLOAD dev:<CR> Rewinds and unloads the specified magnetic tape or DECtape.
USESTAT	USESTAT<CR> or ↑T Prints information on the terminal concerning the user's job. Control T can be used at user level also.

Table 10 **TOPS-10 Command Summary (Cont)**

Command	Description
VERSION	VERSION<CR> Outputs the version number of a program on the terminal.
WHERE	WHERE dev:<CR> Outputs the station number of the specified device.
ZERO	ZERO dev: [directory]<CR> Clears the directory of the specified device.

TOPS-10 Monitor Stopcodes
This subsection describes the TOPS-10 monitor stopcodes. A stopcode message is generated when the monitor detects a serious error in its data base. This subsection describes the five types of monitor stopcodes, and briefly explains and illustrates continuable stopcodes. An alphabetical listing of all monitor stopcodes is presented at the end of this section.

MONITOR STOPCODE DEFINITION
When the TOPS-10 monitor encounters an internal error it issues a three-letter message called a stopcode. The stopcode is displayed on the console terminal (CTY) and alerts you of possible system failure, depending on the severity of the error. The stopcode message is generated by a stopcode macro when the monitor detects an error in the data base. This macro prints the following message on the CTY:

```
?CPUn monitor error. Stopcode name is xxx
```

Where:

n is a symbol identifying the CPU.

xxx is a symbol identifying the error.

The five types of stopcodes described in this specification include the following.

1. DEBUG
2. JOB
3. STOP
4. CPU
5. HALT

DEBUG Stopcodes
DEBUG stopcode is a stopcode that is not immediately harmful to any job or to the system. When the monitor encounters an internal error at the interrupt level, a dump is performed and processing continues. The following message prints on the CTY:

```
?CPUn monitor error. Stopcode name xxx
[Continuing system]
```

JOB Stopcodes
A JOB stopcode indicates that an internal error endangers the integrity of the job that currently is running. The monitor aborts the current job and continues processing. The following message is printed on the CTY.

```
?CPUn monitor error. Stopcode name xxx
[Aborting job]
```

On the user's terminal, the following message is displayed:

```
?Monitor error; UUO at addr
```

Where:

addr is one of the following:

- User location n;
- Exec location n; Exec called from exec location m
- Exec location n; Exec called from user location m
  m and n are virtual memory addresses

STOP Stopcodes
A STOP stopcode indicates an internal error that endangers the integrity of the entire system. All jobs are aborted and the system begins to automatically dump and reload the monitor. The monitor prints the following message on the CTY:

?CPUn monitor error. Stopcode name **xxx**
Reload monitor

If the monitor obtains the necessary information, it prints a supplementary message on the CTY of the form:

JOB **jobn** on TTY**nnn** running **name**
UUO is **octal representation** at user PC **address**
File **filespec**

Where:

jobn	is the number of the job causing the error.
nnn	is the number of the job controlling the terminal.
name	is the name of the program running for that job.
octal representation	is the octal representation of the octal monitor call failing for that job.
address	is the value of the program counter for that job.
filespec	is the file specification for the file being accessed.

CPU Stopcodes
A CPU stopcode has the same effect as a STOP stopcode. When a CPU stopcode occurs on a single-processor system, or on the last processor on a multiprocessor system, the effect is the same as a STOP stopcode. All user jobs are aborted and the system begins to automatically dump and reload the monitor. Otherwise, the CPU stopcode aborts the jobs and a dump is taken, but the monitor must be reloaded manually. The following message is printed on the CTY:

?CPUn monitor error. Stopcode name **xxx**
Reload monitor

For SMP systems in which a CPU stops, the following message is displayed:

[Stopping CPU]

If the monitor obtains the necessary information, it prints a supplementary message on the CTY of the form:

Job **jobn** on TTY**nnn** Running **name**
UUO is **octal representation** at user PC **address**
File **filespec**

Where:

jobn	is the number of the job causing the error.
nnn	is the number of the job controlling the terminal.
name	is the name of the program running for that job.
octal representation	is the octal representation of the monitor call failing for that job.
address	is the value of the program counter for that job.
filespec	is the file specification for the file filespec being accessed.

HALT Stopcodes
A HALT stopcode indicates a fatal error and affects the entire system. The monitor cannot automatically reload. The system halts, and you must manually dump and reload the monitor. (See Part VI of the TOPS-10 Operator's Guide.)

HALT stopcodes generates the following message:

```
KL HALTED.
```

Continuable Stopcode
Continuable stopcodes occur when the monitor executes a stopcode macro; dumps the memory image; and continues the system automatically. HALT and STOP stopcodes are not continuable. A CPU stopcode is continuable on SMP systems in which a CPU stopcode stops the CPU and displays the message:

```
[Stopping CPU]
```

Other types of CPUs that receive the CPU stopcode must be manually reloaded.

The following three examples illustrate continuable DEBUG stopcodes:

Example 1:

```
?CPU0 monitor error. Stopcode name is ICN

CPU Status Block on 30-May-80 19:29:21

CONI APR, = 001060,,004102
CONI PI, = 000000,,000777
CONI PAG, = 000000,,020000
DATAI PAG, = 500100,,000002
[Dumping on DSK:CRASH.EXE[1,4]]
[Continuing system]
```

Example 2:

```
?CPU1 monitor error. Stopcode name is EUE
Job 5 on TTY1 running DDT User [1,2]
UUO is 0 at user PC 002472

CPU Status Block at 4-Oct-79 8:16:36

APRID = 000231,,342002
ERA = 600000,,040513
CONI APR, = 007760,,000003
CONI PI, = 000000,,000377
CONI PAG, = 000000,,620001
DATAI PAG, = 700100,,002255
AR ARX Data Word = 000000,,057000
IO Page Fail Word = 000000,,000000

SBUS Diags:

CNTRLR FNC 0          FNC 1
000004 007040,,040610 000200,,000000

[Dumping on DSK:CRASH.EXE[1,4]]
[Continuing system]
```

Example 3:

```
?CPU1 monitor error. Stopcode name is IEZ
File DSKE0:OPSER.LOG[1,2]
Job 1 on CTY running OPSER User [1,2]
[Dumping on DSK:CRASH.EXE[1,4]]
[Continuing system]
```

The following example illustrates a JOB stopcode message that is displayed on the CTY:

```
%DECsystem-10 not running

?CPU0 monitor error. Stopcode name is IME
Job 1 on TTY5 running FH702 User [1,2]
UUO is 47240040770 at user PC 006477

CPU Status Block on 24-Jan-84 15:06:00

APRID = 640336,364654
ERA = 024000,,006451
CONI APR, = 007760,000001
CONI PI, = 000000,,000377
CONI PAG, = 000000,,660001
DATAI PAG, = 700100,,001340
AR ARX Data Word = 000000,,000000
IO Page Fail Word = 000000,,000000

SBUS Diags:

CNTRLR FNC 0           FNC 1
000004 001740,,017321 000200,,000000
000010 006160,,006603 000500,,001000

[Dumping on BLUI:CRASH.EXE[1,4]]
[Aborting Job]

[DECsystem-10                              Continued]
```

The same JOB stopcode message would also appear on your terminal(TTY)S in addition to the following message:

```
?Monitor error at user PC 006476
```

List Of Stopcodes

A list of stopcodes for all systems that run the TOPS-10 monitor is presented in Table 11 in alphabetical order. The list shows the name of each stopcode, the calling module, the type of stopcode, a phrase message (for which the name is a symbol), and a brief explanation of the containing routine, the error that caused the stopcode, and any data items that can be helpful in analyzing dumps.

Table 11 TOPS-10 Stop Code Summary

Name	Module	Type	Message and Explanation
28B	XTCSER	DEBUG	DA28 Is Broken This stopcode occurs when various error bits get lit while talking to the DA28.
4IF	FSXKON	DEBUG	RS04 Isn't Fancy FSXERR, FSXECC, FSXUNL, FSXERR, and FSXECC are stopcode-only routines. This stopcode occurs when the FILIO module tries to perform an RP04-only function on an RS04, for example, an ECC error recovery or an unload.
5WE	D85INT	DEBUG	DC75 Wrong PDP-11 Code Executing D75III is used when the PDP-11 is recognized as a DC75 PDP-11. This stopcode occurs when the DC75 code version number is not what was expected.
	Data Items:		T1 = version of code executing. Should have been 1.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
6DD	D6SINT	DEBUG	Too Much Direct Data The DN60 front end has violated the DTE queued protocol. This indicates a problem in the front end.
6DI	D6INT	DEBUG	Data Interrupt The front end experienced an interrupt at the wrong time. The DTE protocol was violated. This indicates a problem in the front end.
6ID	D6SINT	DEBUG	Too Much Indirect Data The DN60 front end has violated the DTE queued protocol. This indicates a problem in the front end.
6MS	D76INT	DEBUG	DC76 Message Short GTMSGH gets a word from the Tops-10 queue. This stopcode occurs when the Tops-10 queue is empty, but it should contain data.
6QF	D76INT	DEBUG	DC76 Queue Full PUTMSG queues a message to the PDP-11. This stopcode occurs when the Tops-11 queue is full, but the DC76 is neither halted nor hung.
	Data Items:		P1 = Tops-11 putter; P2 = Tops-11 queue
8BI	D78INT	JOB	Blank I/O Word In converting KA10-style IOWDs to 22-bit format, a zero IOWD was encountered.
8IN	D78INT	JOB	Input Character Count Non-zero The input character count was nonzero in the window slot for the DAS78 at the beginning of an INPUT UUO (when the PDP-11 became ready).
8NC	D78INT	JOB	Not Enough Free Monitor Core D78INT was unable to obtain enough monitor free core to convert a KA10-style IOWD list to 22-bit format.
8ON	D78INT	JOB	Output Character Count Is Not Equal To Zero When the PDP-11 became ready during an OUT UUO, the output character count in the window slot was nonzero.
8PI	D78INT	JOB	Positive IOWD In converting from KA10-style IOWDs to 22-bit format, a positive IOWD was encountered.
8VI	D78INT	DEBUG	Version Incorrect The PDP-11 on the DAS78 came up with a version number other than one.
AAD	FILFND	DEBUG	Access Table Already Dormant ATNLNK unlinks an access table from a name block (NMB) ring. This stopcode occurs when an attempt is made to make an access table dormant, but the table is already dormant.
	Data Items:		T1 = location of access table; T2 = location of predecessor; T3 = location of next in ring

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
AAO	KSSER	JOB	Access Allowed Is Off After paging a job, the monitor converts virtual IOWDs into absolute IOWDs. This stopcode occurs while checking the access bits for a page pointed to by the IOWD, and access to that page is not allowed.
	Data Items:		T1 = total number of words accumulated so far; T2 = number of words for current page; T3 = current page number within this segment; T4 = next page number within this segment. T4 is the page that was not allowed access.
ADn	COMMON	CPU	Address Parity Error For CPU n This stopcode occurs when Bit 29 is on in the CONI APR. This is a serious hardware error. Call your Field Service representative.
AES	FILFND	JOB	Abnormal End Of Search List SLXAES is called from several places in FILFND. This stopcode occurs for many reasons, such as unexpectedly encountering the end of a search list.
AHS	ONCMOD	HALT	Already Have Structure DMKSTR sets up structures according to tables in ONCMOD for the DESTROY option. This stopcode occurs when a duplicate structure name is found in the table.
	Data Items:		T1 = duplicate structure name; P2 = address of structure data block; P3 = address of previous structure data block or DIFSTR if this is the only structure; P4 = system structure number of this structure
ANU	FILIO	DEBUG	AU Not Owned By Us UPAU obtains the Alter-UFD (AU) resource. This stopcode occurs when there is no UFD data block (UFB) for the given DDB or a job requested an Alter-UFD (AU) resource that was already owned by someone else.
	Data Items:		P1 = 0 if no UFB, otherwise the UFB for DDB; P2 = not applicable if no UFB, otherwise job number trying to release the AU resource; UFBAUJ(P1) = job owning the AU resource
AOC	FILFND	DEBUG	Already Own CB GETCB gets the core block (CB) resource. This stopcode occurs when a job requests the CB resource, but already owns it.
	Data Items:		J = job number
APF	VMSER	DEBUG	Allocated Page Free This stopcode occurs when the monitor finds an unallocated page of memory in the list of pages allocated to a segment.
	Data Items:		P2 = disk address-1; P3 = current physical page allocated; P4 = number of pages; T3 = byte pointer to the map slot

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
AR1	ONCMOD	DEBUG	ASKDEC Returned CPOPJ1 CHGASL changes the active swapping list. This stopcode occurs when the routine to input a decimal number returns POPJ1.
ARD	DTESER	STOP	Run Away Driver After receiving a Tops-10 doorbell signaling an indirect transfer, a check is made to make sure that the byte count of the next part of this indirect transfer is greater than or equal to zero. This stopcode occurs when the byte count is less than zero.
	Data Items:		T1 = current desired byte count; T2 = negative byte count that caused the stopcode
ARF	CORE1	STOP	Attempt To Return Free Page GVPAGS returns pages to the free-core list. This stopcode occurs when the monitor checks its table of free pages before returning a page and finds that the page is already marked as being free.
	Data Items:		T1 = first page on free-core list; T2 = page being returned to the free-core list (this is the page that caused the stopcode); T3 = number of pages returned so far; T4 = bit being tested in the page table (400000,,0); PAGTAB(T2) = page-status bits (status-bit definitions can be found in module S.MAC)
ARM	FILFND	DEBUG	Access Rings Messed Up ATNLNK unlinks an access table from a name block ring. This stopcode occurs when an access ring is not linked to any access table.
AVE	QUESER	DEBUG	Already Have EQ AVESTP is a stopcode-only routine. This stopcode occurs when a job has the Enqueue/Dequeue Wait resource when it should not have it.
	Data Items:		J = job number
BAA	DTESER	STOP	Buffer Already There T10GTC allocates a To-10 DTE buffer. This stopcode occurs when a buffer already exists for this DTE.
	Data Items:		T1 = Tops-10 DTE buffer-is-allocated bit (400000,,000000); F = ETD address; ETDBUF(F) = where DTE buffer is allocated, bit is set when a buffer is allocated
BAC	CORE1	DEBUG	Bit Already Clear SETZRS sets zeros in a table. This stopcode occurs when the SETZRS routine attempts to zero bits that are already zero.
	Data Items:		T2 = still on the stack = AOBJN pointer to tables; T1 = number of bits to clear

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
BAD	FILFND	JOB	Block Already Dormant ATSFRO puts an access table in the free-core list. This stopcode occurs when an attempt is made to make the access table dormant, but the table is already dormant.
	Data Items:		T1 = location of access table
BAO	FILIO	DEBUG	Bit Already One TAKBLK allocates blocks from the disk. This stopcode occurs when the monitor attempts to allocate a block that is already allocated.
	Data Items:		PDL = cluster address
BAZ	FILIO	DEBUG	Bit Already Zero GIVBLK returns disk blocks. This stopcode occurs when the monitor attempts to return blocks that are already free; this occurs when a damaged file is deleted. If this stopcode occurs with any frequency, it is suggested that you run DSKRAT on the structure indicated in the stopcode macro printout on the CTY.
BBS	D85INT	STOP	Bad Byte Size DLBP makes the DL10 the byte pointer for data. This stopcode occurs when the number of bytes per word supplied is illegal. The number of bytes per word must be from 2 to 6.
	Data Items:		T4 = wrong byte size
BDN	DTESER	STOP	Bad Device Number GTETDS sets up a DTE control block address and places it in AC F. It is called with F = CPU#,,DTE#. This stopcode occurs when either the CPU number or the DTE number is not in range.
	Data Items:		T1 = 0 if caller tried to call this routine with -1 as the DTE number; T1 = illegal CPU number if that was the reason for the stopcode; T3 = illegal DTE number if that was the cause of the stopcode; CPUN = legal number of CPUs; C CODTN(CPU OFFSET) = legal number of DTEs; F = CPU#,,DTE#
BFC	D60SER	DEBUG	Bad Function Code The front end controller received a bad function code. This is a monitor error.
BFO	TAPUUO	DEBUG	Better Find One INVERT generates a transfer list for a read backwards. This stopcode occurs when the end of the original transfer list for a read backwards is not found.
	Data Items:		T2 = head of old transfer list; T3 = item number to find

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
BIN	FILIO	STOP	Block Number Is Negative MONRED reads a block or a series of blocks. This stopcode occurs when this routine is called with a negative block number.
	Data Items:		T1 = IOWD for data; T2 = block number
BMB	DTESER	DEBUG	Bad Message Block Pointer The address of the message passed to DTESER to output to the -11 was zero.
BMR	FILUUO	JOB	Block Missing From RIB ALLP0A is used to work with a RIB whose blocks are allocated but not used. This stopcode occurs when the RIB for a file shows that the file has more blocks than actually exist.
	Data Items:		T3 = missing block
BNF	COMMON	HALT	BOOTS Not Found BOOT-11 was not found on the specified device, nor was it found in blocks 4-7 of any other structure. BNR FILUUO JOB Block Not RIB NOTOLD creates a new name in a directory block. This stopcode occurs when a pointer to a block is not found in the RIB.
	Data Items:		P2 = block that is being looked for
BNT	FILFND	DEBUG	Block Not There UFORSS gets a UFD or an SFD access block. This stopcode occurs when a core block (AT or UFB) that is known to exist is not found.
BNZ	CORE1	DEBUG	Bit Not Zero CSETOS sets bits in a table. This stopcode occurs when one of the bits to be set in a table is already set.
	Data Items:		T3 = number of bits to set; T4 = address,,position
BPE	CLOCK1	JOB	Breakpoint PC EXEC Mode The user PC on a control-D to a DDT unsolicited breakpoint trap did not have the USRMOD bit, although the user was supposed to be in user mode.
	Data Items:		T1 = PC word that the user is currently running
BPF	CLOCK1	JOB	Breakpoint PUTWRD Failed The attempt to store the return PC (in processing an unsolicited ^D breakpoint) into the location pointed to by .JBBPT failed even though the address had been previously address checked.
BPT	FILFND	JOB	Bad Search List Pointer SLXBPT is used only for this stopcode and is called from FILUUO. This stopcode occurs when an attempt is made to build the search list, but no search list can be found.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
BRC	COMCON	DEBUG	Bad Return From CMPBIT SAVEXE saves a virtual memory system core image. This stopcode occurs when the routine that computes the flag bits for a directory entry gives an error return.
BSN	SEGCON	STOP	Bad Segment Number COMIT compares the job number with the right half of JBTSGN(T1). This stopcode occurs when the right half of J and the right half of JBTSGN(T1) are not the same.
	Data Items:		T1 = high-segment number of job; J = job number
			BSY XTCSER DEBUG DA28 Busy The BUSY bit was (still) on when the DA28 interrupted.
BTC	DTESER	DEBUG	Bad Transfer Count(s) The message from the DTE specified a transfer size greater than the message size.
	Data Items:		T2 = Transfer size; T1 = Message size
BWA	FILIO	JOB	Block Went Away SETLST sets up an I/O list block. This stopcode occurs when NXTBLK, which is used to return the next block address, gives an error or a non-skip return that indicates end-of-file, writing in the middle of a file, or other similar error.
	Data Items:		J = job number
CAL	DTESER	DEBUG	CALUSR Called By Wrong User Routine CALUSR in DTESER calls a service routine based on the current user (NOBODY, ANF, DECnet, and so forth) of the DTE. This stopcode occurs for those users who should not be going through CALUSR.
CAO	FILUUO	DEBUG	Cluster Address Odd ADJALC allocates the initial blocks for a file. This stopcode occurs when a block supercluster address is not an even multiple of a block cluster address.
	Data Items:		T1 = number of blocks in group; P2 = remainder
CAS	REFSTR	HALT	Couldn't Allocate Space This stopcode is always preceded by one of the following messages, which explains why space could not be allocated: Bad block in HOME.SYS space. File structure full, suggest reducing swapping space.
CBB	COMCON	DEBUG	Command Block Bad In copying the program-to-run information from a user defined command block to the SAVGET locations, it was found that the command block was too big to fit.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
CDn	COMMON	CPU	Cache Directory Parity Error, CPU n This is a serious hardware error. Call your Field Service representative.
CDA	FILIO	DEBUG	In-Core Copy Doesn't Agree DD2MN copies pointers from a DDB to a monitor buffer to perform monitor I/O. This stopcode occurs when the cluster pointer from the structure data block does not agree with the in-core copy.
	Data Items:		T1 = in-core copy; T4 = pointer from structure data block
CDD	MSGSER	DEBUG	Can't Disconnect Device This stopcode occurs when a release call on MPX fails because the disconnect format is invalid or the device is not connected. CFP CLOCK1 JOB Can't Find PDB SETRUN sets the job-status run bit. This stopcode occurs when there is no process data block (PDB) for this job.
	Data Items:		J = job number
CGS	ONCMOD	HALT	Couldn't Get Structure Data Block DMKSTR sets up structures according to tables in ONCMOD in the DESTROY option. This stopcode occurs when the maximum number of structures is exceeded.
	Data Item:		.FSMAX = maximum number of structures allowed; T1 = structure name; P3 = address of last structure data block; P4 = system structure number for this structure.
CI7	UUOCON	DEBUG	CI7 Continuable Snapshot Dump A user requested continuable snapshot dump was requested (CONFIG program SNAPSHOT command. See TOPS-10 Operator's Guide.)
CIB	CLOCK1	CPU	CPU Interlocks Broken APRSUB services common APR interrupts. This stopcode occurs when the CPU interlock has been modified. Typically, the stopcode occurs while trying to continue a CPU that has stopped due to a fatal error.
	Data Items:		.CPNBI = CPU interlock that was modified
CIF	FHXKON	DEBUG	RC10 Isn't Fancy This stopcode occurs when the monitor attempts an RP04-only function on an RC10-controlled device.
CIO	REFSTR	DEBUG	CPF Is Odd COMCFP computes a CFP. This stopcode occurs when the number of blocks per supercluster is not a multiple of the number of blocks per cluster.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
CL0	SCANSER	STOP	Chunk Links To 0 DELCHR deletes characters from the user's input buffers when he presses the RUBOUT or DELETE key. This stopcode occurs when a TTY chunk has a backward link to 0.
	Data Items:		T2 = current chunk
CMD	UUOCON	DEBUG	Can't Move Data In reading/writing from/to the swapping space, the JOBPEK UUO was unable to transfer data between the caller's funny space and the caller's specified arguments.
CME	FILFND	DEBUG	CFP Module Error SETCFP computes a CFP. This stopcode occurs when CFP does not start at an even supercluster boundary.
	Data Items:		T2 = supercluster address relative to block 0 of unit; T3 = remainder
CMP	LOKCON	STOP	Can't Move Page In attempting to move pages out of a block of memory that is being set off-line, routine PAGFRE discovered that the free page into which we were trying to move a page was not really free.
	Data Items:		P2 = target (free) page
CMS	VMSER	DEBUG	CORE1 Must Skip SEGCON returns core allocated to a nonsharable high segment. This stopcode occurs when CORE1 gives a nonskip return when asked for core in use.
CMU	SEGCON	DEBUG	Core Messed Up CHKTAL compares CORTAL with CORTAB. This stopcode occurs when the core usage tables are inconsistent.
	Data Items:		U = free+idle+dormant; CORTAL = bit table
CNA	SCHED1	STOP	Core Not Available SWAPI swaps in a job or high segment. This stopcode occurs when an error return is given by the core-allocation routine (CORGET), which indicates that no core is available, although it has already been verified that enough core is available.
	Data Items:		J = job number
CNE	FILUUO	DEBUG	Cluster Not Even ADJALC allocates the initial blocks for a file. This stopcode occurs when the block computed as the start of a supercluster does not begin at an even supercluster address.
	Data Items:		T2 = starting block number
COW	DTESER	STOP	Called For Output On Wrong CPU DTESER was called to output data to a DTE that existed on some CPU other than the one on which DTESER was currently running.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
CRH	REFSTR	DEBUG	Cannot Read Home Blocks REFSTR refreshes a file structure. This stopcode occurs when the routine used to read home blocks (GTHOM) gives an error return.
CSA	COMCON	DEBUG	Couldn't Set Access Allowed SETAA sets the access-allowed bit for a page. This stopcode occurs when the PAGE monitor call function to set access allowed fails.
CSE	FILIO	STOP	Checksum Error CHKSUM computes a folded checksum from the first data word. This stopcode occurs when the pointer for checksumming points to a word that is not in the user's address space.
	Data Items:		RH(M) = address that caused the error; J = job number
CSP	SEGCON	JOB	Cannot Store Path STONAM is used during SAVE, GET, R, and RUN commands to search a path. This stopcode occurs when there is not enough free core to store the full path specification.
	Data Items:		T2 = number of words available
CU0	NETDEV	STOP	Can't Use Zero Dispatch This stopcode occurs when an attempt is made to use a zero dispatch in the SCNSER dispatch table. A zero dispatch is illegal.
CWN	NETSER	DEBUG	Core Allocation Went Negative GIVZWD returns monitor free core. This stopcode occurs when GIVZWD attempts to return more free core than it has.
	Data Items:		core allocation count; -1(P) = number of words returned
D36	D36COM	STOP	DECnet Non-recoverable Stopcode This is a catch-all DECnet stopcode called when DECnet encounters a nonrecoverable condition. There is an additional DECnet sub-stopcode printed out on the CTY in conjunction with the D36 stopcode. Refer to Appendix A for further information.
DBZ	FILIO	DEBUG	DEPLPC Bit Zero USETO0 does a USETO. This stopcode occurs when the last group of pointers for a file is not the last group in the RIB; there should be more file pointers.
	Data Items:		W = last block allocated; T3 = last pointer in core flag

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
DCN	D36COM	DEBUG	DECnet Recoverable Stopcode This is a stopcode-only routine called by DECnet when a recoverable condition occurs. There are many kinds of DCN stopcodes; an additional DECnet sub-stopcode is output with the CTY output. Refer to Appendix A for further information.
DCR	FILUUO	DEBUG	DELRIB CPOPJ Return CLSDL1 deletes a file. This stopcode occurs when DELRIB does not skip, even though it should always give a skip return.
DDS	FILUUO	DEBUG	DELRIB Didn't Skip BADUFD deletes a file. This stopcode occurs when DELRIB does not skip, even though it should always give a skip return.
DER	FILUUO	DEBUG	DELRIB Error Return CLSFUL is used when there is no space on a structure or all pointer slots are taken. This stopcode occurs when DELRIB does not skip, even though it should always give a skip return.
DFU	COMNET	DEBUG	Device Unrecognized DSPOBJ dispatches on the object type. This stopcode occurs when the specified device is not on the network.
	Data Items:		T4 = object type; DEVCHR(F) = device characteristics
DHA	FILIO	DEBUG	Don't Have AU Resource DWNAU releases the Alter-UFD resource. This stopcode occurs when this file attempts to return the AU resource when it does not own it.
	Data Items:		S = should have had IOSAW(200000) set
DHD	FILIO	DEBUG	Don't Have DA DWNDA returns the DISK ALLOCATION queue. This stopcode occurs when this file does not own the DA resource.
	Data Items:		PJOBN = job number
DND	FILIO	DEBUG	Drive Not Dual Ported UUOPWZ executes a CALLI for a disk channel. This stopcode occurs when the drive is not dual ported.
DNE	DTESER	STOP	Data Count Not Even DTMHED sets up a 16-bit byte pointer to point to the first word of a message. It is called with an 8-bit byte count in T2. This stopcode occurs when a caller calls this routine with an odd byte count.
	Data Items:		T2 = odd byte count that caused the stopcode
DNF	FILUUO	DEBUG	DDB Not Found CLRDDB clears a disk data block (DDB). This stopcode occurs when an attempt is made to return a DDB, but no predecessor DDB is found.
	Data Items:		F = location of DDB

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
DNH	DTESER	STOP	Driver Not Hungry DTEXD1 is entered when the driver thinks that there are no more indirect packets left, but the byte count is different.
	Data Items:		T1 = number of bytes left
DNL	QUESER	DEBUG	DEQ Not Interlocked DEQIT dequeues one Q-entry. This stopcode occurs when the job does not own the DQ interlock.
	Data Items:		J = job number
DNR	FILUUO	DEBUG	DELRIB Non-Skip Return SETEN5 is used when the RIB is set up to insert constant values and write them out. This stopcode occurs when DELRIB does not skip, even though it should always give a skip return.
DNS	FILUUO	DEBUG	DELRIB Non-Skip Return CLOSR2 is called by CLRSTS when a rename is in progress at the time of a delete. This stopcode occurs when DELRIB does not skip, even though it should always give a skip return.
DOC	FILFND	DEBUG	Don't Own CB GVCBJ returns the CB resource for a job. This stopcode occurs if GVCBJ is called, but the job does not own the CB resource.
DOM	CORE1	STOP	Don't Own MM Resource On a multiprocessor KL, the processor that wishes to manipulate pages in memory must own the memory-management resource before it can do so. This stopcode occurs when a processor tries to manipulate memory pages and does not own the memory-management resource. This resource can also be owned by a job.
	Data Items:		J = the serial number of the CPU that owns the MM resource (if owned by a CPU); J = the number of the job that is trying to manipulate pages without owning the MM resource (if owned by a job). MMUSER is the job that owns the MM resource.
DPL	COMCON	DEBUG	Directory Page Lost GTSAVP reads in a page that was output earlier. This stopcode occurs either when the page already exists or when an I/O error occurs.
DPN	COMCON	DEBUG	Directory Page Nonexistent RELDIR gets rid of the directory page and restores any pages that were written out earlier, after it creates the directory. This stopcode occurs when a PAGE monitor call with a function of 1 fails.
DQR	DZINT	DEBUG	Illegal Queue Routine The address of the routine to transfer to on dataset timeout for a DZ-11 line was zero.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
DRN	NETDEV	STOP	Data Request Went Negative MCXDAT packages characters into a data message. This stopcode occurs when the data-request count becomes negative after being decremented.
DSS	VMSER	DEBUG	DLTSP Skipped CLRSPG clears spy pages from a user's map when called by GETMIN. This stopcode occurs when CLRSPG cannot clear a spy page.
	Data Items:		T1 = user virtual page number of page
DTE	DTESER	STOP	Generic DTE stopcode This is a catch-all stopcode in DTESER. Examine the stack.
DWA	DTESER	DEBUG	DDB Went Away ZAPDTE is called to remove a DTE DDB from the DDB chain. This stopcode occurs if ZAPDTE cannot find the specified DTE.
	Data Items:		F = address of DTE DDB we were trying to remove
DWC	DTESER	DEBUG	DTECLR Called On Wrong CPU DTECLR was called to clear a DTE, but the DTE was not on the CPU on which DTECLR was executing.
EFI	DTESER	STOP	Eleven Function Illegal DTEQUE places an entry into a DTE's Tops-11 queue. This stopcode occurs when the Tops-11 that called this routine is either out of range or illegal for the direction of transfer.
	Data Items:		P2 = function code that caused the error
EPO	ERRCON	DEBUG	Executive PDL Overflow APRPDL handles PI 7 clock interrupts with the pushdown list (PDL) or bit set. This stopcode occurs when there is a PDL overflow in the exec.
	Data Items:		S = APR error condition
ERB	REFSTR	DEBUG	Error Reading BAT Block REDBAT reads in BAT blocks and returns a new unit pointer. This stopcode occurs when an error is encountered while reading.
	Data Items:		U = address of current unit
ERD	ONCMOD	DEBUG	Error Refreshing Disk WUNSTR refreshes a structure. This stopcode occurs when an I/O error is while refreshing.
ERF	TAPSER	STOP	Error Recovery Procedure Fouled Up ERPINT is used when an interrupt is received while error recovery is in progress. This stopcode occurs when a pointer that should be pointing at an I/O request block is pointing elsewhere.
	Data Items:		T1 = bad pointer; TKBERB(W) = good pointer

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
ERH	REFSTR	DEBUG	Error Reading HOME.SYS This stopcode occurs when the refresher cannot read the home blocks. (See stopcode CRH.)
ERM	ONCMOD	DEBUG	Error Reading MFD SPTSSB creates the SPT table and the swapping SAT table for a disk unit. This stopcode occurs when an I/O error occurs while reading the MFD read-in block.
	Data Items:		P1 = address of first word of MFD RIB (RH); T2 = logical block number to read; U = unit data block address; F = file data block address; S = (RH) standard error bits
ERP	REFSTR	HALT	Extraneous Retrieval Pointer HOMRBS stores a retrieval pointer in the HOME.SYS read-in block. This stopcode occurs when the byte pointer is confused.
	Data Items:		T1 = current byte pointer
ERS	ONCMOD	DEBUG	Error Reading SAT FILMAN finds and sets up all of the structures on the system. This stopcode occurs when I/O takes place while reading the SAT.SYS read-in block for a structure.
	Data Items:		S = error bits
ESS	FILFND	JOB	Empty System Search List SLXESS is a stopcode-only routine and is called throughout FILFND. Examine the stack for the location.
EUE	ERRCON	DEBUG	Executive UUO Error EMUERR warns about monitor calls that are no longer implemented. This stopcode occurs when the monitor tries to execute one of these de-implemented monitor calls.
EWB	REFSTR	DEBUG	Error Writing Block BLKWRT writes out a block. This stopcode occurs when the subroutine to do the actual writing of the block, OWNWRT, gives an error return that indicates an I/O error.
	Data Items:		DEVMBF(F) = IOWD; T2 = logical block number; U = address of unit
EWH	REFSTR	DEBUG	Error Writing Home Block HOMUPD updates the home blocks. This stopcode occurs when the subroutine used to do the physical I/O (WRTRUN) gives an error return.
	Data Items:		T2 = list of items to be written; S = standard error bits
EWR	ONCMOD	DEBUG	Error While Refreshing RFRES2 refreshes a structure during the DESTROY option. This stopcode occurs when an I/O error is encountered during the refresh.
	Data Items:		S = error bits

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
FAD	FILUUO	DEBUG	File Already Dormant CLRSTG is used during a CLOSE monitor call to do general clean-up tasks. This stopcode occurs when the access-table entry for this file is mistakenly marked dormant.
	Data Items:		ACCDOR(T1) = access-table entry for this file
FDP	FILIO	DEBUG	Fixed-Head Device Position FREINT handles unsolicited interrupt from a device. This stopcode occurs when a position-done interrupt occurs for a fixed-head device. This may indicate a hardware problem.
	Data Items:		KONPOS(J) = unit positioning flag
FEM	ERRCON	HALT	Fatal Error In Monitor PARHALT halts a CPU when there is a serious error in the monitor.
FFU	NETSER	STOP	F Fouled Up NETHIB puts a network job in the HIBER state. This stopcode occurs when NETHIB is called with F = 0.
FIP	VMSER	DEBUG	Free-Page In Use This stopcode occurs when the monitor finds an allocated page in its list of unallocated pages.
FLE	SCNSER	STOP	Free List Empty GETCHK fetches chunks from the SCNSER free-chunk chain. This stopcode occurs when the pointer to the first chunk (TTFTAK) is zero.
FNG	DTESER	STOP	Function No Good EATMSG throws away messages from the -11 for unknown devices. This stopcode occurs when the function sent by the -11 is illegal.
	Data Items:		P2 = illegal function code
FON	VMSER	STOP	Funny Address Overlaps Next GVFWDS returns words acquired by GTFWDC or GTFWDU. This stopcode occurs when the size of the current chunk plus the the address of the current chunk overlaps the first word of the next chunk.
	Data Items:		T1 = words to return; T2 = address
FOP	VMSER	STOP	Funny Address Overlaps Previous GVFWDS returns words acquired by GTFWDS or GTFWDU. This stopcode occurs when the first word of the funny address is in the previous chunk.
	Data Items:		T1 = words to return; T2 = address

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
FPE	VMSER	DEBUG	Funny Page Must Exist GVFWDS returns words acquired by GTFWDS or GTFWDU. This stopcode occurs when the monitor tries to get a page map entry, but no funny page exists from which to get the page map entry.
	Data Items:		T1 = words to return; T2 = address
FPF	LOKCON	STOP	PAGFRE Free Page Not Free PAGFRE checks to see if the target page is on the free-core list and, if so, exchanges it with the current page. This stopcode occurs when a page on the free-core list is not marked as being free (Bit 0 = 1).
	Data Items:		T1 = current page location of job; T2 = page on the free-core list that was not marked as free (this is the page that caused the stopcode); P2 = target page
FPI	CORE1	STOP	GTPAGS Free Page In Use This stopcode occurs when the monitor tries to get a page from the free-core list, but reaches the end of the free-core list before finding a free page.
	Data Items:		T1 = first page added to or taken from the free-core list; T2 = negative number of pages to add or take from the free-core list
FPN	LOKCON	STOP	SETMFL Free Page Not Found SETMFL moves monitor pages when its current pages are being set off line. This stopcode occurs when the target page is not found on the free-core list.
	Data Items:		T1 = current page (being set off line); P2 = target page (this is the page that cause the stopcode
HBE	REFSTR	DEBUG	Home Block Read Error HOMUPD updates the home blocks. This stopcode occurs when the subroutine to read in the current home blocks from disk (GTHOM) gives an error return, which indicates an I/O error.
	Data Items:		S = standard error bits
HIF	FILIO	DEBUG	Hole In File USETOO is used to do a USETO. This stopcode occurs when the last block of the file exists, but some preceding block does not.
HNF	ONCE	HALT	High Seg Not Found ONCE could not find the monitor.
HWU	FILIO	JOB	Hardware Wrong Unit POSERC is used during disk error recovery. This stopcode occurs when the wrong unit on a disk controller interrupts. This is a hardware problem.
	Data Items:		P2 = error bits; T1 = retry count

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
I1S	DTESER	DEBUG	Illegal Tops-11 DONE State for QP2 DTESER received a Tops-11 DONE interrupt, but the state specified by the -11 was not legal for queued protocol 2.
IBA	DTESER	DEBUG	Input Buffer Already Allocated When DTESER went to start up I/O on a DTE, it found a buffer already allocated in the ETD block.
IBI	CLOCK1	JOB	Intercept Block Illegal ESTOP stops the user on an error and flags it as an error stop. This stopcode occurs when the user-defined intercept block is illegal for some reason.
IBZ	FILIO	JOB	I/O To Block Zero UUOPWR performs CALLIs for a disk channel. This stopcode occurs when Block 0 is requested, but the file is not HOME.SYS[1,4] or the drive is not an RP04. An RP04 reads block 0 to check for format errors.
	Data Items:		T1 = PPN; T2 = file name
ICL	UUOCON	JOB	Illegal Channel Number JDAADR returns the address of a channel in USRJDA or the extended channel table in T1. This stopcode occurs when the channel number is greater than the maximum number allowed.
	Data Items:		T2 = channel number; HIGHXC# = maximum
ICN	SEGCON	DEBUG	In-Core Count Negative DECCNT decrements the high-segment in-core count for a job that has a very high segment, which must be in core. This stopcode occurs when the count of the number of jobs in core sharing a high segment becomes negative.
	Data Items:		J = job number; JBTSTS(J) = in-core count, which should be zero now
IDC	ONCMOD	HALT	Impossible Drum Condition DRMHLT checks for a unit on line and write protected. This stopcode occurs because a drum can never give an on-line condition.
IDS	DTESER	DEBUG	Illegal Tops-10 DONE State DTESER received a Tops-10 DONE interrupt with an illegal state specified by the -11.
IEZ	KLSER	DEBUG	IOWD Equals Zero After paging a job, the monitor attempted to convert virtual IOWDs to absolute IOWDs. This stopcode occurs when this routine is called with an IOWD of zero.
	Data Items:		T2 = IOWD; P3 = location of channel data block; P4 = frame count,,characters/word if DX10 channel

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
IFI	TAPSER	STOP	Illegal Function At Interrupt TAPIFI is a general interrupt error halt. In one case, for example, the monitor found an illegal function in an I/O request block while at interrupt level. Examine the stack for the specific error address.
IIP	FILIO	STOP	I/O In Progress - Error MONIO reads a block or series of blocks from the disk. This stopcode occurs when the monitor attempts to start I/O for a DDB that already has I/O active.
	Data Items:		S = status bits; T1 = IOWD for data; T2 = block number
IKF	DTESER	DEBUG	Illegal Kontroller Function DTESER was requested to put the DTE into maintenance mode, which is illegal.
IME	KLSER	JOB	Illegal Memory Reference From Executive This stopcode occurs when there is a page fault while in exec mode that is not an address break.
	Data Items:		.CPAPC = page fault PC; .CPPFW = page fault word
IOP	COMMON	CPU	I/O Page Failure There is a serious hardware failure. Call your Field Service representative.
IPA	DTESER	STOP	Illegal Post Address DTEXDI handles a Tops-10 DONE interrupt on an indirect transfer. This stopcode occurs when this routine is called before a Tops-10 post address has been set up. The post address should have been set up by the driver at the time that the direct portion of that message was received.
	Data Items:		T1 = To-10 DTE state; ED.DTN(F) = DTE number; F = ETD address
IPC	KLSER	CPU	Illegal Page Failure Trap Code SEILM processes page failure traps. This stopcode occurs when the trap code returned by the pager after getting a page fail trap is not in the range 0 through 23 or 25.
	Data Items:		T1 = page fail code; .CPTCX = page trap context; .CPTPI = PI state
IPF	VMSER	DEBUG	In-Use Page Free SCNPT scans the page table. This stopcode occurs when SCNPT finds a page on the free-core list that is listed in the page table as allocated.
IPM	VMSER	DEBUG	Illegal Pointer In MEMTAB This stopcode occurs when the monitor finds an inconsistency in the swapping data base.
IPN	VMSER	DEBUG	IPCF Page Nonexistent GVIPCP returns IPCF pages to the free-core list. This stopcode occurs when GVIPCP swaps out IPCF pages that are not in the swap list.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
IPU	LOKCON	STOP	IPCF Page Unowned In attempting to swap out an IPCF page, LOKCON was unable to find a job that contained the IPCF page in its queue.
ITM	DTESER	DEBUG	Illegal Tops-10 Transfer Mode in QP2 DTESER received a Tops-10 interrupt in queued protocol 2 which specified a mode other than byte mode.
ITS	DTESER	DEBUG	Illegal Tops-10 Transfer State DTESER received a Tops-10 interrupt which specified an illegal state.
IUN	FILUUO	DEBUG	Invalid Unit Number ERRFIN finishes a CLOSE when an error occurred. This stopcode occurs when the unit number for the UFD is illegal.
	Data Items:		UN1PTR = pointer to number in structure of the unit; T1 = unit of UFD
JAC	UUOCON	DEBUG	Job Data Area Clobbered IOALL does I/O for all devices assigned to a job. This stopcode occurs when the highest channel number in use is greater than 17.
	Data Items:		T2 = highest channel in use
JDJ	ONCMOD	DEBUG	JFFO Didn't Jump CMPLOG computes the SIXBIT logical unit number within a structure. This stopcode occurs after a call to subroutine MSKUNI, which sets up a search mask for a unit name, and it returns no mask for the unit name given.
	Data Items:		P2 = unit number; STRNAM(P2) = unit name; T2 = should have been the complement of the search mask
JIT	SYSINI	HALT	Job In Transit LOKINI initializes LOKCON on location 140 restarts, unlocks all jobs, and frees up the core they occupy. This stopcode occurs when a job is locked and is being moved.
	Data Items:		J = job number
JNC	FILIO	DEBUG	Job Not In Core ADRINT checks that a job is in core, possibly at interrupt level. This stopcode occurs when the job is not in core.
	Data Items:		T1 = job number
JNE	CLOCK1	STOP	JBTADR Not Equal To CORTAL In cross-checking JBTADR and CORTAL a mismatch was found.
	Data Items:		P3 = amount of free-core specified by the sum of JBTADR entries
KAF	COMMON	CPU	Keep-Alive Failure This routine/stopcode is executed because the console front end detected that the KL did not update the keep-alive counter. This stopcode occurs when the front end executes a JRST 71.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
KCP	RNXKON	DEBUG	KDB Command In Progress CMDWAT waits for a command to complete and calls RNXINR to process it before starting a new command.
KDS	DPXKON	DEBUG	KONEC2 Didn't Skip POSINT handles positioning interrupts. This stopcode occurs when the subroutine KONEC2 does not skip, even though it should always give a skip return.
KID	SYSINI	HALT	Kontroller Is Down DSKINI, which is called on location 140 and 143 starts and restarts, initializes the disk. This stopcode occurs when a controller goes off-line during disk initialization.
	Data Items:		J = kontroller
KNF	XTCSER	STOP	Kontroller Not Free XTCSER received a remote interrupt request, the the kontroller was not free.
KNM	TX1KON	DEBUG	Kontroller Not Mapped In attempt to uncache a DX10 KDB we found the KDB was not in the EPMP.
KSW	TAPSER	DEBUG	Kontroller Status Wrong TAPSIO is used when the upper level wants to start I/O. This stopcode occurs when the tape-controller status is wrong.
	Data Items:		TKBSTS(W) = status
LN1	ERRCON	STOP	Line Not Found EXCALP prints a monitor call PC message for a job. This stopcode occurs when no terminals a log line can be found for the job that is causing the error.
LND	FILUUO	DEBUG	Logical Name Not Found LNMSTP consists of only the stopcode and its recovery. LNMSTP is called when the monitor could not set up the definition of LIB that was present before an ENTER UUO that could not find a file.
LNF	QUESER	DEBUG	Lock Not Found REDTB fills a user table with data from the LOCK-associated table. This stopcode occurs when the system cannot find the LOCK block.
LNP	FILIO	DEBUG	Last Pointer Not A Pointer OUTGRP allocates more space for an output file. This stopcode occurs when an allocation is made, but a RIB error occurred; or when the monitor tried to deallocate the space, but the RIB pointer was invalid.
	Data Items:		T2 = pointer
LNS	SCNSER	STOP	Line Not Set Up TSETBI clears the input and output buffers for a line. This stopcode occurs when this routine is called before the line is set up.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
LNT	ERRCON	STOP	Line Not There HALTI prints the "Halt at ..." message and stops the job. This stopcode occurs when there is no controlling terminal line associated with the job.
LPU	FILUUO	JOB	Last Pointer Unit-Change ALLPOB writes the redundant RIB in the last block of the RIB. This stopcode occurs when the RIB pointer is decoded as a unit-change pointer.
	Data Items:		T2 = pointer
MCM	METCON	DEBUG	Meter Channel Data Block Missing RELCHN releases a channel. This stopcode occurs when an attempt is made to release a meter channel data block (MCDB) that is not there.
	Data Items:		T2 = predecessor MCDB (if any)
MCN	FILFND	DEBUG	Mount Count Negative SLSR6 documents the mount count when the search list is modified. This stopcode occurs when the mount count for a structure becomes negative.
	Data Items:		STRMNT(T3) = mount count
MDM	DTESER	STOP	Master DTE Missing DTEINI is called to initialize all the DTEs on a given CPU. This stopcode occurs after all of the DTEs have been initialized, when none is found in privilege mode.
MIW	ONCE	STOP	Memory Interleaving Is Wrong BYPSYM finds the top of core and moves the symbol table up. This stopcode occurs when there is a memory interleaving error in that some words within a page exist and some do not. The operator must correct the problem.
MIZ	VMSER	DEBUG	MEMTAB Is Zero This stopcode occurs when the monitor finds an inconsistency in the swapping data base.
MMR	LOKCON	STOP	Moving Monitor Page Not Required SETMFL sets memory pages off line. This stopcode occurs when the memory location labeled MOFLPG indicates that monitor pages must be moved, but none of the page's PAGTAB entries has the monitor code bit set.
	Data Items:		MOFLPG = number of monitor pages that were to be moved; LOKREL = number of pages, first page number to set off line
MNA	FILIO	JOB	Monitor Buffer Not Available GTMNBF gets a monitor buffer. This stopcode occurs when there is no buffer space available.
	Data Items:		T2 = number of words requested

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
MNM	SYSINI	STOP	Monitor In Nonexistent Memory KIINI initializes a KI10, KL10, or KS10. This stopcode occurs when a page is found to be nonexistent and the page is not free. Therefore, the monitor already has the page in question.
	Data Items:		T3 = page number
MNR	ERRCON	HALT	Master -11 Not Running DIE recovers/reloads after an internal system error. This stopcode occurs when the -11 with a DTE in privilege mode is not running.
MPN	LOKCON	STOP	Monitor Page Not Found SETMFL sets monitor pages off-line. This stopcode occurs when the source page cannot be found in the monitor.
	Data Items:		P3 = monitor source page
MXM	MSGSER	DEBUG	MPX DDB Missing This stopcode occurs when a pointer in the DDB chain does not point to a multiplexd DDB.
N4C	CPNSER	JOB	Not 4 Cached Pages FIXOTB fixes up OUCHTB when turning off the cache for some page so that OUCHE references through four cached pages. This stopcode occurs when four cached pages cannot be found.
	Data Items:		P3 = page for which cache is being turned off
NAP	FILUUO	JOB	Not Address Pointer UFDNXT initializes the next block for the directory. This stopcode occurs when the new pointer is decoded as other than an address pointer.
	Data Items:		P1 = location is monitor buffer; T2 = bad pointer
NCA	CLOCK1	STOP	No Core Assigned NULADR restores the software state and then the hardware state of the new job to be run. This stopcode occurs when the job to be run has no core assigned to it.
	Data Items:		J = job number
NCC	LOKCON	STOP	Not Enough Contiguous Free Core Certain types of monitor pages must be kept contiguous. This stopcode occurs when LOKCON does not have enough contiguous space to lock jobs after memory has been set off-line.
NCE	UUOCON	DEBUG	No Command Slot Available This stopcode occurs when attempting to insert a user-defined command. The condition should have been caught earlier.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
NCM	IPCSER	JOB	No Core For Message SETQSR sets up IPCF packets to send to QUASAR. This stopcode occurs when no core is available to build the message.
NDJ	SCNSER	DEBUG	No DDB For Job TTYFND finds a terminal number for the job in AC J. This stopcode occurs when no device data block can be found for this job's terminal.
	Data Items:		J = job number
NDL	COMNET	STOP	No DECnet Loaded This stopcode occurs if any DECnet-only routines are called, but DECnet is not assembled into the monitor.
NDP	CLOCK1	JOB	Not DDB Pointer WSYNC waits until the current buffer activity is complete. This stopcode occurs when this routine is called with other than a DDB pointer in F.
	Data Items:		F = the supposed DDB pointer
NDS	CLOCK1	STOP	Null Job Did SAVGET MONSTR sets up ACs for a monitor job that starts at monitor call level. This stopcode occurs when the job number is 0.
NEM	LP2SER	JOB	No Exec Virtual Memory DVLRAM loads the RAM or VFU with data from the user. This stopcode occurs when DVLRAM tries to map the user virtual address into exec virtual memory, but there is none.
	Data Items:		F = DDB; T1 = function
NER	FILUUO	DEBUG	No Extended RIB CLSO2A looks for the last written block in the next RIB. This stopcode occurs when the pointer for the last block of a file is not in the RIB, and there is no extended RIB; the pointer is lost.
NEV	UUOCON	STOP	No Executive Virtual Memory NEWBUF sets up a byte pointer and item count for I/O. This stopcode occurs when the DDB does not have executive virtual memory.
	Data Items:		T1 = input buffer header address
NFB	FEDSER	STOP	No Front-End Device Block FDIGET gets the front-end device (FED) address in F. This stopcode occurs when no device data block is found for this front end.
	Data Items:		T1 = FED unit number (also on this stack)

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
NFC	DTESER	STOP	No Free Core DTERNG handles a To-10 doorbell. This stopcode occurs when the monitor tries to allocate a To-10 buffer and none is available.
	Data Items:		T2 = largest amount of core available (36-bit words); ED.BSZ = amount of 36-bit words needed (ED.BSZ is located in EDTBUF(F) in Bits 1 through 17); F = ETD address
NFD	RPXKON	DEBUG	No Front-End Drive DAVIN1 starts an operation on a drive that is busy on the other port. This stopcode occurs when DAVIN1 cannot find the drive number.
NFS	VMSER	DEBUG	No First Slot This stopcode occurs when, at the start of a fragment, the first physical page of the fragment is not found in the page map.
NFU	SYSINI	DEBUG	No First Unit DSKINI initializes a disk on a location 140 start or restart. This stopcode occurs when the first unit in the system search list cannot be found.
	Data Items:		SYSUNI = should have been the pointer to the first unit. (LH)
NIF	RNXKON	DEBUG	RNXKON Isn't Fancy This stopcode occurs if the monitor tried to unload an RP20 or read/write 10/11 compatability mode on an RP20.
NIS	DTESER	STOP	DTE Not In Indirect State DTEXDI is entered on an indirect message transfer. This stopcode occurs when an indirect transfer is received, but the monitor is not expecting one.
	Data Items:		T1 = Tops-10 DTE state word (the state word breaks down as follows); T1 = 0 (Tops-10 idle); T1 = 1 (waiting for Tops-10 DONE for a direct transfer); T1 = 2 (waiting for Tops-10 doorbell, which indicates that -11 has set up for a Tops-10 indirect transfer); T1 = 3 (waiting for Tops-10 DONE on indirect transfer)
NIV	TAPUUO	STOP	Null Interrupt Vector TPMDON is called by TAPSER when I/O is complete to dispatch to the correct routine for processing. This stopcode occurs when the routine address for this function is null.
	Data Items:		P1 = function; (T1) = should be dispatch address
NJT	ERRCON	STOP	Null Job Has TTY EXCALP prints the PC of a monitor call that caused an error. This stopcode occurs when NULJOB has control of the terminal.
NLB	FILUUO	JOB	No Last Block This stopcode occurs during UFD compression if we cannot find the pointer for the last block of the UFD.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
NMC	ONCMOD	HALT	No More Core OK22B sets up controllers during system startup. This stopcode occurs when the routine used to create a device data block for ONCE-only I/O (SETDDO) gives an error return, which indicates that no core is available.
	Data Items:		T2 = size of chunk needed
NMU	REFSTR	DEBUG	No More Units HOMZR2 writes zeros in unused blocks in HOME.SYS. This stopcode occurs when the count of units is greater than the number that can be accessed.
	Data Items:		T1 = cluster count; T2 = next retrieval pointer
NNF	FILUUO	DEBUG	NMB Not Found GETNMB gets the location of the name block (NMB) from the DDB. This stopcode occurs when there is no access table entry for the user channel.
NNR	FILUUO	JOB	No Next RIB DELGRP returns blocks on an update ENTER. This stopcode occurs when the last block pointer cannot be found in the current RIB, so an attempt is made to scan the next RIB, but there is no other RIB.
NNS	CORE1	DEBUG	Not In Non-Zero Section In attempting to clear/set bits in a bit table, SETR was called requesting the usage of a relative AOBJN pointer. This is only relevant for bit tables in nonzero sections and the code was not executing in a nonzero section.
NNU	ONCMOD	DEBUG	Not A New Unit FILMAN finds and sets up all structures on the system. This stopcode occurs when the monitor expects to find a new unit pointer as the next item read, but does not.
	Data Items:		T1 = AOBJN pointer for scanning retrieval information; T2 = supposed new unit pointer (Bit 18 must be set to be a new unit pointer); T3 = logical unit number in this structure; P2 = address of structure
NOB	COMNET	DEBUG	"Nobody" Got Obsolete Buffer This stopcode occurs if someone tries to pass a message to a Front End which is not owned by anyone. This is probably caused by the line driver trying to return stale data to a previous line user and getting confused.
NOT	SCNSER	DEBUG	No Operator TTY TTYERP finds a terminal device data block for a monitor error message. This stopcode occurs when the monitor cannot find a DDB for a job, so it tries to find the operator's line number from ONCE and still cannot find it.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
NPD	FILIO	DEBUG	No Pointers In DDB EXTRIB creates an extended RIB. This stopcode occurs when an extended RIB is needed, but no pointers exist in the DDB.
	Data Items:		DEVMBF(F) = IOWD to monitor buffer
NPF	LOKCON	STOP	Next Page Free PAGFND finds the target page in this segment's map because it was not on the free-core list. This stopcode occurs when a page in this segment is marked as free.
	Data Items:		T1 = current page; T2 = PAGTAB entry for the next page (this is the page that caused the error)
NPI	KLSER	HALT	Not Parity Instruction SWPTRP sweeps a memory for parity. This stopcode occurs when a page fail trap occurs that is not caused by the sweep routine and, in fact, is caused by the instruction at label CPLMPI.
	Data Items:		T1 = PC of the instruction that caused the trap (also in .UPMP + .LMPFP); .UPMP + .LMPFW = page fail word
NPJ	DATMAN	DEBUG	No PDB For Job This stopcode occurs when no process data block can be found for this job.
	Data Items:		J = job number
NPN	ERRCON	STOP	Nonexistent Page Not Free CPINXF fixes up the core-allocation tables after pages have been marked out in NXMTAB because of parity errors or NXMs. This stopcode occurs when the page being marked off-line is in use not free).
	Data Items:		PAGTAB(T1) = page entry of page causing the stopcode
NPU	ERRCON	STOP	Null Pushdown List Underflow This stopcode occurs when there are more POPs on the null pushdown list than matching PUSHs.
NRF	VMSER	DEBUG	SWPLST Not Really Fragmented This stopcode occurs when there is a pointer to a fragmented SWPLST entry, but the entry is not really fragmented.
NRM	FILUUO	JOB	Next RIB Missing RENRIB is used when allocation or deallocation is done and set up to do the close. This stopcode occurs when the last block pointer is not found in the current RIB and there are no other RIBs.
NRS	ONCMOD	DEBUG	No RIB In SAT FILMAN finds and sets up all structures on the system. This stopcode occurs when the monitor expects to see a read-in block, but does not.
	Data Items:		T1 = the supposed RIB

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
NSE	VMSER	DEBUG	No SWPLST Entry This stopcode occurs when the monitor attempts to compute the unit and block numbers corresponding to a SWPLST entry, but the pointer to SWPLST points to a zero word.
NSR	REFSTR	HALT	No Second RIB FILSET creates a file of contiguous disk space and write zeros in data blocks. This stopcode occurs when the subroutine used to scan a block of retrieval pointers to find the group pointer, SCNPTR, gives an error return because it cannot find it.
	Data Items:		P1 = pointer to cluster count; T2 = number of clusters in this pointer
NSS	REFSTR	DEBUG	No Space For SAT ENDSAT allocates blocks in the HOME.SYS file for SATs. This stopcode occurs when there are no free clusters left.
NSU	FILIO	DEBUG	No Such Unit USTRIB reads in the RIB and scans it from the beginning if the pointers do not encompass the desired block. This stopcode occurs when the subroutine that finds a unit (NEWUN) gives an error return indicating that the desired unit is greater than the last unit in the structure.
	Data Items:		S = error bits; IOBKTL SET
NTE	SCHED1	STOP	Not Processor Queue Error QLNKZ is used in the requeuing of a job. This stopcode occurs when this routine is called for a job that is not in a processor queue.
	Data Items:		J = job number
NUB	FILFND	JOB	No UFB Block STRDN4 creates an access table entry. This stopcode occurs when there is no UFD for a file even though the file exists.
	Data Items:		P2 = Structure data block (LH)
NUE	FILUUO	JOB	No UFB Error SETUFR sets the RIBUFD word in the RIB. This stopcode occurs when an error return is given by the subroutine used to compute the RIBUF word, but actually there is no UFD or SFD, so there can be no UFB error.
	Data Items:		DEVUFB(F) = pointer to UFD; DEVSFD(F) = pointer to SFD, if any
NUI	XTCSER	DEBUG	Non-existent Unit Interrupting XTCSER could not find the UDB for a unit it received an interrupt request from.
NUN	FILUUO	DEBUG	NMB Use-count Negative The name-block use count was decremented to -1.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
NUP	FILUUO	DEBUG	No Unit-Change Pointer LSTUNI finds the last unit-change pointer in a RIB. This stopcode occurs when no change pointer is found or when the pointer is not a unit-change pointer.
	Data Items:		T2 = pointer
NWA	NETDEV	STOP	No-one Wrote Anything TWRPCB writes back the count field and updates the pointer in the PCB. It also removes garbage from the stack.
	Data Items:		T1 = minimum number of bytes required (less count field)
NXS	VMSER	DEBUG	Non-existent Section DNZSPG is called to return a nonzero section page to free core. This stopcode occurs if the section of the specified page does not exist.
NXU	FILIO	DEBUG	Non-existent Unit WRTRIB writes a RIB. This stopcode occurs when a unit-change pointer points to a unit that does not exist in the structure.
	Data Items:		S = error bits; U = 0 if not in any F/S
O1F	VMSER	DEBUG	Only 1 Fragment This stopcode occurs when swapping space is fragmented, but there is only one entry in the fragment table.
OBA	DTESER	DEBUG	Output Buffer Already Allocated When DTESER was called to do I/O on a DTE, it found a buffer had already been allocated for it in the ETD block.
OMR		JOB	Out Of Mapping Registers MAPIO sets up the UNIBUS adapter mapping registers for a given IOWD following the paging of a job. This stopcode occurs when an attempt is made to point to the next mapping register, but there is none.
	Data Items:		P1 = address of next paging register to be used; P2 = address of first paging register used
ONC	FILUUO	DEBUG	Odd-Numbered Cluster UPDGIV deallocates or truncates blocks from a file. This stopcode occurs when the number of blocks allocated to a file is not an even multiple of the number of clusters allocated.
	Data Items:		T1 = number of clusters; T2 = remainder
OVA	ONCE	HALT	Out Of Virtual Address Space ONCMAP selects the physical and virtual address for space in the high segment. This stopcode occurs when the number of virtual pages is greater than 256K.
	Data Items:		R1 = virtual page number
P2L	VMSER	STOP	Page Too Low This stopcode occurs when an address in the user page map is too low; that is, the address is in the monitor space.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
PAO	COMCON	STOPE	Page Already Out PAGRE creates a directory page. This stopcode occurs when an attempt is made to page out a page that is already out. The page is being paged out because the job's physical limit has been exceeded.
	Data Items:		J = job number
PBO	NETSER	STOP	PCB Buffer Overflow On input done, we found that the data input overflowed the PCB's buffer allocation.
PCI	DTESER	STOP	Previously Checked Function Code Illegal ISALL checks to see if the function code is legal for all devices. This stopcode occurs when: (1) the function code is greater than MAXFNC (currently 33); (2) the function code is illegal for the direction of the transfer; or (3) if in 16-bit mode, the function is illegal for that mode.
	Data Items:		T1 = function code
PCN	IPCSER	DEBUG	Packet Count Negative UIPCFR is used on an IPC receive monitor call. This stopcode occurs when the count of unreceived IPCF packets goes negative.
	Data Items:		P1 = PID
PDA	FILIO	DEBUG	Pointers With Different Addresses DD2MN copies pointers from the DDB to the monitor buffer during monitor mode I/O. This stopcode occurs when the RIB pointers and those now in the monitor buffer differ.
	Data Items:		T3 = XORed RIB and monitor buffer pointers; T4 = cluster pointer
PEW	VMSER	DEBUG	PAGTAB Entry Wrong PHYCRZ allocates physical core. This stopcode occurs when a page that is in the free-core list is found while scanning the pages allocated to a segment.
	Data Items:		T1 = number of pages; T2 = starting virtual page number
PEZ	CORE1	STOP	PAGPTR Equals Zero GTPAGS adds to or takes pages from the free-core list. This stopcode occurs when the location PAGPTR, which points to the first free page, is zero.
PFC	VMSER	STOP	Page On Free Core List SETHMT prepared for high-segment swap. This stopcode occurs when the monitor finds a page that is in the free-core list while scanning pages allocated to a segment.
	Data Items:		T1 = first disk address; T2 = first page number; T3 = number of pages

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
PFL	VMSER	DEBUG	Piece On Free List GVFWDS returns words acquired by GTFWDC or GTFWDU. This stopcode occurs when an attempt is made to return a chunk of funny space that is already on the free list.
PFN	KLSER	CPU	Page Fault In Null Job A page fault occured while the null job was running.
PFR	VMSER	DEBUG	Piece Out Of Free Range GVFWDS returns words acquired by GTFWDC or GTFWDU. This stopcode occurs when GVFWDS is called with an address that is not in funny space.
PFS	LOKCON	STOP	Page Is Free In Segment PAGFRE checks to see if the target page is on the free-core list and, if so, exchanges it with the current page. This stopcode occurs when the current page is marked as being free in PAGTAB.
	Data Items:		T1 = current page; T2 = PAGTAB entry of page causing the stopcode
PGL	COMCON	STOP	Pages Got Lost PAGFRE creates a directory page. This stopcode occurs when the page cannot be paged out.
PIE	ERRCON	CPU	Priority Interrupt Error This stopcode occurs when a device interrupts to the wrong location. A jump occurred to an even address between 42 and 66 (octal).
PIF	VMSER	DEBUG	Page Is Free This stopcode occurs when the monitor finds a page that is in the free-core list while it is scanning pages allocated to a segment.
PIN	VMSER	DEBUG	Page In Working Set WSBIT gets bit and index for WSBTB and AABTAB. This stopcode occurs when the monitor finds a page in the working set that has been verified as not in the working set.
PIP	KLSER	STOP	PI In Progress This stopcode occurs when priority interrupt is in progress while handling a stack overflow.
	Data Items:		.CPAPC = error PC

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
PIW	VMSER	DEBUG	Page Isn't In Working Set PAGOMT sets up MEMTAB for paging out. This stopcode occurs when the monitor decides that a page must be in the working set, but it is not.
PJ0	CLOCK1	DEBUG	Requeue Job 0 REQUE requeues a job to run. This stopcode occurs when an attempt is made to requeue job 0 (the null job) to run.
PLP	FILIO	DEBUG	Past Last Pointer USETO1 is used to do a USETO when the requested block is higher than the highest allocated block. This stopcode occurs when the SCNPTR routine, which scans pointers, cannot find a block that should be in the file.
	Data Items:		P1 = top block to allocate; P2 = first block to allocate; DEVLPC(F) = RIB pointer
PMU	CORE1	STOP	PAGTAB Messed Up This stopcode occurs when a zero is encountered as the link to the next page in the segment while setting up the user's page-map page to reflect the location of the pages in physical memory.
	Data Items:		T1 = byte pointer to the map; T2 = page attributes; T4 = number of pages -1 left in this segment
PNE	FILIO	DEBUG	Pointers Not Equal PTRTST reads the pointers into core, compares the old pointers in the RIB with the new pointers in the DDB, and rewrites the RIB if they differ. This stopcode occurs when an error is found in the cluster pointer after the pointers in the RIB have been updated.
	Data Items:		(T1) = pointer in the monitor buffer; T3 = XORed RIB and monitor buffer pointers; T4 = cluster pointer
PNM	FILFND	DEBUG	Physical Name Mismatch DSKCHK checks to see whether C(T1) = 'DSK,''DS' or 'D'. This stopcode occurs when the physical name of the device is lost after it has been determined that a name refers to a disk unit.
PNP	LOKCON	STOP	Page Not Present PAGMOV finds the target page on the free-core list or within the current segment and exchanges it with the source page. This stopcode occurs when the source page cannot be found in the current segment.
	Data Items:		T2 = successor to current page

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
PNW	VMSER	DEBUG	Page Not In Working Set DLTMPG returns a funny page to the free-core list.
POR	SEGCON	STOP	Process Out Of Range COMIT is used to right half of J with JBTSGN(T1). This stopcode occurs when the job number is out of range.
	Data Items:		J = job number
PQE	FILIO	DEBUG	Positioning Queue Empty UNIPOS picks a file on a unit and starts positioning for that file. This stopcode occurs when a disk unit was in the position-wait state, but there are no files in its queue to be positioned.
	Data Items:		U = location of unit data block
PRF	KLSER	CPU	Page Refill Failure This stopcode occurs when a page-fail code of 22 is returned by the pager.
	Data Items:		T1 = page-fail code; .CPTPI = PI state; .CPTCX = trap context
PSF	CORE1	STOP	Page In Segment Free This stopcode occurs when a page is found that is marked in PAGTAB as being free while scanning a job's pages looking for page n or the last page.
	Data Items:		T1 = current page within this segment; T2 = number of pages left to scan; T3 = PAGTAB entry for next page in segment, that is, PAGTAB(T1).
PTL	DTESER	STOP	Packet Too Large DTERNG is entered on a Tops-10 doorbell interrupt. This stopcode occurs when the monitor allocates a Tops-10 buffer of the size that the -11 said the message was, but the message is actually larger than the buffer for this DTE.
	Data Items:		T1 = queue size in 8-bit bytes that the -11 says the message is; T3 = size of the To-10 buffer in 8-bit bytes
PTP	KLSER	HALT	Page Table Parity A page fail code of 25 was received from the pager. The operation that failed is retried 10 times before halting.
	Data Items:		.CPPTP = count of page table parity errors; ** = ACs saved in CPU status block.
PTT	CORE1	DEBUG	Past Top Of Table SETZRS sets zeros in a table. This stopcode occurs when the SETZRS routine attempts to zero more bits than exist.
	Data Items:		(T2) = top of table; T4 = final address to clear bits

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
PUF	SEGCON	JOB	PATH UUO Failed PTHFIL looks up a file and returns the path for it. This stopcode occurs when the PATH. monitor call fails.
PUN	FILUUO	DEBUG	PPB Use-count Negative The PPB use-count was decremented to -1.
QEF	DTESER	STOP	Queue Entry Full STXPPC starts primary protocol on a DTE. This stopcode occurs when the Tops-11 queue is full, even though primary protocol is just starting.
	Data Items:		T1 = address of entry from Tops-11 queue
QFU	QUESER	JOB	Q-Blocks Fouled Up TSTAAC tries to determine if we need to increment the read count in the access table to make it stay around. This stopcode occurs when a zero entry is found in the link to the next queue.
RBQ	SCHED1	STOP	Requeueing To Beginning Of Queue QFIX is used in the requeuing of jobs. This stopcode occurs when an attempt is made to requeue a job to the beginning of the same queue.
RCC	SCNSER	STOP	Range-Checked Chunk This stopcode is called by several places, each doing a range check on a character address. This stopcode occurs when the character address is not within the TTY buffer pool.
	Data Items:		T2 = character address character address (in T2) that was not in the TTY buffer pool. This can be caused by attempting TTY output without first setting up U to point to an LDB.
RCD	SCNSER	DEBUG	Random Chunk Discrepancy This stopcode is called from a number of places in SCNSER where it is noted that chunk pointers and counts are inconsistent.
RDN	TAPUUO	DEBUG	Regular DDB Not Found SETODN sets the density in the other DDB. This stopcode occurs when there is no regular DDB.
	Data Items:		R3 = UDB
RDP	FSXKON	DEBUG	RS04 Doesn't Position FSXPOS is a stopcode-only routine that is used when the FILIO module tries to position an RS04.
RDS	SEGCON	STOP	REMAP Didn't Skip GETFIN remaps the save file after it has been read in its entirety. This stopcode occurs when the remap fails because the arguments are wrong, pages do not all exist in the page specified, or moving the pages to the virtual address specified would cause the high and low segments to overlap.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
REH	ERRCON	HALT	Recursion In Error Handler DIE recovers/reloads after an internal system error. This stopcode occurs when another stopcode occurs before the previous one is done.
RFU	TAPSER	STOP	Recovery Fouled Up ERPINT handles interrupts while error recovery is in progress. This stopcode occurs when the function code for dispatching is greater than 6.
	Data Items:		T2 = function code; T1 = pointer
RHN	FILIO	DEBUG	Reread Home Block-Count Negative SETMDL sets the file to idle when monitor I/O is done. This stopcode occurs when the flag DEVRHB(F) indicates that the HOME blocks are being reread, but the flag that tells the number of units that are rereading HOME blocks (HOMFG) is negative.
RIE	XTCSER	DEBUG	Remote Interrupt Error This stopcode occurs if there is any error bits are lit on an interrupt from a remote system on the DA28.
RIF	DPXKON	DEBUG	RP10 Isn't Fancy These are stopcode-only routines. This stopcode occurs when the monitor attempts an RP04-only function, such as an UNLOAD, on an RP10-controlled device.
RJZ	SCHED1	STOP	Requeue Job Zero QXFER is used in the requeuing of a job. This stopcode occurs when a call is made to this routine with a job number less than or equal to zero or greater than JOBMAX.
	Data Items:		J = job number
RLD	STOP	UUOCON	Reload Monitor This is a result of the RECON. UUO function .RCRLD, which is callable by a privileged user or the CONFIG command SHUTDOWN. (Refer to the TOPS-10 Operator's Guide for more information.
RNP	VMSER	DEBUG	Returning Non-existent Page DNZSPG returns nonzero section pages to free core. This stopcode occurs if an attempt is made to return a page that does not exist.
ROU	ONCMOD	HALT	Ran Out Of Units NXTSAT reads all the SATs on a unit and computes the number of free clusters left in each SAT. This stopcode occurs when the SAT pointers indicate that there is another unit, when in fact there is no other unit.
	Data Items:		P4 = number of units remaining

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
RPM	ONCMOD	DEBUG	Retrieval Pointer Mismatch FILMAN finds and sets up all structures on a system. This stopcode occurs when the unit-change pointer in the file SAT.SYS did not point to the next unit in the file structure.
	Data Items:		T2 = expected unit-change pointer; T3 = logical unit number expected
RPZ	VMSER	STOP	Returning Page 0 The monitor tried to return page 0 to the free page list, but it could not.
RQD	SCNSER	DEBUG	RECINT Queue Discrepancy This stopcode occurs if the user just emptied the RECINT queue but the taker and putter pointers do not match.
RQF	SCNSER	DEBUG	RECINT Queue Full This stopcode occurs the RECINT character queue wraps around.
RSJ	CLOCK1	DEBUG	Requeue Same Job REQUE requeues a job to run. This stopcode occurs when an attempt is made to queue the same job again.
	Data Items:		J = job number
RTM	NETDEV	STOP	Requested Too Much TRQPCB gets a terminal Protocol Control Block (PCB) T1 = minimum number of bytes
RWD	FILIO	DEBUG	Returning Wrong Unit's DA DWNDA gives up a disk allocation request. This stopcode occurs when the unit's DA that is being dequeued is not correct for this job.
	Data Items:		PJOBN = job number
RWS	VMSER	DEBUG	Returning Space To Wrong Section GVFWDS returns funny space pages. This stopcode occurs if the monitor attempts to return funny space from a section from which it cannot be allocated.
SAC	ERRCON	DEBUG	Strange APR Condition This stopcode occurs when an APR interrupt occurs with no known error bits set.
	Data Items:		S = APR error condition
SAU	CPNSER	DEBUG	Scheduler Already Unlocked ULKSCD unlocks the scheduler interlock. This stopcode occurs when the interlock is already free.
	Data Items:		SCKLOK = Interlock

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
SBn	COMMON	CPU	SBUS Error Alone This stopcode occurs when an SBUS error alone came up in CONI APR. This is a serious hardware problem. Call your Field Service representative.
SBT	FILUUO	DEBUG	Shouldn't Be Truncating CLSRIB closes a file. This stopcode occurs when an attempt is made to truncate unwritten blocks, but the highest block number in the file is too small.
	Data Items:		P2 = current block of RIB; P3 = DEVREL
SBW	VMSER	DEBUG	SWPLST Bits Wrong This stopcode occurs when an entry in SWPLST shows both that I/O is in progress and that I/O is complete.
SBZ	VMSER	STOP	Swap Block Zero This stopcode occurs if, in picking the next swap list entry, we find that it specifies an invalid disk address.
SCB	XTCSER	DEBUG	Spurious CONI Bit This stopcode occurs if certain random error bits are lit on the CONI status read on an XTC interrupt.
SCR	DEBUG	SEGCON	Segment Couldn't Be Read INPSEG is called to read in a high segment. This stopcode occurs if INPSEG returns nonskip.
SDE	FILIO	DEBUG	SAT Doesn't Exist GIVBLK returns disk blocks. This stopcode occurs if GIVBLK cannot find the SAT in which the blocks are supposed to exist.
SDS	UUOCON	DEBUG	SWPADR Didn't Skip SWPADR converts a swapping space address to a unit/disk address. This stopcode occurs when the JOBPEK UUO determines it needs to read/write the swapping space and calls SWPADR to convert a swap address to a unit/disk address. SWPADR does not have a nonskip return.
SER	FILUUO	JOB	SETDDO Error Return FAKDDB sets a DDB. This stopcode occurs when subroutine SETDDO gives an error return indicating no core is available to build a device data block, although space had been found just before the call.
	Data Items:		T1 = address in memory found previous to call
SFU	FILIO	DEBUG	Swapper Fouled Up SWAPIO puts a swap request into the queue. This stopcode occurs when this routine is called with no request.
	Data Items:		SQREQ = 0, should have been the request

Table 11   TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
SGH	DTESER	DEBUG	Shouldn't Get Here This is a catchall stopcode in DTESER. Examine the stack for context.
SHU	SCHED1	DEBUG	Swapper Hung Up NOFORC times out devices that are active to a job waiting to be swapped out. This stopcode occurs when the device-hung timer times out.
SIE	VMSER	DEBUG	SWPLST Is Empty DLTSLE was called to delete a SWPLST entry, but there were no entries in SWPLST.
SIN	VMSER	DEBUG	SWPCNT Is Negative This stopcode occurs when the count of the number of outstanding swapping requests becomes negative while an entry from SWPLST is being deleted.
SLF	VMSER	DEBUG	SWPLST Full This stopcode occurs when there is no room for an entry in the swap list table.
SLM	FILUUO	DEBUG	Search List Missing FNDFRA is used when the PPB and/or the UFB are deleted. This stopcode occurs when the SETSRC routine cannot set up a search list, even though it seemed possible when the call started.
SLO	FILFND	JOB	Search List Overflow SLXSLO is a stopcode-only routine. Examine the stack for the location of the error.
SLZ	VMSER	DEBUG	SLECNT Is Zero This stopcode occurs when the subroutine to find an entry in the SWPLST table is called when there are no entries in the table.
SME	ERRCON	HALT	Serious Memory Area A memory parity error occurred in the monitor.
SMU	SCHED1	DEBUG	SWPCNT Messed Up SWAP is used to swap jobs.
	Data Items:		J = job number; SWPCNT = count of completed swapping operations.
SNF	LOKCON	STOP	Segment Not Found LOCK0 locks a segment in core. This stopcode occurs when the monitor cannot find a segment that contains a certain page.
	Data Items:		T3 = absolute page address being looked for

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
SNI	SWPSER	DEBUG	Swapping Not In Progress SWPINT is used when paging or swapping I/O is done for a monitor that includes virtual memory. This stopcode occurs when the swap-in progress count goes negative.
	Data Items:		SPRCNT = Swap-in progress count
SNO		DEBUG	Segment Not Owned By Anyone While attempting to migrate a high segment from a unit that is being taken off the active swapping list, CHKMIG found a segment that should have been in use by some job but no job was linked to the high segment.
SNS	NETDEV	STOP	NTRPCB Not Set Up TWRPCB writes back the count field and updates the pointer in the PCB. It also removes garbage from the stack.
	Data Items:		T1 = minimum number of bytes
SOD	SCHED1	STOP	Space On Disk SWAPI swaps in either a job or high segment. This stopcode occurs when the core-allocation routine (CORGET) assigns space on the disk, but the assignment is illegal.
	Data Items:		J = job number
SOR	ERRCON	STOP	Segment Out Of Range ERRPNT prints common error messages. This stopcode occurs when the job or segment number is too large.
	Data Items:		J = job number
SPM	FILUUO	JOB	Second Pointer Missing UFDNXT initializes the next block for a directory. This stopcode occurs when the pointer to the second RIB is missing from the first RIB.
	Data Items:		T3 = Supposed location of second RIB pointer
SRE	ONCMOD	DEBUG	SAT Read Error NXTSAT reads all SATs on a unit and computes the number of free clusters left in each SAT. This stopcode occurs when a read error occurs while reading the SAT.
SRO	SWPSER	STOP	Space Ran Out NXUN is used when we have filled the current unit and we need more swapping space. This stopcode occurs when there are no more units for swapping.

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
SSD	SWPSER	STOP	Swap Space Disappeared FOUND is used when contiguous space has been found on a unit for swapping. This stopcode occurs when an attempt is made to allocate that space, which for some reason is no longer available.
	Data Items:		U = address
SSO	LOKCON	STOP	Segment Swapped Out LOCK0 locks a segment in core. This stopcode occurs when a high segment that is neither dormant nor idle has no low segment in core.
SWN	SWPSER	DEBUG	SQREQ Went Negative SWPINT is used when paging or swapping I/O is done. This stopcode occurs when the count of paging or swapping requests goes negative.
TC0	XTCSER	DEBUG	XTCSER Stopcode Zero XTCSER has found the controller free and the unit unlocked, but there are requests in the queue waiting to be processed.
TC1	XTCSER	STOP	XTCSER Stopcode One XTCSER should have already set a "Waiting for Input" message, but has not.
TC2	XTCSER	DEBUG	XTCSER Stopcode Two XTCSER expected the DAS28 to be idle, but it was not.
TC3	XTCSER	DEBUG	XTCSER Stopcode Three The number of pseudo active tasks in the XTC UDB went negative.
TC4	XTCSER	DEBUG	XTCSER Stopcode Four The number of pseudo active tasks in the XCT KDB went negative.
TC5	XTCSER	DEBUG	XTCSER Stopcode Five The number of pseudo active tasks in the XTC UDB went negative.
TC6	XTCSER	DEBUG	XTCSER Stopcode Six The number of pseudo active tasks in the XTC KDB went negative.
TC7	XTCSER	STOP	XTCSER Stopcode Seven This stopcode occurs when XTCSER expected to have the controller interlocked but found it did not.
TCI	FILUUO	DEBUG	Truncation Check Inconsistent RENDEL deallocates or truncates on a RENAME. This stopcode occurs when an attempt is made to truncate too many blocks and a check on the same had already succeeded.
	Data Items:		P1 = AOBJN pointer; P3 = number of blocks

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
TMP	FILIO	DEBUG	Too Many Pointers PTRWRT copies RIB pointers into a monitor buffer and writes it. This stopcode occurs when there are more retrieval pointers than can fit in a RIB. The counter DEVRSU should prevent this from happening.
	Data Items:		T2 = remaining pointers (IOWD)
TMR	REFSTR	HALT	Too Many Retrieval Pointers SATRBS stores retrieval pointers in the SAT.SYS read-in block. This stopcode occurs when the SAT byte pointer is messed up.
	Data Items:		T1 = SAT byte pointer
TMU	ONCMOD	HALT	Too Many Units NXTSAT reads all SATs on a unit and computes the number of free clusters left in each SAT. This stopcode occurs when there are pointers to more units after the last has been retrieved.
	Data Items:		U = pointer to more units
TNI	DTESER	STOP	Tops-10 Not Idle DTERNG answers a Tops-10 doorbell interrupt. This stopcode occurs when the monitor decides that this is a direct transfer and expects the DTE to be in an idle state, but it is not.
	Data Items:		T1 = Tops-10 DTE state code; 0 = idle
TQP	DTESER	STOP	Found Tops-11 Queue Pointer This stopcode occurs when a byte pointer to the Tops-11 queue is found while starting primary protocol on a DTE.
	Data Items:		T1 = pointer to first word in Tops-11 queue
UAF	STOP	UNIBUS	Addressing Failure SEILM processes page-failure traps. This stopcode occurs when what appears to be a page fault turns out to be a UNIBUS addressing failure.
UDE	FILIO	DEBUG	Unit Doesn't Exist RIBCUR reads the current RIB. This stopcode occurs when a requested unit is not in any file structure.
	Data Items:		DEYRBU = current RIB logical unit number pointer
UDM	FILUUO	JOB	UFD Data Is Missing UFDALB allocates a block for a UFD. This stopcode occurs when the core tables show that the UFD is longer than it actually is.
	Data Items:		T3 = supposed number of blocks of this UFD.
UFI	FILUUO	STOP	Unit Free-Count Inconsistent CLSOU5 is used during a CLOSE after finding a unit with space on it. This stopcode occurs when an attempt is made to allocate the space, but no space is available.
	Data Items:		T2 = Number of blocks needed

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
UID	D8SINT	DEBUG	Unexpected Input Done T1ODON handles Tops-10 DONE interrupts from the -11. This stopcode occurs when no input is expected.
	Data Items:		T1 = CPU number
UIF	ONCMOD	HALT	Unit Already In File Structure DMKSTR sets up structures according to tables in ONCMOD. This stopcode occurs when a unit appears in more than one structure.
UIL	ERRCON	STOP	UUO At Interrupt Level EMUERR is called when an illegal monitor call occurs at exec level. This stopcode occurs when the monitor call occurs at interrupt level.
UIP	XTCSER	DEBUG	Not A Unique Interrupt XTCSER decided to call routine DDBINT (for DDB doing data I/O) instead of UNIINT, but XKBIUN (pointer to UDB requesting interrupt) was nonzero, implying XTCSER should have called UNIINT. Only one of DDBINT or UNIINT should be called.
ULE	LP2SER	JOB	Unexpected LP20 Error LPTERR handles VFU errors for LP20 controllers.
	Data Items:		F = DDB; T1 = function
ULP		DEBUG	UBA Lost Its PI Assignment KSSEC performs once-a-second tasks for the KS10.
UNF	FILUUO	DEBUG	UFB Not Found NAMNW updates RIBNAM, RIBEXT, and RIBPPN when there is a CLOSE for RENAME. This stopcode occurs when a RENAME is done across UFDs and the UFB is not found.
	Data Items:		T1 = Structure number; T2 = Start of UFB chain
UNJ	COMMON	STOP	UUO From Null Job This stopcode occurs when the null job executes a monitor call other than the doorbell call.
UNL	VMSER	DEBUG	UPMP Not Last This stopcode occurs when the UPMP is not the last page swapped out.
UNR	UUOCON	DEBUG	UPMP Not Right GETUVP is called by the JOBPEK UUO to read a page from the swapping space. This stopcode occurs if the job's UPMP is in core, but is not mapped as the current UPMP.
UPC	FILUUO	JOB	Unit-Change Pointer Clobbered SETENC enters a file. This stopcode occurs when the pointer to a unit of a RIB is lost during RIB definition.
	Data Items:		S = status bits; T3 = location of the access table

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
UPF	KLSER	HALT	Unexpected Page Fail This stopcode occurs when there is a page fail trap during a recovery attempt of an AR/ARX trap, which is not caused by a test reference.
	Data Items:		.UPMP+.LMPFW = page fail code; .UPMP+.LMPFP = page fail PC
UPI	FILIO	DEBUG	Unit Pointer Illegal EXTRIB creates an extended RIB. This stopcode occurs when an attempt is made to create an extended RIB on a nonexistent unit.
	Data Items:		T2 = change unit pointer (should have been a real unit pointer)
USW	TAPSER	DEBUG	Unit Status Wrong TAPSIO is used when the swapper level wants to start I/O on a unit. This stopcode occurs when the unit status is not as expected. For example, the unit was started (possibly on another controller) when it should have been stopped.
	Data Items:		RUBSTS(U) = unit number
WAD	VMSER	DEBUG	WSBTBL And AABTBL Discrepancy This stopcode occurs when there is an access page fault for a page that should have the access allowed bit on in the page map.
WCD	DEBUG	DTESER	Wrong CPU For DTE This stopcode occurs if DTEDSP is called while running on a CPU other than the CPU on which the DTE which is being accessed exists.
WCN	DTESER	STOP	Wrong CPU Number DTEQUE places an entry into a DTE's To-11 queue. This stopcode occurs when the CPU number on which the DTE for this entry is located is not this CPU.
	Data Items:		T1 = the CPU number on which the DTE for this entry is located; .CPCPN = this CPU number; F = DTE control block address
WDU	DEBUG	DTESER	Wrong DTE User This stopcode occurs if DTEDSP is called to perform some function on the DTE, but the type of user of the DTE is not the type of user which called DTEDSP.
WEM	NETSER	STOP	William E. Matson This is a catch-all stopcode. Examine the stack for the location of the error.
WPT	KLSER	HALT	Wrong Parity Trap This stopcode occurs when there is a page fail while trying to recover from an AR/ARX trap, which occurred because of a test reference, but the page fail code is not 36.
	Data Items:		T1 = page fail code; .UPMP + .LMPFW = page fail code; .UPMP + .LMPFP = page fail PC

Table 11 TOPS-10 Stop Code Summary (Cont)

Name	Module	Type	Message and Explanation
WRF	COMMON	CPU	Warm Restart Failed A condition such as a DEX has occured and the monitor has attempted to warm restart, but cannot due to various conditions being in effect at the time of the original failure.
WRJ	COMMON	JOB	Warm Restart Got Job A condition such as a DEX has occured and the monitor has attempted a warm restart. The condition occured while some job other than the null job was running in user mode.
WSM	FILIO	STOP	Wrong Size Moved Routine CSSETL is called to set the size of a BLT to/from the disk cache. This stopcode occurs if the size is greater than one block worth of data.
WTP	CLOCK1	JOB	Wrong Type Of PDL WSCHED is entered at monitor call level when a job goes into I/O wait or sharable-device wait. This stopcode occurs when the address of the pushdown list is too low to be a monitor call pushdown list.
	Data Items:		P = pushdown list
XPW	LOKCON	STOP	Exchanged Page Went Away FIXMAP finds a page with which a page was exchanged and fix the map slot for that page. This stopcode occurs when the monitor cannot find the page that was exchanged.
XTH	SCHED1	DEBUG	XJOB Too High FNDXPN finds the expanding job. This stopcode occurs when the count of the number of jobs that must be swapped out and back in to satisfy a core expansion request is positive, but no expanding job is found.
ZBC	REFSTR	DEBUG	Zero Blocks Per Cluster REFSTR refreshes a structure. This stopcode occurs where the number of blocks per cluster equals zero.
	Data Items:		T1 = IOWD for home block; P2 = address of structure data block

STOPCODE DEFINITIONS

A list of the DCN/D36 stopcode subtypes is presented on the following pages in alphabetical order. The list shows the name of each stopcode, the type of stopcode, the stopcode message (for which the name is a symbol), an explanation of the cause, and suggested recovery procedures. The occurrence of any of the following stopcodes indicates an internal problem with the code and can be reported to DIGITAL through a Software Performance Report. When analyzing a crash dump caused by a DCN/D36 stopcode, an unusual procedure must be followed to find the location in the code that invoked the crash. With these stopcodes, the AC CX points to the instruction immediately following the stopcode invocation. The stopcode invocation looks like:

```
XCT XXXXXX
```

Where:

```
XXXXXX            is a mnemonic identifying the error.
```

Example:

```
XCT ROUNSO
```

The three types of stopcodes described in this document are:

1. INF,
2. CHK, and
3. HLT

INF Stopcodes

An INF stopcode is not immediately harmful to any job or to the system. It acts as an alert that something unusual is happening. A message of the following form is printed on the CTY:

```
%%DECnet buginf:  XXXXXX - message
```

Where:

```
XXXXXX            is a mnemonic identifying the error.

message           is the stopcode message.
```

Example:

```
%%DECnet buginf:  ROUBTF - Bad Test message format
```

CHK Stopcodes

A CHK stopcode is the same as a DEBUG monitor stopcode or a DCN stopcode. Such a stopcode is not immediately harmful to any job or to the system. A CHK stopcode indicates the monitor encountered an internal error at the interrupt level and is performing a dump. Processing continues and a message of the following form is printed on the CTY:

```
?CPU1 monitor error.  Stopcode name is DCN
Job jobn on TTYnnn running name User [PPN]
UUO is octal representation at user PC address
XXXXXX - message
```

Where:

```
jobn              is the number of the job causing the
                  error.

nnn               is the number of the controlling TTY.
```

name	is the name of the program running for that job.
PPN	is the project-programmer number for that job.
octal representation	is the octal representation of the monitor call failing for that job.
address	is the value of the program counter for that job.
XXXXXX	is a mnemonic identifying the error.
message	is the stopcode message.

Example:

```
?CPU1 monitor error.  Stopcode name is DCN
Job 46 on TTY103 running DDT User [10,5535]
UUO is 47000777776 at user PC 002644
LLIDIR - Duplicate Interrupt Message Received
```

HLT Stopcodes

A HLT stopcode is the same as a STOP monitor stopcode or a D36 stopcode. Such a stopcode indicates an internal error that endangers the integrity of the entire system. All jobs are aborted and the system immediately begins to dump and reload the monitor. A message of the following form is printed on the CTY

```
?CPU1 monitor error. Stopcode name is D36
Job jobn on TTYnnn running name User [PPN]
    jobn       nnn          name        PPN
UUO is octal representation at user PC address
    octal representation            address
XXXXXX - message
XXXXXX   message
```

Where:

jobn	is the number of the job causing the error.
nnn	is the number of the controlling TTY.
name	is the name of the program running for that job.
PPN	is the project-programmer number for that job.
octal representation	is the octal representation of the monitor call failing for that job.
address	is the value of the program counter for that job.
XXXXXX	is a mnemonic identifying the error.
message	is the stopcode message.

Example:

```
?CPU1 monitor error.  Stopcode name is D36
Job 46 on TTY103 running DDT User [10,5535]
UUO is 47000777776 at user PC 002644
COM911 - The date is past 9 November 2021
```

LIST OF STOPCODES

Name	Type	Message and Explanation
COM911	HLT	The date is past 9 November 2021 The two-byte Julian half-day field in an event message is limited to 9 November 2021. The routine above calculated the Julian half-day, and found that it overflowed. It is unlikely that the date itself really went past 2021. An AC was probably destroyed, or the routine to get the time from the monitor is returning invalid information.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
COMAFB	CHK	A free block pointer is bad There is a block on a free list, most likely just added to the list, whose address is not in the expected range. The offending pointer is in P1. A subroutine whose address is on the stack is probably returning a block to the wrong free list, or is returning an invalid pointer.
COMCHA	CHK	Number of available FB blocks to large When checking the CH begstr for a type of block, the code determined that more blocks were available than there were originally. DNCHFB is supposed to defend against this. CHNUM was probably trashed.
COMCHB	CHK	CH pointer off by a few A pointer internal to the core management routines is off by a few words. You have probably trashed an AC by adding to it or XORing some bits.
COMCHO	CHK	CH pointer out of range In the core block checking routines, the internal pointer to the CH begstr applying to this type of block is bad. Your executable code was probably trashed.
COMCID	CHK	Couldn't initialize DECNET SCTINI found some reason to object to the DECnet environment. See SCTINI for reasons why it takes a nonskip return.
COMDNP	CHK	DNGPOS called with bad MS In range checking the ac MS, its contents were outside the range of addresses used for the MS block. Trace back to the caller and find out why it has a junk pointer.
COMFBA	CHK	FB available count is wrong DNCHFB walked through a free list and found a different number of blocks on the list than the header indicated. A forward pointer was probably destroyed in a previously returned block.
COMFBB	CHK	FB in data base is off by a few DNCHFB found a block on a free list, most likely just returned, whose address is not on a block boundary for blocks on this free list. The offending pointer is in P1. A caller on the stack is probably returning a junk pointer, either a real pointer to a block that has been incremented or decremented, or a completely junk pointer.
COMFBF	CHK	FB is already on free list The block that P1 points to is already on the free list and is being returned again. A caller on the stack is returning a block that is already free.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
COMFBO	CHK	FB pointer is out of range When checking a free block pointer, the code found that the pointer is not pointing to the free core allocated for this type of block. Identify the routine that supplied this pointer.
COMFBT	CHK	FB pointer is off by a few A free block pointer is off by a few words. The user of this pointer probably added a constant, and forgot to restore it when returning the block. Trace the user of this pointer, and make sure the pointer is valid when given to the memory manager.
COMFWZ	CHK	Tried to free words at zero DNFWDS was called with a pointer of zero.
COMIEL	CHK	Illegal end of list pointer CHAVL, the available count, said there was at least one block on the free list, but the first pointer was zero. A forward pointer was probably destroyed in a previously returned block.
COMMMI	CHK	Memory manager must be initialized The field CHBOT, which indicates where a free core pool starts, is zero. This field gets set when the core manager is initialized. If DNINIM has already been called, check to make sure it is initializing all CH blocks.
COMMMS	CHK	Bad pointer passed to memory manager When DNGWDS gives out a block of core, it leaves a check word immediately before the first word of core given to the user. This word contains the length of the block, and a "check" quantity to verify that this block contains what is expected. This bug means that the check word has been trashed, or the pointer that was passed to the memory manager is bad.
COMMPR	CHK	Message pointer check DNFMSG caller tried to return a piece of memory that is not in the range of message blocks. See stack for caller and find why it is trying to return a bad message block.
COMMS1	CHK	Bad pointer passed to memory manager Header word trashed or bad pointer.
COMMS2	CHK	Bad pointer passed to memory manager Header word trashed or bad pointer.
COMMS3	CHK	Bad pointer passed to memory manager Header word trashed or bad pointer.
COMMTS	CHK	New message block too short
COMMZP	CHK	DNMINI was passed a zero pointer A caller probably meant to ask for zero bytes of user data in T2 and mistakenly put the count in T1. T1 is supposed to contain the pointer to the message block being refreshed. Find caller on the stack and fix it.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
COMODP	CHK	DNGOPS called with bad MS In range checking the ac MS, its contents were found to be outside the range of addresses used for the MS block. Trace back to the caller and find out why he has an invalid pointer.
COMSCO	HLT	Section 1 assertion failed This code is supposed to be running in section one or greater, yet the TESTS1 macro found code running in section zero. Look on the stack for the address of the TESTS1 macro that detected the fault. Find the code that fell into section zero and fix it.
COMSTB	CHK	Smear request too big The caller requested that a very large block be smeared. Find out what the caller really wanted to smear and fix the call.
LLIAAL	CHK	Arg blk to NSPACC wrong length
LLIAK2	CHK	Duplicate msg put on ACK queue
LLICGT	CHK	Can't DNMINI a msg blk DNMINI refused to initialize a message block. This should never happen when the number of bytes requested is zero. There is probably something wrong with the message block, its pointer, or DNMINI.
LLICLS	CHK	Tried to close in non-pre-close state
LLIDDP	CHK	Tried to destroy non-DP port
LLIDIR	CHK	Duplicate Interrupt Message Received The code found a duplicate interrupt message on the unacked interrupt receive queue. One should never get this message because the code is not allowed out of the NSP interlock with anything in this receive queue. Identify the problem. Either the interrupt flow control malfunctioned and sent more than one data request, or the remote node sent an interrupt message without a data request.
LLIFNS	CHK	SCTL passed bad NSPpid
LLIFZM	CHK	Tried to free zero msg
LLIHTG	HLT	INIHSH cant get a hash table
LLIHTS	HLT	NSPHTS not set up
LLIIFC	CHK	Illegal flow control type
LLIILI	CHK	Interrupt message must not be segmented
LLIIVO	CHK	Illegal call vector offset
LLILMA	CHK	RETBUF left LAR # LMA
LLINNI	CHK	NSP not yet initialized NSP (LLINKS) will reject all messages received from either Session Control or Router until DECnet initialization is complete.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
LLINNP	CHK	No memory for reserved NDB
LLINRP	CHK	No memory for reserved ports
LLIOAL	CHK	OPEN arg blk wrong length
LLIODN	CHK	NSIODN got message with NMACK=0
LLIORC	CHK	ORC should never be negative
LLIPIM	CHK	PROCXQ found illegal message type
LLIQIN	CHK	Queued interrupt message illegal
LLIRFN	CHK	NSP called RESPRC with bad fcn code LLINKS's reserved port handler was called with an unknown Session Control function code in T3. If there is a new Session Control function code and this routine doesn't know about it, the function code must be added. Otherwise, look on the stack to find which LLINKS routine called Session Control with an invalid function code. An occurrence of this stopcode indicates that the version of LLINK and Session Control are skewed.
LLIRMG	CHK	NSP called RESPRC without msg blk LLINKS's reserved port handler was called with no message block pointer; T4 is zero. Look on the stack to find which LLINKS routine called Session Control with T4 containing zero.
LLIRMH	CHK	RMVHSH didnt find port
LLIRQ2	CHK	Duplicate msg requeued
LLIS2S	CHK	Illegal flow control at PRCRQS
LLISCM	CHK	ELSCM should not have been set
LLISIF	CHK	SENDC0's DNMINI failed
LLITNE	CHK	Unknown Event Type at NSPEVT T1 contains an illegal NSP event type. Note that NSPEVT is called by SCLINK as well as LLINKS. Caller address is on the stack.
LLIWNE	CHK	Can't get event arg blk There isn't any free memory for an event argument block. Presumably it really ran out, but some may have been lost. Either allocate more free memory or accept that some events will be lost.
LLIXM2	CHK	Duplicate msg queued for xmit
LLIXNN	CHK	NSP not yet initialized
LLIXR2	CHK	Duplicate msg requeued for xmit
LLIXVO	CHK	Illegal Router call
LLIXZM	CHK	NSP called with no message block
NRTBPM	CHK	Bad pointer passed to memory manager
NRTFW0	CHK	Tried to free words at zero
NRTHBC	CHK	NRTHBR should never be called

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
NRTILS	CHK	NRT link in unexpected state
NRTINP	CHK	NRT Input to DECnet failed
NRTOUD	CHK	NRT output to DECnet failed
NRTPCL	CHK	Partial Configuration Msg Loss
NRTSAB	HLT	No memory for NRT's SAB
NRTSET	CHK	SCTPSQ returned wrong channel info
NRTSJB	HLT	No memory for NRT's SJB
NRTSJM	HLT	No memory for NRT's SJB
NTMBCF	CHK	Bad coded field on output While formatting output for a SHOW, the program was requested to generate a Coded field of more than one byte. The program is not coded for this function. Look at the descriptor block pointed to by NT. Check to see if this item is supposed to be a multiple byte Coded. If not, fix the item's entry. If it is correct, write the code to handle multiple-byte Coded fields.
NTMBCX	CHK	Bad call to NMXXND A "layer" (any routine described in NMXDSP) that handles information in Router's routing vector was called. All the information in this vector is supposed to be read-only, but it was called for a set or clear function. Look at the descriptor block pointed to by the AC NT, and determine which item caused this layer to be called. Then fix the item's entry to indicate that this is a read-only parameter.
NTMBDL	CHK	Bad multiple byte length The code generates output for a numeric field, and was asked to generate an illegal number of bytes.
NTMBFP	CHK	Bad format type encountered While the program was in the process of reading a value from the user string, the descriptor tables returned an invalid format for this item. The AC NT points to the descriptor for this item, and field NTSEQ should tell which item is being referenced. Fix the entry for this item so it contains a valid format type.
NTMBLI	CHK	Bad Line id Router gave an error return when asked for the state of a circuit. The only valid error return from this routine is due to a nonexistent circuit.
NTMBSS	CHK	Bad string size in NMXNI4 When attempting to copy an identification string from NMXVAR into a free core block, the string claimed to be too long to fit in either block. Find out how this byte (containing the length) got trashed.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
NTMCNO	HLT	Circuit name overrun More than 16 bytes of data were returned to a 16 byte field. The data beyond the buffer was trashed. Examine the algorithm at NMXC2N to determine why the code returned more bytes than were expected. To avoid this halt, fix the above code to check for overrun while it is producing the bytes.
NTMDVI	CHK	NMXDSP value illegal The code called a layer to obtain a value or set a value for an item. The routine value in the descriptor block pointed to by NT was illegal. Examine the data structure pointed to by NT. Probably this was caused by a trashed NT, since the descriptor block generation macros are supposed to range check this value.
NTMEOR	CHK	Entity type out of range While double-checking the entity ID, before dispatching on it, the code found the type value was illegal. Since the value supplied by the user is checked at GETBLK, this means that field NXENT was trashed.
NTMESL	CHK	Event string too long The code received an event from a DECnet layer, and the length of the data string was too long to fit in the storage block. Either increase the size of .NQMXS or cause the DECnet layer to return a smaller string.
NTMFOR	CHK	Format out of range In formatting output for a SHOW, the format block for this item had an illegal format type. See NTMBFP.
NTMFUR	CHK	Function code out of range The code is going to dispatch by function code, and found that the function code is out of range. Since the function code supplied by the user is checked in GETBLK, this means that field NXFNC was trashed.
NTMILI	CHK	Invalid Line ID NMXDLL was called to perform a function for a line, and the previously validated line ID is bad. The probable cause is that something is trashing NXNUM.
NTMILN	CHK	Illegal number size The code is going to read a numeric value from the user's string. The format descriptor block for this item specified read of an illegal number of bytes.
NTMINT	CHK	Invalid numeric type The code is generating output for a numeric field, and was asked to generate something other than decimal, hexadecimal or octal.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
NTMKOR	CHK	Kontroller out of range in Circuit-id The Kontroller field in a line-id is out of range. The value LD.MAX defines the number of Kontrollers known by D36PAR, and thus by NTMAN. The most likely cause of this error is a trashed AC. Note: A Kontroller is any device driver with which Router will interface. It is used to define the name of a circuit/line, under the assumption that each Kontroller will control only a single line type.
NTMNEC	CHK	No error code, with error return Some routine took the nonskip return, but did not give an error code by calling NTExxx. This means that the program returned to top level and field NXERR was zero. Determine which routine is failing, and make the error return give an error code.
NTMNTR	CHK	Node type is out of range The code was going to select entries to return (for function .NTSHO) and needs to know the node type (executor, remote, or loop) in order to choose the correct one. For other entities (circuit, lines) this field should contain zero. This field is set by ENTCVT.
NTMSOR	CHK	Selection criteria is out of range The code is going to select items to return (for .NTSHO) depending on the selection criteria, and found the criteria to be out of range. Fix the check in GETBLK or find out why field NXSEL is being trashed.
NTMSRF	CHK	Skipness of return fouled up The code returns from NTMAN with a skip return, but there is an error code stored in field NXERR. Identify the caller that is giving the error code (or trashing NXERR) and make it give a nonskip return.
NTMSSB	CHK	Setting a circuit substate Setting a substate is illegal. This entry should be read-only in the descriptor block, and the code should have caught this before. Make the entry in the descriptor block be read-only.
NTMURE	CHK	Unrecognized entity type NTMAN received an event from a DECnet layer, and the entity type is not legal. Find the routine that generated the event and cause it to supply a legal entity type.
NTMXNL	CHK	ROUTER doesn't know about a line The code asked router for the state of a circuit, and router gave an error return. The only valid failure is for a nonexistent circuit. The code should have previously checked the circuit ID for existence at ENTCVT.
ROUBCB	CHK	Bad circuit block pointer

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
		A Kontroller called RTRDSP with a function requiring a circuit block pointer and supplied a pointer (in T2) that failed the range check. Find out which Kontroller is supplying the bad circuit block pointer.
ROUBCD	CHK	Bad Checksum detected when building routing msg This stopcode indicates that something got trashed. Look at P1; it points to the end of the normal routing vector [RTRNRV]+[RTRMXN]. Check the vector itself (pointed to by RTRNRV) and see if the topology appears reasonable. Make sure RTRCKS is less than 16 bits.
ROUBMB	CHK	Bad message block pointer A Kontroller called RTRDSP with a function requiring a message block, and the pointer supplied (in T3) is either 0 or out of range. Determine why the Kontroller gave an invalid pointer. The pointer should originally have been obtained from this module.
ROUBNA	INF	Bad node address in Phase II NI message
ROUBSN	CHK	Bad source node in message from NSP
ROUBTF	INF	Bad Test message format
ROUBTM	INF	Bad Hello or Test message
ROUCGE	CHK	Couldn't get emergency buffer for circuit ROUTER requires that the memory manager save at least 2 buffers per circuit for ROUTER, one for the routing message ROUTER keeps for each circuit and one to guarantee some level of route-through ability. ROUTER was asked to open a circuit, but the memory manager could not guarantee the buffers. Allocate more memory or settle for fewer circuits.
ROUCGV	CHK	Couldn't get memory for event arg block
ROUCNL	CHK	Trying to call nonexistent NSP
ROUDGE	CHK	Didn't get reserved emergency msg blk The code should never run out of emergency blocks. ROUTER should allocate enough blocks and then use no more. Either ROUTER hasn't allocated enough emergency blocks or has used too many. If too many, they are probably in some input queue. Find all calls to DNGEMS and DNMINP and find who used too many.
ROUEBI	CHK	Emergency circuit buffer already in use
ROUEHB	CHK	No Message Block for Event data
ROUEHM	CHK	No Message Block for Event data
ROUIKF	CHK	Illegal Kontroller function CALKON was called with an illegal function code. The only allowed values are KF.QOB, KF.INI, and KF.HLT.
ROUILF	CHK	Illegal function code from Kontroller
ROUILS	CHK	Illegal Circuit Specified in NSP msg

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
ROUIVL	CHK	Invalid circuit state
ROUMMR	INF	Maintenance Message received
ROUNLN	CHK	Trying to return msg to nonlocal NSP
ROUNMR	CHK	NMX out of range
ROUNOR	CHK	Node number out of range
ROUNSD	CHK	Tried to call non-existent device driver
ROUNSO	CHK	NSP sent out-of-range packet
ROURML	CHK	Stored routing message format error in RTRRCR
ROUSMR	INF	Start Message received
ROUSOR	CHK	Setting state out of range
ROUUER	CHK	Unexpected end of routing message
ROUUET	CHK	Unknown event type in RTNEVT
ROUWPV	CHK	Phase II Node Verif received in wrong state
ROUWSP	CHK	Phase II Node Init received in wrong state
ROUXNZ	CHK	R2NCAL called with MB=0
ROUZXT	CHK	Tried to free msg with MB=0
SCLBFS	CHK	Bad format type not caught
SCLBNS	CHK	Bad Node Number in SC Initialization The node number used as the local node number during initialization is out of range. The quantities that are compared are derived from the MONGEN variables %RTADR and %RTMXN. Either choose a different node number for the local node, or increase the size of your network to accommodate your node number by modifying %RTMXN.
SCLCIM	CHK	NSP called SCTLCI without a msg blk
SCLFCT	CHK	Illegal function in call from NSP
SCLFMZ	CHK	Tried to free message with MB zero
SCLGTN	CHK	Global connect timer count went negative
SCLICR	CHK	Negative buffers reserved, from input LLINKS sent a buffer to SCLINK's NSFDR routine. This routine decremented the number of buffers reserved for the appropriate logical link and the count went negative. Either LLINKS offered too many buffers or the count DCNRSB was decremented or zeroed prematurely.
SCLILS	CHK	Illegal State at SCSSTS Session Control's Set Status routine was called with an illegal status code. SCSSTS should only be called from the NEWSTATE macro, which should be incapable of generating an illegal state code. Look on the stack to see who called SCSSTS and fix it.

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
SCLISU	CHK	Illegal SL at SCTINU After a Session Control function was complete, register SL no longer held a value for the SLB pointer. Find out which routine in SCLINK smashed SL.
SCLJTN	CHK	Job connect timer count went negative
SCLMF1	CHK	SNDDRM failed to send DRQs The code should never fail to send data requests when a message block already exists.
SCLMF2	CHK	SNDDRM failed to send DRQs
SCLMI1	CHK	DNMINI failed to re-init an MB
SCLMI	CHK	DNMINI failed to init msg blk
SCLNCP	CHK	No CBLOCK pointer in SJB
SCLNMB	CHK	NSP called SCTL without a msg blk
SCLNRD	CHK	No resources to send DRQs in SCCCR
SCLNZE	CHK	Passing zero error code to SCMUUO The code arrived at a routine that is supposed to store an error code for the user, but found that the error code is zero. This is an illegal value. Find which routine called SCTNIE with T1 containing zero and correct the caller's behavior.
SCLOCR	CHK	Negative buffers reserved, from output LLINKS sent a buffer to SCLINK's Output Done routine. This decremented the number of buffers reserved for the appropriate logical link and the count went negative. Either LLINKS returned too many buffers or the count DCNRSB was decremented or zeroed prematurely.
SCLSBJ	CHK	Bad Job Block Pointer Passed to SCTNSF
SCLSIM	CHK	Got a segmented interrupt message from NSP
SCLSJR	CHK	SCMUUO called NSFJR in SCLINK
SCLSJS	CHK	SCMUUO called NSFJS in SCLINK
SCLSLB	CHK	SLB bad at FRESLB There is no Session Control Job Block (SJB) for this Session Control Link Block (SLB). This error could have happened at any time during the life of the link, after it actively transferred data.
SCLSMS	CHK	STRMAT messed up the stack pointer
SCLSNM	CHK	No Message Block supplied to SCTNSF
SCLSNS	CHK	Sblock not supplied to NSFIS
SCLSPF	CHK	SLB self pointers messed up in FNDSLB
SCLSPM	CHK	Self pointers messed up in SLB
SCLSPS	CHK	SLB Self pointers messed up in SLBMAT
SCLSTM	CHK	Trying to start a CI timer that's already going

LIST OF STOP CODES (Cont)

Name	Type	Message and Explanation
SCLTFJ	CHK	Freeing SJB with SLB entries existing
SCLTFS	CHK	Tried to free wrong SLB
SCLWVS	CHK	Incoming connect with wrong Session Control version
SCLZST	CHK	Illegal state value at SCTNIU The SCLINK error handler found that the state code in this link's SLB was illegal. This is just a gateway check to see if something in SCLINK is wrong. Find the function code in the SAB and trace the probable flow from that function code.
SCMAAE	CHK	Address Check after Function Call
SCMBCN	CHK	Bad channel number The channel number obtained from the SLB field SLCHN does not match the channel number expected by the code. This probably means that the interlocks are not correctly arranged, and the SLB has changed.
SCMNP2	CHK	No SJB for SCTPSI
SCMNPD	CHK	No PDB for Job

GALAXY-10 Stopcodes

STOPCODE DEFINITIONS
An alphabetical list of the GALAXY-10 stopcodes is presented on the following pages. The list shows the name of each stopcode, the module in which it is found, the stopcode message (for which the name is a mnemonic), and an explanation.

Each GALAXY component is made up of one or more modules, thus a stopcode can be generated by a module with a name other than that of the component producing the stopcode. See example on page B-2.

When GALAXY encounters an internal error, a stopcode is generated. A message of the following form appears in the operator log file, followed by the contents of the ACs and the last 9 stack locations.

?Stopcode - **XXX** - in module **module** on **date** on **time**
Reason: message
Program is **program name** version **n(nnnn)** using GLXLIB version n(nnnn)
Crash block starts at location **address**
Last GLXLIB error: **##** **(message)**

Where:

XXX	is the mnemonic identifying the error.
module	is the module in which the stopcode occurred.
date	is the date on which the stopcode occurred.
time	is the time at which the stopcode occurred.
message	is the stopcode message.
program name	is the program running for that job.
n(nnnn)	is the version number.
address	is the location of the crash block.
##	is the number of the last GLXLIB error that occurred.

Example:

```
?Stopcode - PQI - in module QSRT10 on 23-Feb-84 on 18:43:17
Reason: Prime Queue is Interlocked
Program is QUASAR version 4(1173) using GLXLIB version
1(1161)
Crash block starts at location 674000
Last GLXLIB error: 23 (End of list reached)

Contents of the ACs:

0/ 777777777777 3 61353 1

4/ 63377 0 0 10000000

10/ 16 637163000000 0 140421

14/ 0 141577 140 777501063716

Last 9 stack locations:

-1(P)/ 312000630517 -2(P)/ 312 -3(P)/ 312000061244

-4(P)/ 312000017227 -5(P)/ 312000630517 -6(P)/ 312

-7(P)/ 312000017226 -8(P)/ 312000064261 -9(P)/ 0
```

LIST OF STOPCODES

Name	Module	Message and Explanation
ABS	GLXSCN	Atom buffer too small The command from OPR is too long to fit in the Atom buffer for parsing.
AIE	QSRSCH	Attempt to add invalid event queue entry S$EVENT detected that the entry to be added to EVENT QUEUE is not the correct size.
AMT	QSRMDA	Allocated is more than total (VOL .VLVS BLOCKS) SCNVOL detected that the number of words allocated for VOL block pointer is greater than the total number of VOL blocks.
APT	GLXINT	Unknown APR trap at PC <pc> APR CONI = <octal CONI word>
ASE	GLXMEM	Addressing space exhausted
AZA	GLXCOM	Attempt to zero the ACs Bad argument(s) passed to routine .ZCHNK.
BBF	PLRLBP	Bad backspace file Incorrect TCB status detected backspacing a file on tape.
BBR	PLRLBP	Bad backspace record Incorrect TCB status detected backspacing a record on tape.
BCN	PLRLBP	Bad call to nxtfil Routine NXTFIL was called to skip to the next file's HDR1 label, but the TCB status indicates that the tape is not positioned in user data.
BCP	PLRLBP	Bad Call to POSTAP
BDS	GLXSCN	Bad Default String The first character in the default string ($DEFAULT) is a null.
BFC	GLXSCN	Bad function code An invalid parse function code was detected in routine S%CMND.
BLI	QSRMDA	<text> The BLISS routines called by QUASAR detected an error that warranted a stopcode; "text" is the reason returned by the BLISS routine.
BME	QSRMDA	'B' Matrix Entry is Missing RETBMA called D$BMTX to find a user's 'B' matrix entry, but there is no corresponding 'B' matrix entry.
BPN	GLXMEM	Bad page number <page number> VALPAG determined that a page is not part of the initial core image or is not marked in use.
BRS	QSRFSS	BAD REQUEST SIZE The argument passed to routine GETDPA is not in the range 1 to 1000 (octal).

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
BTA	GLXTXT	Bad $TEXT argument given at address <address>BTF GLXSCN Bad table format TABLK detected two identical entries in a table. Table entries must be unique.
BTT	GLXKBD	Backing up terminal twice GLXKBD only stores the current character. The previous character cannot be retrieved.
CAC	GLXMEM	Count of Available Pages Confused M%ACQP detected that PAGSTA points at or past the top of memory.
CAD	CDRIVE	CANNOT ADD/DELETE READER TO/FROM INTERRUPT SYSTEM The PISYS. UUO failed in routine INTCNL.
CAS	PLRDSK	Can't Append to SPT list
CCD	PLRT10	Can't Change Density The call to I$SDEN for setting the density of a tape drive took the error return in I$NDEN.
CCE	ORION	Can't create list entry The call to L%CENT in ADDNOD failed to create a list entry.
CCE	QSRIPC	Can't create list entry C$SEND was unable to create a list entry in the RESEND queue list of IPCF messages to be re-sent.
CCI	SPRINT	Can't clear UFD Interlock The SETUUO UUO failed to clear the UFD (User File Directory) interlock in routine CLRUFL.
CCP	GLXMEM	Cannot create page The PAGE. UUO failed in routine CREPAG.
CCR	PLRTAP	Can't Check Ring status The TAPOP. UUO in T$WRCK failed when checking for write ring status.
CCS	CDRIVE	CANNOT CLOSE SPOOL FILE F%REL failed to close the spooled reader file in CREATE.
CCT	PLRTAP	Can't connect tape to PSI system In T$OPEN, the call to I$PICD failed to connect the tape drive to the PSI interrupt system in order to trap off-line, resulting in hung device conditions.
CCW	PLRT10	Can't Clear Watch bits The SETUUO UUO took the error return in routine I$INIT.
CDC	PLRT10	Can't Determine Density capabilities The TAPOP. UUO took the error return while performing the .TFPDN function in routine I$PDEN.
CDC	QSRT10	Can't get Disk Characteristics for Unit <unit name>

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
		The DSKCHR UUO failed in routine I$GATR.
CDD	QSRT10	Can't Determine Tape Densities The TAPOP. UUO failed while performing the .TFPDN function in routine I$GATR.
CDK	PLRT10	Can't Determine Kontroller type The TAPOP. UUO took the error return while performing the .TFKTP function in routine I$PDEN.
CDM	GALGEN	Cannot Determine Monitor Type The GETTAB UUO failed trying to determine monitor type.
CDT	QSRT10	Can't Determine Tape Track Status The TAPOP. UUO failed while performing function .TFTRK in routine I$GATR.
CEI	BATCON	Can't enable interrupts The PISYS. UUO failed in routine SYSINI.
CFC	GLXMEM	Count of Free Pages Confused GLXMEM's database is corrupt.
CFF	SPRINT	CAN'T FIND FILES TO LOAD In EXECUTE, SPRINT could not position to the head of the list of files necessary to generate an "execute" command.
CFO	ORION	Cannot GETTAB operator PPN
CFU	QSRT10	Can't Find UCB for Unit <unit name> In I$ISTR, QUASAR was unable to find the UCB (Unit Control Block) corresponding to the system structure identified.
CFV	QSRMDA	Can't Find VSL Address in VOL Entry In DELBSL, there is no link from a VOL (volume) block back to the VSL (Volume Set List). A link should have existed because the VOL block was using a link in the VSL pointing to the VOL block.
CGC	CDRIVE	Cannot Get Reader Hardware Characteristics DEVOP. UUO failed in INPGET while trying to obtain reader characteristics.
CGC	QSRT10	Can't Get Controller Type for Tape Drive <tape drive> TAPOP. UUO failed while performing function .TFKTP in routine I$GATR.
CGD	PLRT10	Can't Get Density TAPOP. UUO took the error return while performing the .TFDEN function in I$GDEN.
CGD	QSRT10	Can't get Disk Physical Unit SYSPHY UUO failed in routine I$INIT.
CGF	ORION	Cannot GETTAB FRCLIN line number
CGP	GLXIPC	Can't Get a PID

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
		C%INIT called C%CPID to create a PID / (Process ID) and C%PID was unsuccessful.
CGS	CDRIVE	Cannot Get Spool File PPN GETTAB UUO failed in routine RDINIT.
CGS	PLRT10	Can't GETTAB States word The GETTAB UUO took the error return in routine I%OPRP while trying to get the %CNSTS word from the monitor.
CGS	QSRT10	Can't Get Status of Tape Drive <tape drive> The TAPOP. UUO failed while performing function .TFSTS in routine I$GATR.
CIF	GALGEN	Command Initialization Failed The call to SCMND returned FALSE in GETANS.
CLS	GLXKBD	Can't look up status of terminal JFN The FILOP. UUO in routine K%OPEN failed while performing the .FOGET function for a terminal. Location CHNJFN contains the channel number.
CME	QSRMDA	'C' Matrix Entry is Missing D$DLCK called D$CMTX to find a user's 'C' matrix entry but the entry does not 'exist.
CMU	PLROPR	Can't Make TCB The call to G$MTCB returned FALSE in I$CREC.
CMV	PLROPR	Can't Make TCB The call to G$MTCB returned FALSE in I$CUNL.
CNE	ORION	Central site node not present The call to FNDNOD in W$NODE returned a failure while using G$HOST as an argument.
COP	QSRT10	Cannot Open Prime Queue The FILOP. UUO failed in I$OQUE when QUASAR was trying to open the master queue file.
COR	QSRT10	Cannot Open Redundant Queue FILOP. UUO failed in I$OQUE when QUASAR was trying to open the secondary queue file.
COS	CDRIVE	CANNOT OPEN SPOOL FILE F%OOPN returned an error that was not 'file already exists' in GETFIL.
CPF	PLRT10	Clear label Parameters Failed TAPOP. UUO took the error return in I$CLLP while performing the .TFLPR+.TFSET function.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
CRB	PLRT10	Can't Read Buffer size TAPOP. UUO took the error return in I$RDLP performing the .TFBSZ function.
CRD	QSRQUE	CREATE REJECTED DEFER DATA In routine Q$DEFER, the call to Q$CREATE detected errors.
CRL	GLXFIL	Can't read last byte of file
CRL	QSRQUE	CREATE REJECTED LOGOUT DATA The call to Q$CREATE in Q$LOGOUT detected errors.
CRM	PLRT10	Can't Read user's Mode The TAPOP. UUO took the error return in routine I$RDLP while performing the .TFMOD function.
CRM	QSRQUE	CREATE REJECTED MODIFY The call to Q$CREATE in Q$MODIFY detected errors.
CRS	QSRQUE	CREATE REJECTED SPOOLING DATA The call to Q$CREATE in Q$SPOOL detected errors.
CRS	SPRINT	Can't read searchlist This stopcode indicates one of two conditions. Either the JOBSTR UUO failed in GETSRC while trying to read SPRINT's current search list, or the PATH. UUO failed in GETSRC /while trying to read SPRINT's current path.
CSB	PLRTAP	Can't Set Blocksize
CSD	PLRTAP	Can't Set Density
CSI	GLXINT	Cannot set up interrupt system The PIINI. UUO failed in routine SETINT.
CSI	PLRTAP	Can't Set Industry compatible mode
CSM	PLRTAP	Can't Set DIGITAL compatible Mode
CSS	QSRT10	Can't get System Structure List The SYSSTR UUO failed in routine I$ISTR.
CSS	SPRINT	Cant set searchlist This stopcode indicates one of two conditions. Either the STRUUO UUO failed in SETSRC trying to set SPRINT's search list, or the PATH. UUO failed in SETSRC trying to set SPRINT's path.
CST	OPR	Can't set timer for parsing The call to I%TIMR returned FALSE in SETIME.
CSU	PLRTAP	Can't Switch Units The TAPOP. UUO failed in T$NUNI when trying to switch tape drive units.
CTL	GLXFIL	Cannot trim LSN in buffered mode F%IBUF does not handle Line Sequenced Numbered files. F%IBYT must be used.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
CUD	QSRFSS	CLEARING UNUSED DPA QUASAR tried to release unused space in the failsoft file.
CUF	PLRT10	CHKACC UUO Failed The CHKACC UUO failed in routine I$CKAC.
CUT	OPR	Can't unset timer after parsing The call to I%TIMR returned FALSE in CLTIME.
CWT	PLRTAP	Can't Write Tape-mark
DBC	ORION	DeBug Crash - Keep this crash Execution continued at the location following a $DEBRK macro in an interrupt ' service routine.
DDF	ORION	Delete DN60 node failed A call to L%DENT returned FALSE in DELNOD.
DSP	ORION	Delete send Failure pid table entry / inconsistency The call to CHKFSL in DELSPL failed to find an entry in the "send failure PID table."
DTL	QSRFSS	DPA TOO LARGE VALDPA detected a Disk Page Address for the failsoft file that is too large.
DTS	QSRFSS	DPA TOO SMALL VALDPA detected a Disk Page Address for the failsoft file that is too small.
DTU	GLXINT	Date/Time unavailable The GETTAB UUO to get Universal Date/Time failed in routine I%NOW.
DUF	GLXINT	DEBRK UUO failed
EEP	QSRT10	ERROR EXPANDING PRIME QUEUE The FILOP. UUO failed in routine I$WRIT.
EER	QSRT10	ERROR EXPANDING REDUNDANT QUEUE The FILOP. UUO failed in routine I$WRIT.
EWS	CDRIVE	ERROR WRITING SPOOL FILE The call to F%OBUF took the error return in OUTCRD.
FCE	GLXMEM	Free count exceeds FREINI The current count of free pages exceeds 'the initial count of free pages.
FCN	GLXMEM	Free count negative Routine REDUCE detected that the count of free pages went negative.
FFT	GLXKBD	Action FILOP. failed to terminal The FILOP. UUO failed in K%OPEN while trying to perform either the .FOSET or the .FOWRT function.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
FIT	GLXFIL	FD location requested with illegal type. Routine F%FD was called with illegal arguments.
FSE	GLXKBD	File System Error TXTINP detected an error returned from F%IBYT that was not an EOF error.
FUD	QSRFSS	FOUND UNUSED DPA An unused DPA (Disk Page Address) indicates that the failsoft file system database is corrupt.
GNF	PLRT10	GETTAB for user's Name Failed
GOF	GLXIPC	SYSTEM GOPHER IS NOT AROUND In C%INIT, GETTAB to get [SYSTEM]GOPHER's PID failed.
GSF	PLRT10	GETTAB for Serial number Failed
IAC	OPR	Argument count <count> not valid in display message An argument block of zero was found in a message from ORION.
IAM	QSRT10	Invalid Account Validation Message Returned In I$VACT, the call to A$OB2Q was unable / to find the queue header for the object type passed in the account validation ACK (acknowledgement) message.
IBN	GLXSCN	Illegal base for number The base for a number to be parsed was not in the range 2-10 (decimal).
IBP	GLXKBD	Illegal byte pointer in K%TXTI The byte pointer that CONVBP was going to convert is zero. This invalid byte pointer was found at RD+.RDDBP.
IBS	GLXFIL	Illegal byte size given An invalid byte size, out of the range 1-36 (decimal), was given in a call to open a file.
IBU	BATCON	Illegal BATCON UUO In BATCON's LUUO handler, UUOCON, an opcode was detected that was out of range.
IDM	OPR	Message argument type <argument type> not valid for display message The argument type was something other than the argument type constant, .CMTXT.
IDM	OPRLOG	Invalid Display Message Type <msg type>
IDM	PLRLBP	Invalid Date from Monitor
IEC	OPR	Invalid error code for failure An OPR failure error code is not in the range expected in routine SETFAL.
IFC	OPRPAR	INVALID FUNCTION CODE FROM COMMAND
IFM	GLXFIL	Illegal file mode in subroutine call An operation was attempted on a file, but the file was opened /in a mode that prevents the requested operation from succeeding.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
IFN	GLXFIL	Illegal IFN provided in call The IFN passed to CHKIFN was not in IFNTAB.
IIF	GLXIPC	IPCF to interrupt system connect failed In CPIDI, the PISYS. UUO failed while trying to connect a job to the interrupt system.
IIP	GLXKBD	Illegal Input Pointer CONVBP detected a byte pointer of zero at RD+.RDIOJ.
IJM	QSRADM	Interlocked Job Missing Inconsistency in QUASAR's queue database was detected in KILPSB.
IJW	QSRADM	Interlocked Job Wrong Inconsistency in QUASAR's queue database was detected in KILPSB.
ILM	GLXINT	Illegal memory reference at PC <pc>
IMV	QSRMDA	Invalid MDR/VSL Forward/Backchain Pointers NSTUSR detected that a VSL does not contain a pointer to an MDR. Every VSL should point to an MDR.
IOS	QSRMDA	Invalid Owner Specified in Reassign Message DEASSIGN detected that the job number in the DEASSIGN message does not match the job number in the MDR pointed to by the UCB of the device being deassigned.
IPE	PLEASE	Internal parser error
IPF	PLRTAP	Illegal Positioning Function
IPH	OPRNET	INVALID PROCESS HANDLE TO KILL
IPP	OPRPAR	Invalid PDB Header in Parse Block
IQN	GLXTXT	Illegal qualifier number <number> at <address> An illegal argument qualifier was used in a $TEXT macro.
IRF	GLXIPC	IPCF Reception failure In RCVMSG, the IPCFR. UUO took the error return while trying to receive an IPCF message.
ITD	QSRMDA	Invalid Tape Density Specified for <tape drive name> The density for a tape drive returned in the .STSTS does not match any of the legal densities contained in the UCB for that tape drive.
ITR	QSRMDA	Invalid Tape Resource Number Returned VSLRSN detected that the tape resource number contained in a VSL is zero.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
IVU	QSRMDA	Invalid VOL/UCB Forward/Backchain Pointers DSMACK detected that there is not a pointer P/'to a UCB in a VOL block of a volume that is being dismounted from a device. When a volume is mounted, there should be a pointer in the VOL block to the UCB of the device, and a pointer in the UCB to the VOL block of the volume mounted.
IVV	QSRMDA	Invalid VSL/VOL Forward/Backchain 'Pointers D$FOWN did not detect a pointer in a VOL block back to a VSL when the VOL block was found by a pointer in a VSL. VSL and VOL blocks must be doubly linked.
LDF	LPTSPL	Line Printer Device Status DEVOP. Failed
LEM	CDRIVE	Lousy error message from D60SIN An unidentified error code was returned by D60SIN. S1 = error code.
LGF	PLRTAP	Label Get Failed
LNA	QSRMDA	Logical Name Assignment Failed The DEVLNM UUO failed in REASSI while trying to assign a logical name to a /device.
LNI	SPRINT	LOG not initialized LOGTXT was called to put a character in the log, there is not a LOG page set up for usage.
LRF	PLRTAP	Label Release Failed
MCF	PLRT10	MTAID. UUO Failed
MDS	QSRMEM	MOVING DIFFERENT SIZES QUASAR's queue database is corrupt.
MQE	QSRMDA	Missing QE for a pseudo process D$CHKB detected that there was no QE page address in the MDR for a batch job in the input queue.
MST	OPR	Missing syntax table
MSZ	SPROUT	Message size too large
NAM	QSRMDA	Negative 'A' Matrix Entry Computed The count of an entry in the 'A' matrix went 'P'P/negative. There cannot be a negative number of any physical resource.
NBM	QSRMDA	Negative 'B' Matrix Entry Computed The count of an entry in the 'B' matrix went negative. There cannot be 'a negative number of allocations (claims) for a resource.
NBR	QSRSCH	Nextjob'ing bad request In preparing a "next job" message for an object, NEXTJB called F$RDRQ to find the address of the EQ (external queue) page on disk and the address returned was zero.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
NCM	QSRMDA	Negative 'C' Matrix Entry Computed The count in a 'C' matrix went negative. There cannot be a negative number of owners (sharers) of a device.
NCS	OPRNET	NO CURRENT SERVER DATA BASE IN SKEW
NDE	ORION	Node database empty In DELNOD, it was discovered that the OPR node database is empty. There must be at least a central host node.
NEB	PLRTAP	No Error Bit Routine RETERR was called, but no error bits were lit in P1.
NFB	CDRIVE	FIRST BLOCK IN MESSAGE NOT THE OBJECT BLOCK
NFP	GLXMEM	No free pages M%IPRM was unable to get a free page for 'an IPCF receive.
NFV	PLRT10	No free PS vectors I$PICD detected that there are no free 'interrupt vectors available.
NGF	QSRT10	NECESSARY GETTAB FAILED The GETTAB UUO in DOGTAB failed.
NIP	GLXINT	No interrupt is in progress The DEBRK. UUO took the skip return.
NMF	QSRFSS	NO MORE FILESPACE The failsoft file is full.
NUE	QSRMDA	Null UCB chain encountered In D$INIT, the call to L%FIRST failed to return the first UCB block in the UCB chain.
NUV	QSRDSP	No UCB ptr and No VSL ptr from VOL MDA's database is corrupt.
NVD	PLRT10	No valid density I$PDEN was unable to determine a valid density for reading a tape drive.
NXM	GLXINT	Non-existant memory at PC <pc>
OBR	GLXOTS	Obsolete routine executed
ODE	ORION	OPR delete entry error DELOPR detected that the list of "operators" is empty.
ODI	ORION	OPR database inconsistent The call to VALOPR in SPDOPR failed.
ONV	QSRMDA	Offset of New Volume is Invalid In D$VSR, the calculated offset into the block of VOL block pointers in the VSL is negative.
OOR	GLXOTS	OTS only routine executed
OSF	OPR	ORION send failed

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
OTS	GLXFIL	File Open Block is too small
PAF	GLXMEM	Page access check failed While performing function .PAGCA, the PAGE. UUO failed in routine M%IPRC.
PDL	GLXINT	Pushdown list overflow at PC <pc>
PEF	GLXMEM	Page existence check failed The PAGE. UUO failed while performing P'function .PAGCA in PAGFRE.
PIR	GLXIPC	PID Index out of range The system PID index passed to SPID is invalid.
PKF	GLXMEM	Page kill failed The PAGE. UUO failed in KILPAG.
PLM	PULSAR	Previous List TCB has been meddleD
PNR	PULSAR	PULSAR Not Restartable
PQI	QSRT10	Prime Queue is Interlocked In I$OQUE, the FILOP. UUO error return indicates that the master queue is being modified.
PRF	PLRTAP	Positioning Request Failed
PWE	QSRT10	PRIME WRITE ERROR The OUT UUO in I$WRIT took the error return. IO.BKT was not one of the error bits returned using the GETSTS UUO.
QBI	QSRMDA	QUASAR blew it The caller of SNDREC failed to provide a device name for the RECOGNIZE message being sent to PULSAR.
QNR	QUASAR	QUASAR Not restartable
QSF	CDRIVE	Send to QUASAR FAILED
QSF	LPTSPL	Send to QUASAR FAILED
QSF	SPRINT	QUASAR send failed
QSF	SPROUT	Send to QUASAR FAILED
RAR	GLXIPC	Releasing already released IPCF message In C%REL, RCVMDB+MDB.MS contains zero.
RAT	PULSAR	Requesting work for active TCB
RCN	ORION	G$RSDC is negative database confused RSDMSG detected that the resend "retry count" is negative.
RCN	QSRFSS	REQUEST COUNT NEGATIVE
RCO	ORION	G$RSDC off .. does not match list data In RSDMSG, the resend "retry count" indicates that there are more messages to resend, but the list is empty.
RCW	QSRFSS	REBUILD COUNT WRONG

Name	Module	Message and Explanation
		This stopcode indicates problems rebuilding part of the in-core queues from the current section of the failsoft file.
REF	QSRT10	READING END OF FILE A second EOF error return was generated because there is no more data to be read.
REI	ORION	Remembered entry <entry > in list <list > invalid NXTMSG detected an error in its IPCF message database.
RIE	QSRT10	READ I/O ERROR In I$READ, the IN UUO took the error return, and the error was not EOF.
RJM	QSRADM	Requeue job missing QUASAR's object database is corrupt.
RKD	PLRDSK	Running a killed disk TDB
RKM	PLRTAP	Running a killed magtape TDB
RLT	PLRTAP	Failed Reading Label Type
RMB	QSRMDA	Resource Number Missing in 'B' Matrix The unique resource identifier is missing in a 'B' matrix entry.
RMC	QSRMDA	Resource Number Missing in 'C' Matrix The unique resource identifier is missing in a 'C' matrix entry.
RNF	GLXMEM	Received non-existent page M%IPRC detected that a page created by IPCF does not exist.
RNW	GLXMEM	Ridiculous number of words requested The number of words requested is greater than number of words available in M%GMEM.
RPF	PLRT10	Read label Parameters Failed The TAPOP. UUO failed in I$RDLP.
RRF	QSRFSS	Rebuild Routine Failed One of the queue rebuild routines in REBTBL failed.
RSE	PULSAR	Reschedule from exec level
RSF	PLRT10	TAPOP. to Read Statistics Failed
RTS	GLXFIL	Rename block too small
RTT	PLRTAP	Releasing Tape Twice
RUJ	QSRSCH	Releasing Uninterlocked Job In JOBDUN, the ITN of the job and object do not match when trying to release the job-object interlock.
RWE	QSRT10	REDUNDANT WRITE ERROR In I$WRIT, the OUT UUO took the error

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
		return. IO.BKT is not the error when writing redundant queue.
RZP	GLXMEM	Request for zero pages
SCE	QSRMDA	Structure Catalog Entry is Missing
		A known structure in the 'A' matrix was not found in the structure catalog.
SDF	OPR	Setup dialog failed
SF	ORION	Send failure table inconsistent
SFO	OPR	Setup failure by OPR
SFP	GLXSCN	Scanning floating point not implemented
		XCMFLT was called.
SIO	PLRTAP	Switch units with OPEN Label DDB
SLT	PLRT10	Set Label Type failed
SND	PLRTAP	Switch units with non-existent device <device>
SPF	PLRT10	Set label Params Failed
SSR	PLRLBP	Strange Skip Record
STS	OPRPAR	SHARED SWITCH TABLE SIZE OF <table size> TOO SMALL FOR TABLE OF SIZE <table size>
TBI	PLEASE	S%TXTI block incorrect
TDE	OPRPAR	TABLE DELETE ERROR
TFF	GLXKBD	FILOP. OUT failed to terminal
TML	GLXTXT	Too many levels of call
		SAVLVL detected that it was called more than once to save T%TEXT context.
TML	LPTSPL	TOO MANY LOG BUFFERS REQUIRED
		LOGBUF detected that more than ten pages are being used to build LPTSPL's RUN LOG.
TMS	CDRIVE	Too many setups
		CDRIVE was told to start more readers than it can handle.
TMS	LPTSPL	Too many setups
		LPTSPL was told to start more printers than it can handle.
TMS	QSRFSS	TOO MANY SECTIONS
		Corrupt failsoft queue.
TMS	SPROUT	Too many setups
		SPROUT was told to start more plotters than it can handle.
TMT	GLXSCN	Too much text
		The buffer for the command being parsed cannot hold any more text.
TNO	GLXKBD	Terminal never opened
TUF	PLRT10	TAPOP. UUO failed
		The TAPOP. UUO failed while trying to get REELID in I$RDEV.

LIST OF STOP CODES (Cont)

Name	Module	Message and Explanation
UDR	PLRT10	Unknown drive type DSKCHR bits = characteristics bits>
UFI	GLXFIL	Unknown File Information Descriptor F%INFO was called with an invalid argument.
UIR	GLXIPC	Unexpected IPCF interrupt received C%INTR was called on an IPCF interrupt but PSIFLG is zero.
ULS	PLRDSK	Unit parameter list is short
UMS	SPRINT	Unsupported Recording Mode Specified <mode>
UMT	GALGEN	Unrecognized Monitor Type
UNR	GLXOTS	Unimplemented routine executed
URM	SPRINT	Unknown Recording Mode <mode> Error in NEXTJOB Message
USM	QSRT10	Unique stream missing Calls to either L%FIRST or L%NEXT in UNIFST returned FALSE.
VPF	QSRMDA	Volume Pointer Not Found SCNVOL detected that VSL's VOL block(s) links are inconsistent.
VSA	QRSMDA	VSL Address is Missing in a VOL There is no pointer to a VSL in a VOL block found by a pointer from a VSL. VSL and VOL blocks should be doubly linked.
WBL	QSRT10	WRITING BAD LENGTH The block length to be written in I$WRIT is greater than one page (512 words).
WFO	GLXINT	WTO Function <function> Out of range at address <address>
WLT	OPR	Wrong length table entry block
WNF	PULSAR	Waiting TCB not found
WQV	QSRFSS	Wrong version of master queue file
ZTE	OPR	Zero entry in syntax table entry block
ZTS	OPR	Zero tables setup for OPR
ZWR	GLXMEM	Zero words of memory returned --------

Table of Contents

TOPS-20 SYSTEM PROGRAM LIBRARY
The programs in TOPS-20 System Program Library are listed and described in Table 1.

Table 1 TOPS-20 System Program Library

Program	Description
ACTGEN	ACTGEN is an account generator program used to create and install an account validation data base for use by TOPS-20 in validating accounts. It is intended primarily for use by the system manager and operator. Wheel or operator capabilities must be enabled to run ACTGEN. ACTGEN is documented in the DECSYSTEM-20 System Manager's Guide.
BOOT	BOOT is used to load the TOPS-20 monitor from disk into KL10 memory. On normal system startup, BOOT is automatically loaded and started by RSX20F, and will load the TOPS-20 monitor without operator intervention. BOOT is also responsible for dumping KL10 memory after system malfunction, for later analysis. BOOT is documented in the following documents: DECSYSTEM-20 Software Installation Guide DECSYSTEM-20 Operator's Guide
CHECKD	CHECKD checks TOPS-20 disk file structure and bit table for consistency. In the process of checking the directory structure, CHECKD finds all disk space which is in use; this allows CHECKD to compute the disk pages lost. CHECKD can optionally release this lost space. CHECKD can also be used to completely rebuild the disk bit table or to scan the directory structure for a specified disk address. CHECKD may also be used to create new file structures. CHECKD is documented in the DECSYSTEM-20 Operator's Guide.
CHKPNT	CHKPNT has three major functions: 1. Compile account statistics on disk space utilization 2. Set the monitor checkpoint interval 3. Copy system-generated accounting data into the accounting file. CHKPNT is documented in the TOPS-20 Operator's Guide.
CNVDSK	System Utility converts file system to permit archiving files
CREF	CREF takes the modified listing files produced by the language processors and produces a final, printable listing with cross reference tables appended. CREF is documented in the DECSYSTEM-20 User's Guide.
DDT	DDT is a symbolic assembly language debugger. DDT allows up to 8 breakpoints as well as symbolic patching and manipulation of various datatypes.
DLUSER	DLUSR is a program which obtains identifying information about each directory on a system and places it in a file. The program can then use this file to create the same directories later, in the event of a system rebuild. DLUSER is documented in the DECSYSTEM-20 Operator's Guide and DECSYSTEM-20 System Manager's Guide.
DUMPER	DUMPER is a program for saving and restoring disk files using magtape. It is used by operations personnel for file system maintenance, and may be employed by users who wish to keep certain files on magtape and/or transfer them between systems.

Table 1 TOPS-20 System Program Library (Cont)

Program	Description
DX20LD	DX20 Microcode Loader
EDIT	EDIT is a line-oriented editor which is used to create and edit text files. It resembles the TOPS-10 editor SOS in function and command structure.
FE	FE is a utility for file transfers between the TOPS-20 file system and the FILES-11 file system. It handles protocol for the FE device such that FE: can be addressed as a FILES-11 device, usually through 11 PIP.
	CAUTION **The FE device is intended for use only in software development and updating procedures by knowledgeable people. Use without proper caution may produce unpredictable results.**
	FE depends on the existence of the RSX-20F task T20ACP, which should reside on the -11 file system as T20ACP.TSK.
	Use of FE and file conversion procedures are described in the Guide To Using the FE Device, USEFE.MEM.
FILCOM	The FILCOM program compares two files and outputs the differences between them.
	With FILCOM you may compare both ASCII files and binary files. FILCOM compares ASCII files line by line and binary files word by word.
FORMAT	FORMAT provides the mechanism for formatting and/or verifying RP04, RP05, RP06 disk packs that are configured to RH20s. FORMAT produces a pack in the identical format to one that was created using the diagnostic, DDRPI. FORMAT runs during timesharing only, while DDRPI can FORMAT in stand-alone mode only.
GALAXY	GALAXY is the Batch and Spooling Subsystem for the DECsystem-10 and DECSYSTEM-20. GALAXY comprises all the software (excluding operating systems software) necessary to do batch processing and input and output spooling and all queue management and task scheduling required for those functions.
	GALAXY Release 3 consists of the following programs:

Program	What It Does
QUASAR	Central queue manager, task scheduler, and GALAXY system controller
BATCON	Batch job processor
LPTSPL	Lineprinter output spooler (unspooler)
SPRINT	Card reader input stacker/spooler
QUENCH	Timesharing users' interface to the GALAXY system
QMANGR	Interface module for FOROTS, BASIC, etc.

Program	Description
INFO	System Utility for Inter-Program Communication
LINK	LINK is the linking loader for the DECSYSTEM-20. OVRLAY is the overlay handler for the DECSYSTEM-20.
	LINK and OVRLAY are documented in the DECSYSTEM-20 User's Guide and in the DECSYSTEM-20 LINK User's Guide.
MACRO	Symbolic Assembler
MACSYM	Symbol Parameter Files

Table 1 TOPS-20 System Program Library (Cont)

Program	Description
MAIL	MAIL is a program which allows users to send messages to other users. Messages sent by MAIL are stored in the receiver's disk directory so that they may be referenced when convenient. MAIL depends on the programs INFO and MAILER to perform its stated tasks. Also, the program RDMAIL is used by message recipients to read messages. MAIL is documented in the TOPS-20 User's Guide.
MAKDMP	Create DUMP.EXE file for memory system image on system crash.
MAKLIB	MAKLIB is used to update and index .REL files. MAKLIB will insert, delete or replace modules. It is also used to index FORLIB.REL and LIBOL.REL to speed up the loading process. MAKLIB is documented in the DECSYSTEM-20 User's Guide.
MAKRAM	MAKRAM is a program to generate LP20 translation RAM files. MAKRAM commands are described in MAKRAM.HLP.
MAKVFU	MAKVFU is a program to generate LP05 Direct Access Vertical Format files. MAKVFU commands are described in MAKVFU.HLP.
MAPPER	Performance tuning tool
MONSYM	Symbol Parameter Files
MOUNTR	Labeled tape handler
MTBOOT	Tape Bootstrap
OPLEAS	OPLEAS is the program that enables the operator to talk to users running PLEASE. Requests for contact with the operators are queued; thus the user can type a request for operator action and know that the request will be received even if the operator is currently busy. OPLEAS also handles structure and tape mount requests submitted via the EXEC TMOUNT and SMOUNT commands. OPLEAS is documented in the TOPS-20 User's Guide.
PA1050	RA1050 is the TOPS-10 UUO simulator produced from the file PAT.MAC. It gets mapped into the address space of any program that executes a TOPS-10 UUO. Its function is to intercept all TOPS-10 UUOs and simulate them with the appropriate TOPS-20 JSYSs.
PLEASE	PLEASE provides a facility for one user at a time to talk to an operator. Requests for contact with the operator are queued; thus the user can type a request for operator action and know that the request will be received even if the operator is currently busy. PLEASE runs in conjunction with OPLEAS. PLEASE is documented in the TOPS-20 User's Guide.
PTYCON	PTYCON is a pseudoteletype (PTY) controller. It allows a user multiple job control from a single terminal. PTYCON provides the means to converse with a number of subjobs and to control the manner and times when output is received from the subjobs.
REAPER	Disk space maintenance utility
RDMAIL	RDMAIL is a program which allows a user to read the messages which have been sent to him. It always reads the messages from the file MAIL.TXT. RDMAIL is documented in the DECSYSTEM-20 User's Guide.
RMS	Record Management Services for BASIC-PLUS-2, COBOL-74

Table 1 TOPS-20 System Program Library (Cont)

Program	Description
RUNOFF	RUNOFF is a text-processing program. RUNOFF will format input text, generate tables, build lists, handle page and section numbering. RUNOFF allows a user to make all sorts of changes to the text of a document and still produce a clean, well-formatted result. RUNOFF is documented in Getting Started with Runoff.
SETSPD	SETSPD is a privileged system program which processes the 3-CONFIG.CMD file and, in so doing, sets many initial parameters about the system such as initial line speeds, system logical names, and magtape logical to physical correspondences.
SPEAR	SPEAR is a library of functions that sorts, evaluates, and reports on events recorded in the local system event file.
SYSJOB	SYSJOB is a program for controlling system background programs. It is normally started only by job 0, and it creates additional processes and jobs as necessary. An operator or other privileged job may pass commands to SYSJOB via an exec command (↑E) SPEAK to affect the status of the background programs. SYSJOB is documented in the DECSYSTEM-20 Operator's Guide under the (↑E) SPEAK command.
TGHA	MF20 on-line diagnostic/utility
TV	Video Text Editor
UETP	User Environment Test Package
ULIST	ULIST provides a mechanism for listing user and directory information. The listing may be directed to the printer, the user's terminal, or to a file. ULIST will provide information on user and directory groups, directory numbers, quotas, and protections, and will list user passwords if desired.
VERIFY	Installation verifier
WATCH	WATCH is a system program which provides a list of various system statistics and job run times upon request. A user can thus periodically check system performance with this utility.

The following unsupported software (binary and source) is distributed with TOPS-20 and is provided on an "as is" basis without DIGITAL warranty express or implied.

USAG20	Accounting Utility for USAGE.OUT
USAH20	Requires FORTRAN/COBOL/SORT License
SYSDPY	System Performance Tool
DDT11	Debugging tool for FE communication software
SED	Screen Editor
ALU	Source Control Utility
REDIT	Source Edit Utility
REV	File Manipulation Utility
BLIS10	BLISS-10 Compiler
FEDDT	Debugger for front-end dumps

TOPS-20 COMMAND LANGUAGE

The TOPS-20 Operating System supports approximately 70 basic commands. These commands are described in Table 3.

Special symbols and control characters used by TOPS-20 are described in Table 2.

COMMAND FORMAT

TOPS-20 commands use the following format.

COMMAND$(guide word)ARG$(guide word)ARG$(...<CR>

The base command and each argument is delimited by an altmode (ESCAPE KEY). The command string is terminated by a carriage return <CR>.

ERROR MESSAGES

Table 3 lists and describes many of the most commonly used BUGCHKS and BUGHLTS associated with a TOPS-20 operating system. The list was taken from TOPS-20 BIG SYSTEM, TOPS-20 MONITOR 3A (2013). A complete list for any given TOPS-20 operating system may be printed by typing

PRINT PS:<SYSTEM>BUGSTRING.TXT<CR>

Table 2 TOPS-20 Symbols and Control Characters

Character	Description
↑C↑C	Two control C characters will return the terminal to monitor command level.
@	Prompt - A single @ sign indicates the monitor is at command level and ready to accept commands.
,<CR>	A command and carriage return typed following a command name causes the monitor to enter subcommand level for the command named.
@@	Prompt - A double @@ sign indicates the monitor is at a subcommand level and ready to accept subcommands only.
<CR>	A single carriage return terminates a command or subcommand.
<CR><CR>	A double carriage return terminates a subcommand and returns the monitor to command level.
?	A question mark typed at the command level or subcommand level will cause the monitor to print a list of the available commands. A question mark typed following a partially typed command will cause the monitor to print a list of all commands or subcommands which begin with the characters typed. A question mark typed following a guide word will cause the monitor to print a list of the possible arguments. A question mark printed by the monitor indicates the user has made an error in typing a command.
$ (altmode) (ESCAPE)	If there is no ambiguity in a partially typed command, pressing the ESCAPE key will cause the remaining characters and the first guide word of the command to be printed. If a partially typed command is ambiguous pressing the ESCAPE key will cause the terminal bell to ring. The ESCAPE key is also used to terminate an argument and causes the next guide word to be printed.

Table 2 TOPS-20 Symbols and Control Characters (Cont)

Character	Description
RUBOUT DELETE	The RUBOUT or DELETE key will cause the last DELETE character typed to be deleted.
↑W	Typing a control W will cause the last field typed to be deleted.
↑U	Typing a control U will cause the entire command line to be deleted.
↑R	Typing a control R will cause the current command line to be reprinted.
↑O	Typing a control O will stop the current printout.
!	The exclamation mark is used to delimit text following a command. This is useful for sending messages during a KLINIK linkup.

Table 3 TOPS-20 Command Summary

Command	Description
System Access Commands	
ATTACH	Connects your terminal to a designated job. See also: DETACH, UNATTACH.
DETACH	Disconnects your terminal from the current job without affecting the job. See also: ATTACH, UNATTACH.
DISABLE	Returns a privileged user to normal status. See also: ENABLE.
ENABLE	Permits privileged users to access and change confidential system information. See also: DISABLE.
LOGIN	Gains access to the TOPS-20 system. See also: LOGOUT.
LOGOUT	Relinquishes access to the TOPS-20 system. See also: LOGIN.
UNATTACH	Disconnects a terminal from a job; it does not have to be the terminal you are using. See also: ATTACH, DETACH.
Information Commands	
DAYTIME	Prints the current date and time of day.
INFORMATION	Provides information about your job, files, memory, errors, system status, and many other parameters.
SYSTAT	Outputs a summary of system users and available computing resources.
Terminal Commands	
ADVISE	Sends whatever you type on your terminal as input to a job connected to another terminal. See also: BREAK, RECEIVE, REFUSE, TALK.
BREAK	Clears terminal links and advising links. See also: ADVISE, RECEIVE, REFUSE, TALK.
RECEIVE	Allows your terminal to receive links and advice from other users. See also: ADVISE, BREAK, REFUSE, TALK.
REFUSE	Denies links and advice to your terminal. See also: ADVISE, BREAK, RECEIVE, TALK
SET	Declares certain action to be taken when errors are detected in TOPS-20 commands.

Table 3 TOPS-20 Command Summary (Cont)

Command	Description
TAKE	Accepts commands from a file, just as if you had typed its contents on your terminal.
TALK	Links two terminals so that each user can observe what the other user is doing, yet does not affect the other user's job. See also: ADVISE, BREAK, RECEIVE, REFUSE.
TERMINAL	Declares the hardware type of terminal you have, and lets you inform TOPS-20 of any special characteristics of the terminal.
Device Handling Commands	
ASSIGN	Reserves a device for use by your job. See also: DEASSIGN, DEFINE.
BACKSPACE	Moves a magnetic tape drive back any number of records or files. See also: REWIND, SKIP, UNLOAD.
DEASSIGN	Releases a previously assigned device. See also: ASSIGN.
EOF	Writes an end-of-file mark on a magnetic tape.
REWIND	Positions a magnetic tape backward to its load point. See also: BACKSPACE, SKIP, UNLOAD.
SKIP	Advances a magnetic tape one or more records or files. See also: BACKSPACE, REWIND, UNLOAD.
UNLOAD	Rewinds a magnetic tape until the tape is wound completely on the source reel. See also: BACKSPACE, SKIP, REWIND.
File Systems Commands	
ACCESS	Grants ownership and group rights to a specified directory. See also: CONNECT, END-ACCESS
APPEND	Adds information from one or more source files to an existing disk file. See also: EDIT.
CLOSE	Closes a file or files left open by a program.
CONNECT	Removes you from your current directory and connects you to a specified directory.
COPY	Duplicates a source file in a destination file.
CREATE	Starts EDIT for making a new file. See also: EDIT.
DELETE	Marks the specified file(s) for eventual deletion (disk files only) or deletes the specified files (all other devices). See also: EXPUNGE, UNDELETE.
DEFINE	Associates a logical name with one or more file names. See also: ASSIGN.
DIRECTORY	Lists the names of files residing in the specified directory and information relating to those files. See also: FDIRECTORY, TDIRECTORY, VDIRECTORY.
EDIT	Starts EDIT for changing an existing file. See also: APPEND, CREATE.
EXPUNGE	Permanently removes any deleted files from the disk. See also: DELETE, UNDELETE.

Table 3 TOPS-20 Command Summary (Cont)

Command	Description
END-ACCESS	Relinquishes ownership rights to a specified directory. See also: ACCESS
FDIRECTORY	Lists all the information about a file or files. See also: DIRECTORY, TDIRECTORY, VDIRECTORY
LIST	Prints one or more files on the line printer with or without formatting. See also: PRINT, TYPE
PRINT	Lists one or more files on the line printer. See also: LIST, TYPE
QUEUE	Places an entry into or examines a specified queue, for example, the line printer output queue.
RENAME	Changes one or more descriptors of an existing file specification.
SDISMOUNT	Notifies the system that the given structure is no longer needed. See also: SMOUNT, SREMOVE
TDIRECTORY	Lists the names of all files in the order of the date and time they were last written. See also: DIRECTORY, FDIRECTORY, VDIRECTORY
SMOUNT	Requests that a structure be made available to the user. See also: SDISMOUNT, SREMOVE.
TYPE	Types the specified files on your terminal. See also: PRINT, LIST.
SREMOVE	Makes a structure unavailable and requests its removal. See also: SDISMOUNT, SMOUNT.
UNDELETE	Restores one or more disk files marked for deletion. See also: DELETE, EXPUNGE.
TMOUNT	Requests that a magnetic tape be made available to the user.
VDIRECTORY	Lists the names of all files, as well as their protection, size, and date and time they were last written. See also: DIRECTORY, FDIRECTORY, TDIRECTORY.
Program Control Commands	
COMPILE	Translates a source program using the appropriate compiler. See also: DEBUG, EXECUTE, LOAD, MERGE.
CONTINUE	Resumes execution of a program interrupted by a control C. See also: REENTER, START.
CREF	Runs the CREF program which produces a cross-reference listing and automatically sends it to the line printer.
CSAVE	Saves the program currently in memory so that it may be used by giving a RUN command. The program is saved in a compressed format. See also: SAVE.
DDT	Merges the debugging program, DDT, with the current program and then starts DDT. See also: DEBUG, MERGE.
DEBUG	Takes a source program, compiles it, loads it with DDT and starts DDT. See also: COMPILE, DDT, MERGE.

Table 3 TOPS-20 Command Summary (Cont)

Command	Description
EXECUTE	Translates, loads, and begins execution of a program. See also: COMPILE, LOAD
FORK	Makes the TOPS-20 language work for a particular address space.
GET	Loads an executable program from the specified file. See also: LOAD
LOAD	Translates a program and loads it into memory. See also: EXECUTE
MERGE	Loads an executable program into memory and merges it with the current contents of memory. See also: DEBUG
POP	Stops a copy of the TOPS-20 Command Language and returns control to the previous copy of the Command Language. See also: PUSH
PUSH	Starts a new copy of the TOPS-20 Command Language. See also: POP
R	Runs a system program. See also: EXECUTE, GET, LOAD, RUN, START
REENTER	Starts the program currently in memory at an alternate entry point specified by the program. See also: CONTINUE, START
RESET	Clears the job to which your terminal is currently attached.
RUN	Loads an executable program from a file and starts it at the location specified in the program. See also: EXECUTE, GET, LOAD, START
SAVE	Copies the contents of memory into a file in executable format. If memory contains a program, you may now execute the program by giving the RUN command with the proper file specification. See also: CSAVE
START	Begins execution of a program at the location specified in the entry vector. See also: CONTINUE, EXECUTE, GET, LOAD REENTER
Batch Commands	
SUBMIT	Enters a file into the Batch waiting list. When it is your job's turn, the commands contained in the file are executed.

Table 4 lists the BUGHLT and BUGCHK codes. For more information refer to TOPS-20 BUGHLT Documentation of the TOPS-20 Software Notebooks.

Table 4 BUGHLT and BUGCHK Names and Descriptions

Name	Type	Description
ABKSKD	[HLT]	ADDRESS BREAK FROM SCHEDULER CONTEXT
ACTBBD	[CHK]	VERACT - ACCOUNT FILE CORRUPTED
APRAPE	[HLT]	ADDRESS PARITY ERROR
APRNX1	[HLT]	NXM DETECTED BY APR
APRNX2	[HLT]	NXM DETECTED BY APR
ARCASS	[CHK]	ARCF: File dir # and mapped dir # do not match
ARCVER	[CHK]	ARCMSG: NOUT failed
ARSTXX	[HLT]	ARRST: FDB disappeared for destination file
ASAASG	[CHK]	DSKASA - ASSIGNING ALREADY ASSIGNED DISK ADDRESS
ASGBAD	[CHK]	DSKASA - ASSIGNING BAD DISK ADDRESS
ASGBPG	[CHK]	INIBTB-FAILED TO ASSIGN BAD PAGE(S)
ASGFR0	[HLT]	ASGFRE - ILLEGAL TO ASSIGN 0 FREE SPACE
ASGINT	[CHK]	ASGFRE CALLED OKINT
ASGREP	[CHK]	ILLEGAL PRIORITY GIVEN TO ASGRES
ASGREQ	[CHK]	ILLEGAL POOL NUMBER GIVEN TO ASGRES
ASGSW2	[HLT]	SWPOMG-CANNOT ASSIGN RESERVED DRUM ADDRESS
ASGSWB	[CHK]	SWPINI-CANNOT ASSIGN BAD ADDRESS
ASOFNF	[HLT]	DELFIL: ASGOFN GAVE FAIL RETURN FOR LONG FILE XB
ASTJFN	[HLT]	GETFDB: CALLED FOR JFN WITH OUTPUT STARS
BADBAK	[CHK]	FILIN2 - BACKUP COPY OF ROOT DIRECTORY IS NOT GOOD
BADBAT	[CHK]	BAT BLOCKS UNREADABLE
BADDAC	[HLT]	INSACT - NULL ACCOUNT STRING SEEN
BADIDX	[CHK]	IDXINI: PARTIALLY UNSUCCESSFUL INDEX TABLE REBUILD
BADPTR	[HLT]	BAD SECTION POINTER - SECMAP
BADREC	[HLT]	FILINI - Reconstruction of ROOT-DIRECTORY failed
BADROT	[HLT]	FILIN2: ROOT-DIRECTORY IS INVALID
BADTAB	[CHK]	VERACT - SPURIOUS HASH TABLE ENCOUNTERED
BADTTY	[HLT]	TRANSFER TO NONEXISTENT TTY CODE
BADTYP	[HLT]	BAD LABEL FIELD DESC
BADXT1	[HLT]	INDEX TABLE MISSING AND CAN NOT BE CREATED
BADXT2	[CHK]	INDEX TABLE MISSING AND WAS CREATED
BADXTB	[HLT]	FILIN2: Could not initialize index table
BKUPDF	[HLT]	BKUPD - BAD CST1 ENTRY OR INCONSISTENT CST
BLKF1	[CHK]	BYTINA: BLKF SET BEFORE CALLING SERVICE ROUTINE
BLKF2	[CHK]	BYTOUA: BLKF SET BEFORE CALL TO SERVICE ROUTINE
BLKF3	[CHK]	CLZDO: BLKF SET BEFORE CALL TO SERVICE ROUTINE
BLKF4	[CHK]	.GDSTS: BLKF SET BEFORE CALL TO DEVICE ROUTINE
BLKF5	[CHK]	.MTOPR: BLKF SET BEFORE CALL TO DEVICE ROUTINE
BLKF6	[CHK]	.SDSTS: BLKF SET BEFORE CALL TO DEVICE ROUTINE
BOOTCR	[HLT]	GETSWM - NOT ENOUGH CORE FOR SWPMON
BOOTER	[HLT]	GETSWM - ERROR LOADING SWPMON
BOOTLK	[HLT]	GSMDSK - FAILED TO LOCK NEEDED PAGES
BOOTMP	[HLT]	GSMDSK - CANNOT MAP BOOTSTRAP PAGES
BTBCR1	[HLT]	FILINI - NO BIT TABLE FILE AND UNABLE TO CREATE ONE
BTBCRT	[HLT]	FILINI - COULD NOT INITIALIZE BIT TABLE FOR PUBLIC STRUCTURE
CDILVT	[HLT]	ILLEGAL DEVICE FUNCTION CODE
CKPLEN	[CHK]	USGINI - ILLEGAL CHECKPOINT ENTRY LENGTH
CLZABF	[CHK]	CLZFFW: SERVICE ROUTINE BLOCKED ON AN ABORT CLOSE
CRDBAK	[CHK]	CRDIR3: COULD NOT MAKE BACKUP COPY OF ROOT-DIRECTORY
CRDBK1	[CHK]	CRDIR4: COULD NOT MAKE BACKUP COPY OF ROOT-DIRECTORY
CRDNOM	[CHK]	CRDIR-FAILED TO MAKE MAIL.TXT FILE
CRDOLD	[CHK]	CRGDGB: OLD FORMAT CRDIR IS ILLEGAL
CRDSDF	[CHK]	CRDIR1: SETDIR FAILED ON NEW DIRECTORY
CRSPAG	[CHK]	VERACT - ACCOUNT DATA BLOCK CROSSES A PAGE BOUNDARY
CST2I1	[HLT]	PAGE TABLE CORE POINTER AND CST2 FAIL TO CORRESPOND
CST2I2	[HLT]	MVPT-CST2 INCONSISTENT
CST2I3	[HLT]	PAGE TABLE CORE POINTER AND CST2 FAIL TO CORRESPOND
DEABAD	[CHK]	DSKDEA - DEASSIGNING BAD DISK ADDRESS
DEAUNA	[CHK]	DEDSK-DEASSIGNING UNASSIGNED DISK ADDRESS
DELBDD	[INF]	DELDIR: BAD DIRECTORY DELETED. REBUILD BIT TABLE
DEVUCF	[CHK]	DEVAV - UNEXPECTED CHKDES FAILURE
DGUTPG	[HLT]	DIAG - LOCKED PAGE LIST PAGE LOCKED AT DIAG UNLOCK
DGZTPA	[HLT]	DIAG - LOCKED PAGE LIST PAGE WAS ZERO
DIRACT	[CHK]	ACTBAD: ILLEGAL FORMAT FOR DIRECTORY ACCOUNT BLOCK IN DIRECTORY:
DIRB2L	[CHK]	RLDFB2: DIRECTORY FREE BLOCK TOO LARGE IN DIRECTORY:
DIRB2S	[CHK]	RLDFB1: DIRECTORY FREE BLOCK TOO SMALL IN DIRECTORY:
DIRBAD	[CHK]	SETDI4: SMASHED DIRECTORY NUMBER:
DIRBAF	[CHK]	RLDFB5: BLOCK ALREADY ON DIRECTORY FREE LIST IN DIRECTORY:
DIRBCB	[CHK]	RLDFB3: DIRECTORY FREE BLOCK CROSSES PAGE BOUNDARY IN DIRECTORY:
DIRBLK	[CHK]	BLKSCN: ILLEGAL BLOCK TYPE IN DIRECTORY:
DIRDNL	[CHK]	ULKDIR-DIRECTORY NOT LOCKED, DIRECTORY NUMBER:
DIREXT	[CHK]	EXTBAD: ILLEGAL FORMAT FOR DIRECTORY EXTENSION BLOCK IN DIRECTORY:
DIRFDB	[CHK]	ILLEGAL FORMAT FOR FDB IN DIRECTORY:
DIRFKP	[CHK]	SETDIR-DIR PAGE 0 BELONGS TO FORK IN DIRECTORY:

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
DIRFRE	[CHK]	FREBAD: ILLEGAL FORMAT FOR DIRECTORY FREE BLOCK IN DIRECTORY:
DIRIFB	[CHK]	RLDFB4: ILLEGAL BLOCK TYPE ON DIRECTORY FREE LIST IN DIRECTORY:
DIRNAM	[CHK]	NAMBAD: ILLEGAL FORMAT FOR DIRECTORY NAME BLOCK IN DIRECTORY:
DIRPG0	[CHK]	DR0CHK: ILLEGAL FORMAT FOR DIRECTORY PAGE 0 IN DIRECTORY:
DIRPG1	[CHK]	DRHCHK: DIRECTORY HEADER BLOCK IS BAD IN DIRECTORY:
DIRRHB	[CHK]	RLDFB6: ATTEMPTING TO RETURN A HEADER BLOCK IN DIRECTORY:
DIRSY1	[CHK]	DELDL8: DIRECTORY SYMBOL TABLE FOULED UP FOR DIRECTORY:
DIRSY2	[CHK]	MDDNAM: SYMBOL TABLE FOULED UP IN DIRECTORY:
DIRSY3	[CHK]	LOOKUP: SYMBOL SEARCH FOULED UP IN DIRECTORY:
DIRSY4	[CHK]	NAMCM4: DIRECTORY SYMBOL TABLE FOULED UP IN DIRECTORY:
DIRSY5	[CHK]	SYMBAD: ILLEGAL FORMAT FOR DIRECTORY SYMBOL TABLE IN DIRECTORY:
DIRSY6	[CHK]	RBLDST: PREMATURELY RAN OUT OF ROOM IN SYMBOL TABLE IN DIRECTORY:
DIRULK	[CHK]	ULKMD2: ATTEMPT TO UNLOCK ILLEGALLY FORMATTED DIR, DIR NUMBER:
DIRUNS	[CHK]	UNSBAD: ILLEGAL FORMAT FOR DIRECTORY USER NAME BLOCK IN DIRECTORY:
DLDEF	[INF]	LOGICAL NAME DEFINE FAILED FOR FE CTY
DN20ST	[INF]	DTESRV- DN20 STOPPED
DRMFUL	[CHK]	DRUM COMPLETELY FULL
DRMIBT	[HLT]	DRMASN-BIT TABLE INCONSISTENT
DRMNFR	[HLT]	DRMAM-CANNOT FIND PAGE WHEN DRMFRE NON-0
DSKBT1	[CHK]	DSK BIT TABLE FOULED, CANNOT FIND FREE PAGE ON TRACK WITH NON-0 COUNT
DSKBT3	[CHK]	DISK BIT TABLE ALREADY LOCKED AT LCKBTB
DST2SM	[HLT]	SWPINI-DST TOO SMALL
DTECAR	[CHK]	CARRIER FNC WITH NO LINE NUMBER
DTECDM	[INF]	DTESRV- TO -10 COUNTS DO Not MATCH
DTEDAT	[CHK]	TAKTOD- ILLEGAL FORMAT FOR TIME/DATE
DTEDEV	[CHK]	ILLEGAL DEVICE
DTEDIN	[INF]	DTESRV- TO -10 IN PROGRESS ON DOORBELL
DTEDME	[INF]	DTESRV- ZERO Q COUNT
DTEERR	[CHK]	DTESRV-DTE DEVICE ERROR
DTEIDP	[CHK]	BAD INDIRECT PACKET
DTEIFR	[CHK]	DTESRV-ILLEGAL FNC REQUEST
DTELPI	[INF]	DTECHK- DTE LOST PI ASSIGNMENT
DTEMCC	[CHK]	DOFRGM-DN20 DISAGRESS WITH COUNT
DTEODD	[CHK]	TAKLC-ODD BYTE COUNT FOR LINE CHARACTERS
DTEP2S	[CHK]	TO10DN-PACKET TOO SMALL
DTEPGF	[CHK]	DTE TRANSFER PAGE FAIL
DTEPNR	[INF]	DTESRV- INCORRECT INDIRECT SETUP
DTETIP	[CHK]	DTETDN-TO10 DONE RECEIVED WITH NO TRANSFER IN PROGRESS
DTETTY	[CHK]	NON-TTY DEVICE
DTEUIF	[CHK]	DTESRV-UNIMPLEMENTED FUNCTION FROM 11
DVCHRX	[CHK]	DVCHR1 - UNEXPECTED CHKDES FAILURE WITHIN .DVCHR
DX2DIE	[CHK]	PHYX2 - DX20 HALTED
DX2DNF	[INF]	PHYX2 - DRIVE NUMBER NOT FOUND IN UDBS
DX2FGS	[CHK]	PHYX2 - FAIL TO GET SENSE BYTES
DX2FUS	[CHK]	PHYX2 - FAIL TO UPDATE SENSE BYTES
DX2HLT	[INF]	PHYX2 - DX20 HALTED
DX2IDM	[CHK]	PHYX2 - ILLEGAL DATA MODE AT DONE INT
DX2IDX	[INF]	PHYX2 - ILLEGAL RETRY BYTE POINTER
DX2IEC	[CHK]	PHYX2 - ILLEGAL ERROR CLASS CODE
DX2IFS	[CHK]	PHYX2 - ILLEGAL FUNCTION AT START IO
DX2IRF	[INF]	PHYX2 - ILLEGAL FUNCTION DURING RETRY
DX2MCF	[CHK]	PHYX2 - DX20 MICROCODE CHECK FAILURE
DX2N2S	[INF]	PHYX2 - MORE TU70S THAN TABLE SPACE, EXCESS IGNORED
DX2NRT	[CHK]	DX2ERR - IS.NRT SET ON SUCCESSFUL RETRY
DX2NUD	[CHK]	PHYX2 - CHANNEL DONE INTERRUPT BUT NO UNIT ACTIVE
DX2NUE	[CHK]	PHYX2 - NO ACTIVE UDB AND DX20 COMPOSITE ERROR SET
DX2RFU	[CHK]	PHYX2 - ERROR RECOVERY CONFUSED
DX2UNA	[INF]	PHYX2 - ATTENTION INTERRUPT AND UDB NOT ACTIVE
DX2UPE	[CHK]	PHYX2 - FAIL TO UPDATE SENSE BYTES DURING INITIALIZATION
DXBASD	[CHK]	PHYP2 - ASYCHRONOUS STATUS FROM NON-POSITIONING DRIVE
DXBDIE	[CHK]	PHYP2 - DX20B MICROCODE HALTED
DXBDMI	[CHK]	PHYP2 - DX20B MICROCODE IS INVALID
DXBEUI	[CHK]	PHYP2 - ERROR TRYING TO INITIALIZE A UNIT
DXBEWC	[CHK]	PHYP2 - ERROR PRESENT WHEN CONNECTING TO A UNIT
DXBFEX	[HLT]	PHYP2 - ILLEGAL FUNCTION STARTING IO
DXBFGS	[CHK]	PHYP2 - FAILED TO GET SENSE BYTES
DXBFUS	[CHK]	PHYP2 - FAILED TO UPDATE SENSE BYTES
DXBHLT	[INF]	PHYP2 - DX20B CONTROLLER HALTED
DXBIEC	[CHK]	PHYP2 - UNKNOWN ERROR CODE FROM DX20
DXBIF2	[HLT]	PHYP2 - ILLEGAL FUNCTION STACKING IO
DXBILF	[HLT]	PHYP2 - ILLEGAL FUNCTION AT DONE INTERRUPT
DXBLTF	[HLT]	PHYP2 - LATENCY OPTIMIZATION FAILURE
DXBMSR	[HLT]	PHYP2 - MULTIPLE SECTORS INDICATED IN ECC RECOVERY
DXBNUD	[CHK]	PHYP2 - NO UNIT ACTIVE FOR DONE INTERRUPT
DXBTNF	[HLT]	PHYP2 - UNIT TYPE NOT FOUND IN TABLE
DXBTTS	[CHK]	PHYP2 - TABLES TOO SMALL FOR THIS MANY DRIVES
DXBUA1	[CHK]	PHYP2 - DONE INTERRUPT AND UNIT WAS NOT ACTIVE
DXBUNA	[CHK]	PHYP2 - ATTENTION INTERRUPT AND UNIT WAS NOT ACTIVE

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
DXBZEC	[CHK]	PHYP2 - ZERO ECC BYTE RETURNED
EPTMPE	[HLT]	PFCDPE: PARITY ERROR OCCURRED IN THE EPT
EXILGO	[CHK]	EXECI - Interrupt during login or logout
EXPAFK	[HLT]	EXPALL: JOB 0 CFORK FAILED
EXPRCD	[CHK]	EXPALL: RCDIR FAILURE
FATCDP	[HLT]	FATAL CACHE DIRECTORY PARITY ERROR
FATMER	[HLT]	FATAL MEMORY ERROR
FEBAD	[CHK]	FEHSD-WRONG FE
FEBFOV	[CHK]	FEHSD-BUFFER OVERFLOW
FEOCPB	[CHK]	FEFSYS - FAILED TO BACKUP ROOT-DIRECTORY
FEUSTS	[CHK]	FESSTS-UNKNOWN STATUS
FILBAK	[CHK]	FILCRD: COULD NOT CREATE BACKUP OF ROOT-DIR
FILBOT	[CHK]	COULD NOT CREATE BOOTSTRAP.BIN FILE
FILBTB	[HLT]	UNABLE TO WRITE BIT TABLE FILE
FILCCD	[CHK]	Could not create directory
FILFEF	[CHK]	Could not create Front End File System
FILHOM	[CHK]	UNABLE TO REWRITE HOME BLOCKS IN WRTBTB
FILIRD	[HLT]	FILINW: COULD NOT INITIALIZE THE ROOT DIRECTORY
FILJB1	[CHK]	FILCRD: No room to create standard system directories
FILMAP	[HLT]	FILIN2: COULD NOT MAP IN ROOT-DIRECTORY
FILRID	[HLT]	FILINW: INDEX TABLE ALREADY SET UP FOR ROOT DIR
FIXBAD	[CHK]	Could not re-write Home Blocks to point to FE Filesystem
FIXBDB	[CHK]	COULD NOT RE-WRITE HOME BLOCKS TO POINT TO BOOTSTRAP.BIN
FKCTNZ	[CHK]	FORK LOCK NEST COUNT NON-ZERO
FKWSP1	[CHK]	LOADBS-UNREASONABLE FKWSP
FLKNS	[CHK]	FUNLK-LOCK NOT SET
FLKTIM	[CHK]	FLOCK-FORK LOCK TIMEOUT
FRKBAL	[CHK]	AGESET-FORK NOT IN BALSET
FRKNDL	[CHK]	FORK NOT PROPERLY DELETED
FRKPTE	[HLT]	BADCPG-FATAL ERROR IN FORK PT PAGE
FRKSLF	[HLT]	SUSFK - GIVEN SELF AS ARG
GIVTMR	[INF]	GIVOK TIMEOUT
GLFNF	[HLT]	GLREM - FORK NOT FOUND
GTFDB1	[CHK]	DSKINS: GETFDB FAILURE.
GTFDB2	[HLT]	NEWLFP: GETFDB FAILURE FOR OPEN FILE.
GTFDB3	[HLT]	DSKREN-GETFDB FAILURE FOR OPEN FILE
GTFDB6	[HLT]	CRDIOA: CANNOT DO GETFDB ON ROOT-DIRECTORY
HARDCE	[CHK]	HARD CACHE ERRORS--CACHE DESELECTED
HPSCHK	[CHK]	SCHEDULER - EXCESSIVE TIME IN HIGH PRIORITY
HSHERR	[CHK]	VERACT - HASH VALUE OUT OF RANGE
HSYFRK	[HLT]	HSYS-JOB 0 CFORK FAILED
IBCPYW	[HLT]	COPY-WRITE POINTER IN INDEX BLOCK
IBOFNF	[HLT]	FILINI: ASOFN FAILURE FOR ROOT DIRECTORY IB
IDXNOS	[HLT]	FILINI - COULD NOT ASSIGN FREE SPACE FOR IDXTAB
ILAGE	[HLT]	BAD AGE FIELD IN CST0
ILBOOT	[HLT]	GETSWM-ILLEGAL VALUE OF BOOTFL
ILCHS1	[HLT]	PHYSIO - ILLEGAL CHANNEL STATUS AT SIO
ILCHS2	[HLT]	PHYSIO - ILLEGAL CHANNEL STATE AT STKIO
ILCNSP	[HLT]	PHYSIO - ILLEGAL CALL TO CONSPW
ILCNST	[HLT]	PHYSIO - ILLEGAL CALL TO CONSTW
ILCST1	[HLT]	ILLEGAL ADDRESS IN CST1 ENTRY, CANNOT RESTART
ILDEST	[HLT]	ILLEGAL DESTINATION IDENTIFIER TO SETMPG OR SETPT
ILDRA1	[CHK]	DASDRM-ILLEGAL OR UNASSIGNED DRUM ADDRESS
ILDRA2	[HLT]	DRMIAD-ILLEGAL DRUM ADDRESS
ILFPTE	[HLT]	ILLFPT: ILLEGAL SECTION NUMBER REFERENCED
ILGDA1	[HLT]	GDSTX - BAD ADDRESS
ILGDA2	[HLT]	GDSTX - BAD ADDRESS
ILGOKM	[CHK]	ILLEGAL FUNCTION FOR GETOKM CALL
ILIBPT	[CHK]	BAD POINTER TYPE IN INDEX BLOCK
ILIRBL	[HLT]	PHYSIO - IORB LINK NOT NULL AT ONFPWQ
ILJRFN	[CHK]	JFKRFH - BAD JRFN, IGNORED
ILLDMS	[CHK]	BADDMS: ILLEGAL DMS JSYS FROM MONITOR CONTEXT
ILLFLT	[CHK]	KA10 FLT PT INSTRUCTION IN MONITOR
ILLGO	[HLT]	INVALID CHANNEL LOGOUT
ILLIND	[HLT]	ILLEGAL INDIRECT
ILLTAB	[CHK]	TABLK2: TABLE NOT IN PROPER FORMAT
ILLUUO	[HLT]	KIBADU: ILLEGAL UUO FROM MONITOR CONTEXT
ILMNRF	[HLT]	ILLEGAL REFERENCE TO MON ADR SPACE
ILOFN1	[HLT]	MSCANP-ILLEG IDENT
ILOKSK	[HLT]	OKSKED EXECUTED WHEN NOT NOSKED
ILPAG1	[HLT]	SWPOTO-INVALID PAGE
ILPAGN	[HLT]	MRKMPG-INVALID PAGE NUMBER
ILPDAR	[HLT]	PHYSIO - ILLEGAL DISK ADDRESS IN PAGEM REQUEST
ILPID1	[CHK]	CREPID: ATTEMPT TO CREATE ILLEGAL PID
ILPID2	[CHK]	DELPID: VALIDATED PID TURNED ILLEGAL
ILPLK1	[HLT]	MLKPG-ILLEGAL ARGS
ILPPT1	[HLT]	UPDOFN-BAD POINTER IN PAGE TABLE
ILPPT3	[HLT]	BAD POINTER IN PAGE TABLE
ILPSEC	[HLT]	ILLEGAL SECTION NUMBER

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
ILPTN1	[HLT]	MRPACS-ILLEG PTN
ILRBLT	[HLT]	PHYSIO - IORB LINK NOT NULL AT ONF/STWQ
ILRFPD	[HLT]	PDL-OV IN ILLEGAL PAGE REFERENCE
ILSPTH	[HLT]	SETPT-SPTH INCONSISTENT WITH XB
ILSPTI	[HLT]	ILLEGAL SPT INDEX GIVEN TO SETMXB
ILSRC	[HLT]	ILLEGAL SOURCE IDENTIFIER GIVEN TO SETPT
ILSWPA	[HLT]	SWPIN - ILLEGAL SWAP ADDRESS
ILTWQ	[HLT]	PHYINT - TWQ OR PWQ INCORRECT
ILTWQP	[HLT]	PHYSIO - PWQ OR TWQ TAIL POINTER INCORRECT
ILULK1	[HLT]	MULKPG - TRIED TO UNLOCK PAGE NOT LOCKED
ILULK2	[HLT]	TRIED TO UNLOCK PAGE NOT LOCKED
ILULK3	[HLT]	MULKMP - ILLEGAL MONITOR ADDRESS
ILULK4	[HLT]	MULKCR - ILLEGAL CORE PAGE NUMBER
ILUST1	[HLT]	PHYSIO - UNIT STATUS INCONSISTENT AT SIO
ILUST2	[CHK]	PHYSIO - UNIT STATUS INCONSISTENT AT SPS
ILUST3	[HLT]	PHYSIO - SCHSEK - IMPOSSIBLE UNIT STATUS
ILUST4	[HLT]	PHYSIO - CONTROLLER ACTIVE AT SPS
ILUST5	[HLT]	PHYSIO - ILLEGAL CHANNEL OR CONTROLLER STATE AT STKIO
ILUST6	[HLT]	PHYSIO - ILLEGAL UNIT STATE AT STKIO
ILXBP	[HLT]	SETPT-BAD POINTER IN XB
IMINX1	[INF]	UNUSUAL ANI INTERRUPT, CONI ANI IS
IMINX2	[INF]	IMIERR CALLED, CONI ANI IS
IMPAB2	[CHK]	ASNTBF: ASNTBF FAILED WHEN NCPLCK SET
IMPABF	[INF]	ASNTBF FAILED
IMPAFB	[HLT]	IMPCQ: ATTEMPT TO UNLOCK BUFFER ON FREELIST
IMPALF	[HLT]	IMPLKB: ATTEMPT TO LOCK BUFFER ON FREELIST
IMPAUF	[HLT]	IMPEIN: ATTEMPT TO UNLOCK BUFFER ON FREELIST
IMPBLK	[CHK]	SNDRFC: Sending RFC for a bad NCP link number
IMPBSC	[INF]	Message has bad size or count
IMPCCF	[HLT]	CANNOT CREATE IMP FORK
IMPCTH	[INF]	IMPNCL TOO HIGH
IMPCUL	[INF]	RECD CTL MSG FOR UNKNOWN LINK
IMPHIF	[INF]	HSTINI FAILED TO FIND HOST NAME FILE
IMPHNW	[CHK]	LHOSTN DISAGREES WITH THE IMP
IMPIFC	[INF]	ILL FMT CTL MSG
IMPIFH	[CHK]	IMPGC-IMPOSSIBLE FAILURE OF IMPHFL
IMPIOP	[CHK]	AN20 CAUSED AN IO PAGE FAIL
IMPLAE	[INF]	IMPOPL: Link already exists
IMPLEO	[INF]	Cannot find LT entry for output message
IMPLTF	[CHK]	IMPLT FULL
IMPMMX	[INF]	MESSAGE STUCK OR OVERDUE TOO LONG
IMPMSL	[INF]	PKMSG - MSG TOO LARGE
IMPMSO	[INF]	MESSAGE STUCK IN OUTPUT QUEUE
IMPMUL	[INF]	RECEIVED MSG FOR UNKNOWN LINK
IMPNBC	[HLT]	PKMSG: NEGATIVE RESIDUAL BYTE COUNT
IMPNEA	[INF]	NVT RECEIVED BYTES EXCEEDING ALLOCATION
IMPNII	[HLT]	No IMP input buffers
IMPNMA	[INF]	PKBY1: NO MSG ALLOCATION
IMPREA	[INF]	RECD EXCESS ALL
IMPREM	[INF]	UPBRB: Received excessive messages
IMPRMI	[HLT]	IMP - REGULAR MESSAGE ON IRREG QUEUE
IMPRNE	[INF]	RECD NCP ERR
IMPRNO	[INF]	RFNM OVERDUE
IMPTMB	[CHK]	NVTXG1: TOO MANY BREAKS OUTSTANDING
IMPUBF	[HLT]	IMULKB: ATTEMPT TO UNLOCK BUFFER ON FREELIST
IMPUFB	[HLT]	IMIP1: ATTEMPT TO UNLOCK BUFFER ON FREELIST
IMPURT	[INF]	IMPDV received unexpected RET
IMPUUO	[HLT]	IMPOSSIBLE MUUO
IMPUXO	[CHK]	IMP JB0 FORK - UNEXPECTED INTERRUPT
IMPXBO	[INF]	IRREG MSG BUFFER OVERFLOW
IMPXUT	[INF]	Received irreg msg with unknown link or type
INCFLK	[CHK]	Fork lock set at return to user
INDCNT	[INF]	DTESRV- BAD INDIRECT COUNT
INVDTE	[HLT]	DTEQ- INVALID DTE SPECIFIED
IOPGF	[HLT]	IO PAGE FAIL
IPCFKH	[CHK]	CHKPDD: COULD NOT FIND LOCAL FORK HANDLE
IPCFRK	[CHK]	PIDINB: CANNOT CREATE FORKS FOR IPCF
IPCJB0	[CHK]	PIDINI: NOT IN CONTEXT OF JOB 0
IPCMCN	[CHK]	MESREC: MESSAGE COUNT WENT NEGATIVE
IPCOVL	[HLT]	PIDINI: PIDS AND FREE POOL OVERLAP, IPCF WILL NOT WORK!
IPCSOD	[CHK]	GETMES: SENDER'S COUNT OVERLY DECREMENTED
ITRLGO	[CHK]	ITRAP - Instruction trap while logging in or out
J0NRUN	[HLT]	JOB 0 NOT RUN FOR TOO LONG, PROBABLE SWAPPING HANGUP
JSBNIC	[HLT]	SETPPG-JSB NOT IN CORE
JTENQE	[HLT]	JTENQ WITH BAD NSKED
KLIOVF	[CHK]	DTESRV-KLINIK DATA BASE TOO LARGE
KPALVH	[HLT]	KEEP ALIVE CEASED
LCKDIR	[HLT]	ATTEMPT TO LOCK DIRECTORY TWICE FOR SAME FORK

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
LNGDIR	[CHK]	LONG DIRECTORY FILE IN DIRECTORY:
LNMILI	[CHK]	LNMLUK: ILLEGAL VALUE OF LOGICAL NAME TABLE INDEX
LOKINT	[CHK]	LOCK BEING LOCKED WHILE OKINT
LOKODR	[HLT]	LOCK REQUESTED OUT OF ORDER
LOKWRG	[HLT]	WRONG FORK IS RELEASING LOCK
LUUMNO	[HLT]	LUUO IN MONITOR CONTEXT
LUUMON	[HLT]	.LBCHK: ILLEGAL LUUO FROM MONITOR CONTEXT
MAP41F	[HLT]	MAPF41 FAILED TO SKIP
MAPBT1	[HLT]	OFN FOR BIT TABLE IS ZERO
MAPCLF	[CHK]	FAILED TO CLEAR MAPS WHEN KILLING JOB
MDDJFN	[HLT]	GETFDB: CALLED FOR NON-MDD DEVICE
MNTLNG	[HLT]	MNTBTB - BIT TABLE IS A LONG FILE
MONPDL	[HLT]	STACK FAULT IN MONITOR
MPEUTP	[HLT]	PFCDPE-UNKNOWN TRAP ON TEST REFERENCE
MPIDXO	[CHK]	MAPIDX - No OFN for Index Table File
MTANOA	[CHK]	IRBDN2: IRBDON CALLED FOR AN ACTIVE IORB
MTANOI	[CHK]	GETUBF: NO QUEUED IORB'S FOR INPUT
MTANOQ	[CHK]	IRBDN1: IRBDON CALLED FOR NON-QUEUED UP IORB
MTAORN	[CHK]	MTDIR0: MAGTAPE IORB OVERRUN
MTARIN	[HLT]	MTAINT: INTERRUPT RECEIVED FOR NONACTIVE IORB
MTMSG	[INF]	FAILED TO SEND MT MESSAGE TO "TAPE" CONTROLLER
NCPFUN	[INF]	NCP FSM RECEIVED FUNNY INPUT
NEGJRT	[CHK]	UCLOCK: NEGATIVE JOBRT DETECTED
NETBAF	[HLT]	RLNTBF: ATTEMPT TO RELEASE BUFFER ALREADY ON FREE LIST
NETBAU	[HLT]	ASNTBF: ATTEMPT TO ASSIGN A BUFFER ALREADY IN USE
NETDET	[CHK]	NVTDET: COULD NOT CLOSE NVT
NETRBG	[CHK]	RLNTBF: ATTEMPT TO RELEASE BUFFER AT GARBAGE LOCATION
NETRBL	[HLT]	ASNTBF: REQUEST FOR BUFFER LARGER THAN MAXWPM
NETWNS	[HLT]	WATNOT: WAS CALLED FROM SCHEDULER LEVEL.
NEWBAK	[HLT]	FILRFS - NEWIB FAILURE FOR BACKUP ROOT-DIR
NEWROT	[HLT]	FILRFS - NEWIB FAILURE FOR ROOT-DIRECTORY
NOACB	[HLT]	MENTR - NO MORE AC BLOCKS
NOADXB	[HLT]	RELOFN-NO DSK ADR FOR XB
NOALCM	[CHK]	ALCMES: CANNOT SEND MESSAGE TO ALLOCATOR
NOARCS	[INF]	ARCMSG: PID for QUASAR is not valid
NOBAT1	[CHK]	FAILED TO WRITE PRIMARY BAT BLOCK
NOBAT2	[CHK]	FAILED TO WRITE SECONDARY BAT BLOCK
NOBTB	[CHK]	FILINI - UNABLE TO OPEN BIT TABLE FILE
NOBTBN	[HLT]	FILINI - UNABLE TO GET SIZE OF BOOTSTRAP.BIN FILE
NOCTY	[HLT]	UNABLE TO ALLOCATE DATA FOR CTY
NODIR1	[CHK]	SPLMES: DIRST FAILED ON EXISTING DIRECTORY NAME
NODTEN	[CHK]	DTESRV - NO DTE BUFFERS AVAILABLE IN CRITICAL CASE
NOFEFS	[HLT]	FILINI - UNABLE TO GET SIZE OF FRONT END FILE SYSTEM
NOFNDU	[HLT]	FNDUNT-CANNOT FIND DEVICE FOR JFN
NOINTR	[CHK]	ITRAP AND PREVIOUS CONTEXT WAS NOINT
NOIORB	[HLT]	SETIRB - MISSING IORB
NOLEN	[HLT]	UPDLEN: NO LENGTH INFO FOR OFN
NOMCCD	[HLT]	TTYSRV: ROUTINE CALLED FOR LINE TYPE NOT SUPPORTED
NOPGT0	[HLT]	OPNLNG: NO PAGE TABLE 0 IN LONG FILE.
NOPID	[CHK]	PIDKFL: PID DISAPPEARED
NORSXF	[HLT]	FAILED TO GET SPACE FOR MASTER DTE
NOSEB2	[HLT]	PGMPE-NO SYSERR BUFFER AVAILABLE
NOSERF	[CHK]	CANNOT GTJFN ERROR REPORT FILE
NOSKTR	[CHK]	ITRAP FROM OR CSKED CONTEXT
NOSLNM	[CHK]	SLNINI: CANNOT CREATE SYSTEM LOGICAL NAME
NOSPLM	[CHK]	RELJFN: COULD NOT SEND SPOOL MESSAGE TO QUASAR
NOTOFN	[HLT]	UPDOF0-ARG NOT OFN
NOUTF1	[CHK]	SPLOPN: NOUT OF DIRECTORY NUMBER FAILED
NOUTF2	[CHK]	SPLMES: NOUT OF GENERATION NUMBER FAILED
NOXADR	[HLT]	EXTENDED ADDRESSING CONFUSION
NPWQPD	[CHK]	PHYSIO - NULL PWQ AT POSITION DONE
NRFTCL	[CHK]	PHYSIO - NO REQUESTS FOUND FOR CYLINDER SEEKED
NSKDIS	[HLT]	DISMISS WHILE NOSKED OR WITH NON-RES TEST ADDRESS
NSKDT2	[CHK]	PGRTRP-BAD NSKED OR INTDF
NSPUDF	[HLT]	UNSUPPORTED NETWORK FUNCTION
NULQTA	[HLT]	QCHK - NO QUOTA INFO SETUP
NWJTBE	[CHK]	NO FREE JTB BLOCKS
OFFONX	[HLT]	ARRST: File marked offline has index blk ptr
OFFSPE	[HLT]	OFFSPQ- PAGE NOT ON SPMQ
OKSKBG	[HLT]	OKSKD0 - OKSKED WHEN NOT NOSKED
OPOPAC	[HLT]	MRETN - TRIED TO OVER-POP AC STACK
OVFLOW	[HLT]	ASOFN - ALLOCATION TABLE OVERFLOW
OVRDTA	[INF]	PHYSIO - OVERDUE TRANSFER ABORTED
P2RAE1	[CHK]	PHYH2 - RH20 REGISTER ACCESS ERROR READING REGISTER
P2RAE2	[CHK]	PHYH2 - REGISTER ACCESS ERR WRITING REG
P2RAE3	[CHK]	PHYH2 - REGISTER ACC ERR ON DONE OR ATN INTERRUPT
PAGLCK	[HLT]	DESPT-PAGE LOCKED
PAGNIC	[HLT]	GETCPP-PAGE NOT IN CORE
PGNDEL	[HLT]	REMFPB-PAGE NOT COMPLETELY DELETED

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
PGUNDX	[HLT]	PGUNTD-IN NESTED TRAP
PH2DNA	[INF]	PHYH2 - DONE INTERRUPT AND CHANNEL NOT ACTIVE
PH2IHM	[CHK]	PHYH2 - ILLEGAL HDW MODE - WORD MODE ASSUMED
PH2PIM	[CHK]	PHYH2 - RH20 LOST PI ASSIGNMENT
PH2WUI	[HLT]	WRONG UNIT INTERRUPTED
PHYCH1	[HLT]	PHYSIO - HOME BLOCK CHECK IORB ALREADY ON TWQ
PHYCH2	[INF]	PHYSIO - HOME BLOCK CHECK IORB TIMED OUT
PHYCH3	[INF]	PHYSIO - HOME BLOCK CHECK IORB TIMED OUT BUT WAS NOT ON TWQ
PHYICA	[HLT]	PHYINI - ILLEGAL ARGUMENT TO CORE ALLOC
PHYICE	[INF]	PHYINI - FAILED TO ASSIGN RESIDENT STG
PHYLTF	[HLT]	PHYSIO - SCHLTM - UNEXPECTED LATOPT FAILURE
PHYNIR	[CHK]	PHYSIO - NULL INTERRUPT ROUTINE AT OPERATION DONE
PHYP0E	[HLT]	PHYALZ - PAGE 0 STORAGE EXHAUSTED
PI1ERR	[CHK]	UNEXPECTED UNVECTORED INTERRUPT ON CHANNEL 1
PI2ERR	[CHK]	UNEXPECTED UNVECTORED INTERRUPT ON CHANNEL 2
PI4ERR	[CHK]	UNEXPECTED UNVECTORED INTERRUPT ON CHANNEL 4
PI6ERR	[CHK]	UNEXPECTED UNVECTORED INTERRUPT ON CHANNEL 6
PIDFLF	[CHK]	CREPID: FREE PID LIST FOULED UP
PIDOD1	[CHK]	MUTCHO: PID COUNT OVERLY DECREMENTED
PIDOD2	[CHK]	DELPID: OVERLY DECREMENTED PID COUNT
PIITRP	[HLT]	INSTRUCTION TRAP WHILE PI IN PROGRESS OR IN SCHEDULER
PISKED	[HLT]	ENTERED SCHEDULER WITH PI IN PROGRESS
PITRAP	[HLT]	PAGER TRAP WHILE PI IN PROGRESS
PM2SIO	[CHK]	PHYM2 - ILLEGAL FUNCTION AT START IO
PM8SIO	[CHK]	PHYM78 - ILLEGAL FUNCTION AT START IO
PRONX2	[HLT]	NXM DETECTED BY PROCESSOR
PSBNIC	[HLT]	SETPPG-PSB NOT IN CORE
PSINSK	[CHK]	PSI FROM NOSKED OR CRSKED CONTEXT
PSISTK	[HLT]	PSI STORAGE STACK OVERFLOW
PTAIC	[HLT]	SWPIN - PT PAGE ALREADY IN CORE
PTDEL	[HLT]	DESPT-PT NOT DELETED
PTMPE	[HLT]	PAGE TABLE PARITY ERROR
PTNIC1	[HLT]	SWPIN - PAGE TABLE NOT IN CORE
PTNON0	[HLT]	SETPT0 - PREVIOUS CONTENTS NON-0
PTOVRN	[HLT]	UPDPGS-COUNT TOO LARGE
PVTRP	[HLT]	PROPRIETARY VIOLATION TRAP
PWRFL	[HLT]	FATAL POWER FAILURE
PWRRES	[CHK]	POWER RESTART
PYILUN	[HLT]	PHYSIO - ILLEGAL UNIT NUMBER
RCVNOE	[CHK]	RCVOK - NO ENTRY FOUND IN QUEUE
RCVTMR	[CHK]	RCVOK TIMEOUT - IGNORING ACCESS CONTROL JOB
RELBAD	[CHK]	RELFRE-BAD BLOCK BEING RELEASED
RELFRM	[HLT]	ILLEGAL TO DEASSIGN 0 FREE SPACE
RELINT	[CHK]	RELFRE CALLED OKINT
RELRNG	[CHK]	RELFRE: BLOCK OUT OF RANGE
RESBAD	[CHK]	RELRES: ILLEGAL ADDRESS PASSED TO RELRES
RESBAZ	[CHK]	RELRES: FREE BLOCK RETURNED MORE THAN ONCE
RESBND	[CHK]	RELRES: RELEASING SPACE BEYOND END OF RESIDENT FREE POOL
RESCHK	[HLT]	RELRES: RESIDENT FREE SPACE WAS OVERWRITTEN
RFILPF	[CHK]	REFILL ERROR PAGE FAIL
RH2ICF	[HLT]	PHYRH2 - INVALID CHANNEL FUNCTION
RP4FEX	[HLT]	PHYP4 - ILLEGAL FUNCTION
RP4IF2	[HLT]	PHYP4 - ILLEGAL FUNCTION AT STKIO
RP4IFC	[HLT]	PHYP4 - ILLEGAL FUNCTION AT CNV
RP4ILF	[HLT]	PHYP4 - ILLEGAL FUNCTION ON INTERRUPT
RP4LTF	[HLT]	PHYP4 - FAILED TO FIND TWQ ENTRY AT RP4LTM
RP4PNF	[HLT]	PHYP4 - DISK PHYSICAL PARAMETERS NOT FOUND
RP4SSC	[CHK]	PHYP4 - STUCK SECTOR COUNTER
RP4UNF	[HLT]	PHYP4 - UNIT TYPE NOT FOUND:
RPGERR	[HLT]	BADCPG-FATAL ERROR IN RESIDENT PAGE
RSMFAI	[HLT]	RESSMM-FAILED TO ASSIGN SWAP MON PAGE
SBSERF	[INF]	SBSERR-COULD NOT GET ERROR BLOCK
SCDUUO	[HLT]	UUO IN SCHEDULER
SCPT01	[HLT]	SCNPT - ENTRY IS NOT AN IMMEDIATE POINTER
SCPT02	[HLT]	SCNPT - PAGE WAS NOT DELETED
SEBISS	[CHK]	SEBCPY-INSUFFICIENT STRING STORAGE IN BLOCK
SEBUDT	[CHK]	SEBCPY-UNKNOWN DATA TYPE
SECEX1	[HLT]	SETMPG-ATTEMPT TO MAP NON-EX SECTION
SERFOF	[CHK]	CANNOT OPENF ERROR REPORT FILE
SERFRK	[HLT]	SERINI-CANNOT CREATE SYSERR FORK
SERGOF	[CHK]	SETOFI-CANNOT GTJFN/OPEN SYSERR FILE
SHRN00	[HLT]	DESPT-SHARE COUNT NON-ZERO
SHROFD	[HLT]	DWNSHR-OFN SHARE COUNT UNDERFLOW
SHROFN	[HLT]	UPSHR-OFN SHARE COUNT OVERFLOW
SKDCL1	[HLT]	CALL TO SCHEDULER WHEN ALREADY IN SCHEDULER
SKDMPE	[HLT]	MPE IN SCHEDULER OR PI CONTEXT
SKDPF1	[HLT]	PAGE FAIL IN SCHED CONTEXT
SKDTRP	[HLT]	INSTRUCTION TRAP WHILE IN SCHEDULER

TOPS-20

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
SNPIC	[CHK]	SNPFN3: INSTRUCTION BEING REPLACED HAS CHANGED
SNPLKF	[CHK]	SNPFN0: CANNOT LOCK DOWN PAGE INTO MONITOR
SNPODB	[CHK]	SNPF4C: COUNT OF INSERTED BREAK POINTS OVERLY DECREMENTED
SNPUNL	[CHK]	SNPF5A: CANNOT UNLOCK SNOOP PAGE
SPTFL1	[HLT]	SPT COMPLETELY FULL
SPTFL2	[HLT]	SPT COMPLETELY FULL
SPTPIC	[HLT]	SWPIN - SPT PAGE ALREADY IN CORE
SPTSHR	[HLT]	UPSHR-SPT SHARE COUNT OVERFLOW
SPWRFL	[CHK]	SPURIOUS POWER FAIL INDICATION
SRQBAD	[CHK]	SCDRQ-BAD CALL TO SCDRQ7
STKOVF	[HLT]	MONITOR STACK OVERFLOW
STRBAD	[HLT]	ASOFN-ILLEGAL STRUCTURE NUMBER
STRTER	[HLT]	FATAL ERROR WHILE PROCESSING PREVIOUS STARTUP ERROR
STZERO	[HLT]	FILINI: STRTAB ENTRY FOR PS IS 0
SUMNR1	[CHK]	AJBALX-SUMBNR INCORRECT
SUMNR2	[CHK]	WSMGR-SUMNR INCORRECT
SWPASF	[CHK]	CHKBAT-FAILED TO ASSIGN BAD SWAPPING ADDRESS
SWPFPE	[CHK]	SWAP ERROR IN SENSITIVE FILE PAGE
SWPIBE	[CHK]	SWAP ERROR IN INDEX BLOCK
SWPJSB	[CHK]	SWAP ERROR IN JSB PAGE
SWPMNE	[HLT]	SWAP ERROR IN SWAPPABLE MONITOR
SWPPSB	[CHK]	SWAP ERROR IN PSB PAGE
SWPPT	[CHK]	SWAP ERROR IN UNKNOWN PT
SWPPTP	[CHK]	SWAP ERROR IN UNKNOWN PT PAGE
SWPSTL	[CHK]	SWAP SPACE TOO LOW AT STARTUP
SWPUPT	[CHK]	SWAP ERROR IN UPT, OR PSB
SWPXXX	[HLT]	UNRECOVERABLE SWAP ERROR FOR CRITICAL PAGE
SYMNOM	[CHK]	Unable to map symbol table page
SYMNOU	[CHK]	Unable to unmap symbol table page
SYSERF	[CHK]	LOGSST-NO SYSERR STORAGE FOR RESTART ENTRY
TM2CCI	[CHK]	PHYM2 - TM02 SSC OR SLA WONT CLEAR
TM2HER	[CHK]	TM2ERR - IS.HER SET ON SUCCESSFUL RETRY
TM2IDM	[CHK]	PHYM2 - ILLEGAL DATA MODE AT DONE INT
TM2IDX	[INF]	PHYM2 - ILLEGAL RETRY BYTE POINTER
TM2IF2	[CHK]	PHYM2 - ILLEGAL FUNCTION ON COMMAND DONE
TM2IRF	[INF]	PHYM2 - ILLEGAL FUNCTION DURING RETRY
TM2N2S	[INF]	PHYM2 - MORE DRIVES THAN TABLE SPACE, EXCESS IGNORED
TM2NUD	[CHK]	PHYM2 - CHANNEL DONE INTERRUPT BUT NO UNIT ACTIVE
TM2RFU	[CHK]	PHYM2 - ERROR RECOVERY CONFUSED
TM2UNA	[INF]	PHYM2 - DONE INTERRUPT AND UDB NOT ACTIVE
TM8AEI	[INF]	PHYM78 - ASYNCHRONOUS ERROR INTERRUPT
TM8N2S	[INF]	PHYM78 - MORE DRIVES THAN TABLE SPACE, EXCESS IGNORED
TM8NUD	[CHK]	PHYM78 - CHANNEL DONE INTERRUPT BUT NO UNIT ACTIVE
TM8SNS	[CHK]	CAN'T SENSE TU78 STATUS
TRPSIE	[CHK]	NO MONITOR FOR TRAPPED FORK
TTBAD1	[HLT]	BAD DEVICE DESIGNATOR FOR TERMINAL AT ATACH2
TTDAS1	[HLT]	HLTJB: UNABLE TO DEASSIGN CONTROLLING TERMINAL
TTFSMS	[INF]	Failed to send system message
TTICN0	[HLT]	TCI - NO BUFFER POINTER BUT COUNT NON-0
TTILEC	[CHK]	TTSND-UNRECOGNIZED ESCAPE CODE
TTLOKB	[HLT]	BAD TTY LOCK IN TTLCK
TTNAC1	[CHK]	LINE NOT ACTIVE AT PTYOPN
TTNAC3	[HLT]	CTY NOT ACTIVE AT FSIPBO
TTNAC4	[HLT]	CTY NOT ACTIVE AT FSIPBI
TTNAC5	[HLT]	CTY NOT ACTIVE AT FSIINI
TTNAC7	[CHK]	DEALLOCATING INACTIVE LINE
TTNAC8	[HLT]	CANNOT ASSIGN TERMINAL AT DEVINI
TTONOB	[HLT]	TTY OUTPUT - NO BUFFER BUT COUNT NON-0
TTQADX	[CHK]	TTYSRV-UNKNOWN FUNCTION REQUESTED
TTULKB	[CHK]	BAD TTY UNLOCK IN ULKTT
TTYBBO	[CHK]	TTYSRV-BIG BUFFER OVERFLOW
TTYNTB	[CHK]	RAN OUT OF TTY BUFFERS
TTYSTP	[INF]	TTYSRV - LINE HAS BEEN SHUT OFF BECAUSE OF EXCESSIVE INPUT RATE
TWQNUL	[HLT]	PHYSIO - PWQ OR TWQ WAS NULL AT A SEEK OR TRANSFER COMPLETION
UBANXM	[HLT]	I/O NMX FROM UNIBUS DEVICE
UIONIR	[HLT]	UDSKIO - NO IORB FOR NOSKED FORK
ULKBAD	[CHK]	UNLOCKING TTY WHEN COUNT IS ZERO
ULKINT	[CHK]	LOCK BEING UNLOCKED WHILE OKINT
ULKSTZ	[CHK]	OVERLY DECREMENTED STRUCTURE LOCK
UNBFNF	[CHK]	UNBLK1 - FORK NOT FOUND
UNFWSS	[HLT]	UNIT NOT FOUND CREATING SDB FOR STRUCTURE
UNPGF1	[HLT]	MEMPAR-PARITY ERROR DURING MEM SCAN
UNPGF2	[HLT]	UNKNOWN PAGE FAILURE TYPE
UNPIRX	[CHK]	UNPIR-NO PSI IN PROGRESS
UNXMPE	[HLT]	PFCDPE-UNEXPECTED PARITY ERROR TRAP
UPTMPE	[HLT]	PFCDPE: PARITY ERROR IN UPT
USGHOL	[INF]	LOST PAGE(S) IN USAGE FILE
UXXCKP	[HLT]	COULDN'T CREATE CHECKPOINT FILE

Table 4 BUGHLT and BUGCHK Names and Descriptions (Cont)

Name	Type	Description
UXXCL1	[CHK]	UNABLE TO CREATE NEW USAGE FILE
UXXCL2	[CHK]	UNABLE TO OPEN NEW USAGE FILE
UXXCL3	[CHK]	UNABLE TO CLOSE USAGE FILE
UXXCRE	[HLT]	CANNOT CREATE USAGE FILE
UXXFAI	[CHK]	USAGE JSYS FAILURE
UXXFIT	[INF]	CHECKPOINT FILE NOT IN CORRECT FORMAT FOR THIS SYSTEM, REBUILDING...
UXXILL	[HLT]	USGMES: ILLEGAL FUNCTION CODE
UXXMAP	[HLT]	USGMAP: CALL TO JFNOFN FAILED
UXXOPN	[HLT]	UNABLE TO OPEN USAGE FILE
UXXWER	[CHK]	WRITE ERROR IN USAGE FILE
WAITNI	[HLT]	WAIT JSYS while not interruptable
WRTBT4	[CHK]	ASOFN ON BIT TABLE FILE FAILED
WRTCPB	[CHK]	WRTBTB - FAILED TO BACKUP ROOT-DIRECTORY
WRTLNG	[HLT]	WRTBTB - BIT TABLE IS A LONG FILE
WSPNEG	[CHK]	SOSWSP-WSP NEGATIVE
WSSPNA	[HLT]	WSSFKP-FORK SPECIAL PAGE BAD AGE
WSSPNC	[HLT]	WSSFKP-FORK SPECIAL PAGE NOT IN CORE
XBWERR	[CHK]	UPDOFN-DSK WRITE ERROR ON XB
XSCORE	[HLT]	CST TOO SMALL FOR PHYSICAL CORE PRESENT