



RTE-A

Driver Reference Manual

**Software Services and Technology Division
11000 Wolfe Road
Cupertino, CA 95014-9804**

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THE MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information that is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced, or translated to another language without the prior written consent of Hewlett-Packard Company.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARs 252.227.7013.

Printing History

The Printing History below identifies the edition of this manual and any updates that are included. Periodically, update packages are distributed that contain replacement pages to be merged into the manual, including an updated copy of this printing history page. Also, the update may contain write-in instructions.

Each reprinting of this manual will incorporate all past updates; however, no new information will be added. Thus, the reprinted copy will be identical in content to prior printings of the same edition with its user-inserted update information. New editions of this manual will contain new information, as well as all updates.

To determine what manual edition and update is compatible with your current software revision code, refer to the Manual Numbering File. (The Manual Numbering File is included with your software. It consists of an "M" followed by a five digit product number.)

Second Edition	Mar 1983	
Update 1	Dec 1983	Add Modem Card 37222A
Third Edition	Jun 1984	15-Mb,MAC,9144,7974,2566
Update 1	Jan 1985	MUX, 7978, 7941, 7945
Reprint	Jan 1985	Update 1 Incorporated
Fourth Edition	Jan 1986	
Update 1	Oct 1986	New Serial Drivers
Reprint	Oct 1986	Update 1 Incorporated
Fifth Edition	Aug 1987	Rev. 5000 (Software Update 5.0)
Update 1	Jan 1989	Rev. 5010 (Software Update 5.1)
Update 2	Jul 1990	Rev. 5020 (Software Update 5.2)
Sixth Edition	Dec 1992	Rev. 6000 (Software Update 6.0)
Seventh Edition	Nov 1993	Rev. 6100 (Software Update 6.1)
Eighth Edition	Apr 1995	Rev. 6200 (Software Update 6.2)

Preface

This manual describes the device and interface drivers of the RTE-A Operating System. Device and interface drivers allow the operating system to communicate with peripheral devices via the computer's interface cards. The drivers are programmed via standard EXEC I/O requests.

This manual is written for the system programmer or system manager experienced with the RTE-A Operating System and familiar with HP 1000 peripherals. The manual assumes a familiarity with the HP 1000 A-Series Computer I/O structure.

Chapter 1 explains the differences between device and interface drivers, and explains the parameters used in EXEC I/O calls for read, write, control, and status requests to the drivers. The general programming information of Chapter 1 should be read before reading the driver descriptions in Chapter 2. The descriptions assume that you understand information in Chapter 1.

Chapter 2 describes the RTE-A device and interface drivers. Each device driver description lists the devices that the driver supports, and explains programming the driver in detail. The device driver descriptions present driver- and device-specific programming information. The interface driver descriptions explain interface- and driver-specific programming considerations.

The updated serial I/O interface and device drivers first added to the operating system as of Revision 4010, are documented at the end of Chapter 2. This includes support of the A400 4-channel on-board I/O (OBIO) and the HP 12040D 8-Channel Multiplexer. It also includes the HP 12005A/B Asynchronous Serial Interface Card added at Revision 5000. Supplemental information on these drivers is also contained in the Appendices. Appendix J contains the documentation on the serial drivers prior to Revision 4010.

Note that at Revision 6200 of this manual, for easier reference Chapter 2 was restructured to include all device and interface drivers, including SCSI and the Revision 4010 serial I/O drivers.

Appendix A lists the HP character set.

Appendix B is a quick-reference guide to all of the drivers.

Appendix C explains the differences between interface driver IDM00 and the interface drivers released at Revision 4010. It also discusses conversion from a Revision C 8-channel MUX to a Revision D version, or to an A400 4-channel MUX.

Appendix D contains example protocol charts for block mode ASCII reads and writes.

Appendix E contains system generation considerations that enable you to pre-configure the drivers to "come up" in the right state when the system is booted.

Appendix F is a sample of a page mode application.

Appendix G is a state diagram of the transitions caused by the program sequence calls (CN 20, 21, 40, and 41) in RTE-A.

- Appendix H contains an explanation of HP-supplied source code programs for modem control and automatic restart from power failure.
- Appendix I explains how to create a new special character PROM by modifying the standard HP lookup PROM.
- Appendix J contains documentation on the serial I/O drivers introduced prior to Revision 4010. These include DD*00, DD*20, IDM00, ID*00, and ID*01. At Revision 4010, these drivers were replaced by the serial I/O drivers documented in Chapter 2.

Table of Contents

Chapter 1 I/O Request Conventions

Device and Interface Driver Separation	1-1
Device Table (DVT)	1-2
Interface Table (IFT)	1-2
Read/Write Parameter Definitions	1-4
Read/Write Request Conventions	1-4
ecode	1-5
cntwd	1-5
bufr	1-8
bufln	1-8
pram3 and pram4	1-8
0 (Zero)	1-8
keywd	1-8
A- and B-Register Contents	1-9
Control Request Conventions	1-11
Control Request Parameter Definitions	1-12
ecode	1-12
cntwd	1-12
pram1 through pram4	1-14
0 (Zero)	1-15
keywd	1-15
A- and B-Register Contents	1-15
Status Request Conventions	1-16
Status Parameter Definitions	1-17
ecode	1-17
cntwd	1-17
stat1	1-17
stat2	1-18
stat3 and stat4	1-18
Extended Device Status	1-19
EXEC Error Returns	1-21
ecode No-Abort Bit	1-22
Device Model Numbers for System Generation	1-23

Chapter 2 Device and Interface Drivers

HP-IB Line Printer Driver DD*12	2-2
DD*12 Write Request	2-2
Control Word cntwd	2-2
bufr and bufln	2-3
A- and B-Register Contents	2-3
DD*12 Control Request	2-3
Control Word cntwd	2-4
Function Code 00: Clear and Reset	2-4
Function Code 11B: Line Skipping and Form Feed	2-4

DD*12 Status Request	2-5
Control Word cntwd	2-5
stat1 and stat2 Status Parameters	2-6
stat3 and stat4 Status Parameters	2-6
DD*12 Extended Status	2-6
HP-IB Printer Driver DDC12	2-8
Bringing the Printer Up Automatically	2-8
Powerfail Recovery	2-8
DDC12 Write Request	2-9
Control Word cntwd	2-9
bufr and bufln	2-11
DDC12 VFC Definition Request	2-11
bufr and bufln	2-11
DDC12 Control Request	2-12
Control Word cntwd	2-12
DDC12 Status Request	2-17
Control Word cntwd	2-17
stat1	2-17
stat2	2-17
stat3 and stat4	2-18
DDC12 Extended Status Request	2-19
DDC12 Error Information	2-20
HP-IB Magnetic Tape Driver DD*23	2-22
DD*23 Read/Write Request	2-22
Control Word cntwd	2-22
bufr and bufln	2-22
A- and B-Register Contents	2-22
DD*23 Control Request	2-23
DD*23 Status Request	2-24
Control Word cntwd	2-24
stat1 and stat2 Status Parameters	2-24
stat3 and stat4 Status Parameters	2-25
DD*23 Extended Status	2-25
HP-IB Magnetic Tape Driver DD*24	2-27
DD*24 Read/Write Request	2-27
Control Word cntwd	2-27
bufr and bufln	2-28
A- and B-Register Returns	2-28
DD*24 Control Request	2-28
Control Word cntwd and Parameters	2-28
A- and B-Register Returns	2-30
DD*24 Status Request	2-30
Control Word cntwd	2-30
Parameters stat1 and stat2	2-30
Parameters stat3 and stat4	2-31
DD*24 Extended Status	2-31
SCSI Tape Device Driver DDQ24	2-34
DDQ24 Driver Read and Write Calls	2-34
DDQ24 Z-Buffer Calls	2-35
Read SCSI Command and Sense Data Request After Check Condition	2-35
Control Calls	2-36
Control 0B Rewind	2-37
Control 4B Rewind	2-37

Control 1B Write File Mark	2-37
Control 2B Backward Space Record	2-38
Control 3B Forward Space Record	2-38
Control 5B Rewind and Unload Tape	2-38
Control 6B Dynamic Status	2-39
Control 7B Write Set Mark	2-39
Control 10B Backward Space Set Mark	2-40
Control 11B Forward Space Set Marks	2-40
Control 13B Forward Space File	2-40
Control 14B Backward Space File	2-41
Control 15B Set Tape Density, Enable/Disable Compression	2-41
Control 16B Enable/Disable Driver Request Sense After Check Condition ...	2-42
Driver Parameter Table Use in DDQ24	2-42
Status	2-42
Logical Unit Status (DVT Word 6)	2-43
Driver Error (DVT Word 16)	2-43
Transmission Log (DVT Word 17)	2-44
SCSI Status and Transaction Status (DVT Word 18)	2-44
SCSI Sense Key and Additional Sense Code (DVT Word 19)	2-45
DDQ24 Driver Communication Word (DVT Word 20)	2-45
Disk Device Driver DD*30	2-46
DD*30 Read/Write Request	2-46
Control Word cntwd	2-46
bufn and bufnl	2-46
track and sector Parameters	2-47
A- and B-Register Contents	2-47
DD*30 Control Request	2-47
DD*30 Status Request	2-48
Control Word cntwd	2-48
stat1 and stat2 Status Parameters	2-48
stat3 and stat4 Status Parameters	2-49
Driver Parameter Area	2-49
DD*30 Extended Status	2-49
DD*30 Error Information	2-52
Disk Device Driver DDM30	2-54
DDM30 Read/Write Request	2-54
Control Word cntwd	2-54
bufn and bufnl	2-54
track and sector Parameters	2-55
A- and B-Register Contents	2-55
DDM30 Control Request	2-55
DDM30 Status Request	2-56
Control Word cntwd	2-56
stat1 and stat2 Status Parameters	2-56
stat3 and stat4 Status Parameters	2-56
Driver Parameter Area	2-57
DDM30 Extended Status	2-57
DDM30 Error Information	2-59
SCSI Disk Device Driver DDQ30	2-61
DDQ30 Read and Write Calls	2-61
Z-Buffer Calls	2-62
Control Calls	2-62
Control 16B Enable/Disable Driver Request Sense After Check Condition ...	2-63

Control 76B Alter Driver Parameter Table (RTE Block Information)	2-63
Driver Parameter Table Use in DDQ30	2-64
Logical Unit Status (DVT Word 6)	2-65
Driver Error (DVT Word 16)	2-65
Transmission Log (DVT Word 17)	2-66
SCSI Status and Transaction Status (DVT Word 18)	2-66
SCSI Sense Key and Additional Sense Code (DVT Word 19)	2-66
Driver Communication Word (DVT Word 20)	2-67
CS/80 Disk Device Driver DD*33	2-68
DD*33 Disk Read/Write Request	2-68
Read/Write Request Control Word cntwd	2-69
bufr and bufln	2-69
track and sector	2-69
Logical/Physical Sector Correspondence	2-69
A- and B-Register Contents	2-70
DD*33 Disk Control Request	2-70
Control Request cntwd	2-70
DD*33 Status Request	2-71
Status Request Control Word cntwd	2-71
stat1 and stat2 Status Parameters	2-71
STAT3 and STAT4 Status Parameters	2-72
DD*33 Driver Parameter Area	2-73
DD*33 Extended Status	2-73
Reject Errors Field (Class 1)	2-75
Fault Errors Field (Class 2)	2-76
Access Errors Field (Class 3)	2-77
Information Errors Field (Class 4)	2-78
DD*33 CTD Read/Write Request	2-79
bufr and bufln	2-79
hiblk and loblk	2-79
A- and B-Register Contents	2-80
DD*33 CTD Control Requests	2-80
Using the Cartridge Tape Drive	2-80
Clear Cache	2-81
Close Cache	2-81
Unload Tape	2-82
DD*33 Direct Disk Control	2-83
Calling Format	2-83
XINMD	2-87
XRELS/XRELD	2-87
XRQST/XDESC	2-87
XCOPY	2-88
XCOMP	2-89
DD*33 Error Information	2-89
Bad Tape Indicator	2-91
Disk Interface Driver ID*27	2-92
ID*27 Read/Write Request	2-92
Control Word cntwd	2-92
bufr and bufln	2-92
track and sector Parameters	2-92
A- and B-Register Contents	2-93
Control Word Read Request Example	2-93
ID*27 Control Request	2-93

ID*27 Status Request	2-94
Control Word cntwd	2-94
stat1 and stat2 Status Parameters	2-94
stat3 and stat4 Status Parameters	2-95
Driver Parameter Area	2-95
ID*27 Extended Status	2-95
ID*27 Error Information	2-98
Interface Driver IDQ35	2-99
Driver Error (DVT Word 16)	2-100
PROM Storage Module Driver ID*36	2-101
ID*36 Read/Write Request	2-101
Control Word cntwd	2-101
bufr and bufln	2-101
track and sector	2-101
A- and B-Register Contents	2-102
Example	2-102
ID*36 Control Requests	2-102
ID*36 Status Reporting	2-102
Control Word cntwd	2-102
stat1 and stat2 Status Parameters	2-103
stat3 and stat4 Status Parameters	2-103
ID*36 Extended Status	2-103
Driver Parameter Area	2-104
ID*36 Error Information	2-104
HP-IB Interface Card Driver ID*37	2-105
ID*37 Operating Modes	2-106
Auto-Addressing	2-106
Direct I/O	2-106
ID*37 End-Of-Record Processing	2-107
ID*37 Auto-Addressed Read/Write Requests	2-108
Control Word cntwd	2-108
bufr and bufln	2-109
pram3 (Secondary Address)	2-109
ID*37 Auto-Addressed Control Requests	2-109
Function Code 00: Selected Device Clear (SDC)	2-110
Function Code 06B: Device Dynamic Status	2-110
Function Code 16B: Set REN True	2-111
Function Code 17B: Go to Local (GTL)	2-111
Function Code 20B: SRQ Program Scheduling	2-111
Function Code 21B: Disable SRQ Program Scheduling	2-112
Function Code 22B: Set Interface Driver Timeout	2-112
Function Code 23B: Parallel Poll Assignment	2-112
Function Code 24B: Set Device Address	2-112
Function Code 27B: Group Execute Trigger (GET)	2-113
Function Code 30B: Disable SRQ Interrupts	2-113
Function Code 31B: Restore SRQ Interrupts	2-113
Function Code 40B: Enable Parallel Poll Program Scheduling	2-114
Function Code 41B: Disable Parallel Poll Program Scheduling	2-114
ID*37 Direct I/O Read/Write Requests	2-114
Device Addresses	2-115
Control Word cntwd	2-115
bufr and bufln	2-115
pram3 and pram4 Control Buffer Descriptors	2-116

ID*37 Direct I/O Control Request	2-116
Control Word cntwd	2-116
Function Code 00: Clear and Reset Device	2-117
Function Code 06B: Dynamic Bus Status	2-117
Function Code 16B: Set REN True	2-118
Function Code 17B: Go To Local and Clear Local Lockout (GTL)	2-118
Function Code 23B: Parallel Poll Configure	2-118
Function Code 25B: Local Lockout (LLO)	2-119
Function Code 27B: Group Execute Trigger (GET)	2-119
Function Code 40B: Enable Parallel Poll Program Scheduling	2-119
Function Code 41B: Disable Parallel Poll Program Scheduling	2-119
Function Code 51B: Bailout (ABORT)	2-119
pram3 and pram4 Control Buffer Descriptors	2-119
ID*37 Status Request	2-120
Control Word cntwd	2-120
stat1 and stat2 Status Parameters	2-120
stat3 and stat4 Status Parameters	2-120
Driver Parameter Area	2-120
ID*37 Extended Status	2-121
HP-IB Universal Commands	2-121
RAM Disk Interface Driver IDR37	2-123
IDR37 Read/Write Request	2-123
Control Word cntwd	2-123
bufn and bufnl	2-123
track and sector Parameters	2-123
A- and B-Register Contents	2-124
Control Word Read Request Example	2-124
IDR37 Control Request	2-124
IDR37 Configuration Request	2-125
IDR37 Deallocate Request	2-125
IDR37 Status Request	2-126
Control Word cntwd	2-126
stat1 and stat2 Status Parameters	2-126
stat3 and stat4 Status Parameters	2-126
Driver Parameter Area	2-127
IDR37 Extended Status	2-127
GPIO/Parallel Interface Card Driver ID*50	2-128
ID*50 Read/Write Requests	2-128
Control Word cntwd	2-128
bufn and bufnl	2-128
pram1 Optional Parameter	2-128
ID*50 Control Request	2-129
Control Word cntwd	2-129
Function Code 00: Clear and Reset Card	2-129
Function Code 06B: Return Dynamic Status of Card	2-129
Function Code 20B: Enable Program Scheduling	2-130
Function Code 21B: Disable Program Scheduling	2-130
Function Code 23B: Enable/Disable Asynchronous Interrupts	2-130
Function Code 24B: Set PIC Control Lines	2-131
Function Code 40B: Configure Card Control Word	2-131
ID*50 Status Reporting	2-133
Control Word cntwd	2-133
stat1 and stat2 Status Parameters	2-134

stat3 and stat4 Status Parameters	2-134
Driver Parameter Area	2-134
ID*50 Parallel Interface Card Control Register	2-134
ID*50 Parallel Interface Card Status	2-136
ID*50 Program Scheduling	2-137
PIC Intercomputer Communications Driver ID*52	2-138
ID*52 Read/Write Requests	2-138
Control Word cntwd	2-138
bufr and bufln, wbuf and wbufl	2-138
pram1 Optional Parameter	2-138
ID*52 Control Requests	2-139
Control Word cntwd	2-139
Function Code 00: Clear and Reset Card	2-139
Function Code 01B: Send EOF	2-139
Function Code 06B: Return Dynamic Status of Card	2-140
Function Code 11B: Top-of-Form EOF	2-140
Function Code 20B: Enable Program Scheduling	2-141
Function Codes 21B/23B: Disable Program Scheduling	2-141
Function Code 22B: Set Timeout	2-141
Function Codes 45B/46B: Use Level/Pulse Mode DVCMD	2-141
ID*52 Status Reporting	2-142
Control Word cntwd	2-142
stat1 and stat2 Status Parameters	2-142
stat3 and stat4 Status Parameters	2-142
Driver Parameter Area	2-142
ID*52 Parallel Interface Card Control Register	2-143
ID*52 Parallel Interface Card Status	2-144
Serial I/O Drivers	2-145
Special Characters	2-146
Carriage Return (octal 015) (Ctrl-M)	2-146
Line Feed (octal 012) (Ctrl-J)	2-146
Backspace (octal 010) (Ctrl-H)	2-146
Delete (octal 177)	2-146
EOT (octal 004) (Ctrl-D)	2-146
Break	2-147
ASCII vs. Binary Read Modes	2-147
Serial I/O Interface Drivers	2-147
Serial I/O Device Drivers	2-148
Slave device support using DDC01	2-148
Capability	2-148
Generation	2-148
Initialization	2-149
TELNET Pseudo Terminal LU	2-149
Using the Bypass Bit (Bit 15)	2-149
Read Request	2-150
bufr and bufln	2-150
cntwd (Read Request Control Word)	2-151
Pseudo Full Duplex (A400 Only)	2-153
Block Mode Read	2-153
AH: Auto-Home Bit	2-154
Special Status Read	2-154
Write Request	2-156
bufr and bufln	2-156

cntwd (Write Request Control Word)	2-156
Control Requests	2-158
cntwd (Control Request Control Word)	2-158
Function Code 6B: Dynamic Status	2-159
DVT6: Device Status	2-160
DVT16: Error Code (System and Driver Defined Errors)	2-161
DVT17: Transmission Log	2-163
DVT18: Interface Card Status	2-163
DVT19: Interface Driver Information	2-163
DVT20: Driver Communication Area	2-164
Dynamic Status Special Forms	2-164
Function Code 11B: Line Spacing/Page Eject	2-164
Function Code 16B: Configure Baud Rate Generator	2-165
Function Code 17B: Define Termination Character	2-166
Function Code 20B: Enable Program Scheduling	2-167
Program Scheduling Conditions	2-167
Pass Program Name	2-167
RTE Compatibility: Pass Program Name	2-168
Function Code 21B: Disable Program Scheduling	2-168
Function Code 22B: Set Device Timeout	2-169
Function Code 25B: Read HP Terminal Straps	2-169
Function Code 26B: Flush Input Buffers	2-169
Function Code 30B: Set Port Configuration	2-170
CF: Character Framing	2-171
M: Modem Control	2-171
x: (Don't Care Bit)	2-172
SB: Stop Bit Selection	2-172
PA: Parity Check	2-173
SS: Speed Sense at Logon	2-173
S: Speed (Baud Rate Selection)	2-174
Default BRG Ranges (ID800/ID801 Only)	2-175
PO: Port Number	2-176
Function Code 31B: Modem Environment	2-177
Function Code 32B: Generate Break	2-177
Function Code 33B: FIFO Buffer Mode Control	2-178
Function Code 34B: Set Port Protocol	2-180
ENQ/ACK Handshake Details	2-185
Function Code 35B: Reset a Baud Rate Generator	2-186
Function Code 40B: Enable Program Scheduling	2-186
Function Code 41B: Disable Program Scheduling	2-186
Device/Interface Driver Communication	2-187
Multibuffer Linked List Structure	2-189

Appendix A HP Character Set

Appendix B Quick Reference Guide

DD*00	B-2
DD*12	B-4
DDC12	B-5
DD*20	B-8
DD*23	B-10
DD*24	B-12
DDQ24	B-14
DD*30	B-17
DDM30	B-18
DDQ30	B-19
DD*33	B-21
ID*00/01	B-23
IDM00	B-25
ID*27	B-26
ID*36	B-28
ID*37	B-30
ID*50	B-32
ID*52	B-33
ID100, ID101, ID400, ID800, ID801, IDZ00 Serial Interface Drivers	B-35
IDQ35 SCSI Interface Driver	B-35

Appendix C Comparison of Serial Drivers

Introduction	C-1
Read Requests	C-2
Special Status Read	C-2
Auto-Home	C-2
CRLF Echo Inhibit	C-2
Pseudo Full Duplex (4-Channel MUX Only)	C-2
User-Defined Terminators	C-2
Write Requests	C-3
Configuration	C-3
CN 6: Dynamic Status	C-3
CN 11: Line Spacing/Page Eject	C-3
CN 16: Set Baud Rate Generator (8-Channel MUX Only)	C-3
CN 30: Speed Sensing	C-3
CN 31: Modem Environment	C-4
CN 32: Generate Break (8- and 4-Channel MUX)	C-4
CN 33: FIFO Control (Type-Ahead)	C-4
FIFO Mode and Type-Ahead Mode	C-4
CN 34: Set Port Protocol	C-7
CN 36: Set Read Length	C-7
CN 37: Set Read Type	C-7

Appendix D Example Protocol Charts

Appendix E System Generation Considerations

Choosing the Correct Driver	E-4
-----------------------------------	-----

Appendix F Sample Page Mode Application

Appendix G Program Scheduling

Appendix H Source Code Programs

RMTERM - Remote Terminal Download	H-1
AUTOR - Powerfail Automatic Restart	H-2
HPMDM - Modem Control	H-2
CALLB - Security Callback	H-5

Appendix I Creating a Special Character PROM

Appendix J Serial I/O Drivers (prior to Rev. 4010)

Serial I/O Device Drivers	J-1
Terminal Device Driver DD*00	J-1
Read and Write Modes	J-2
Normal ASCII Write Mode	J-2
Transparent ASCII Write Mode	J-2
Character Mode Normal ASCII Read	J-2
Character Mode Transparent ASCII Read	J-3
Block Mode Keyboard Read	J-4
DD*00 Read/Write Request	J-4
CRT Terminal Control Word CNTWD	J-4
Hard-Copy Terminal Control Word CNTWD	J-5
BUFR and BUFLN	J-6
PRAM3 and PRAM4 Optional Parameters	J-6
A- and B-Register Contents	J-6
DD*00 Control Request	J-7
Control Request CNTWD	J-7
Function Code 00: Clear and Reset Device	J-7
Function Code 06B: Dynamic Status	J-8
Function Code 11B: Line Skip/Form Feed	J-10

Function Code 20B/40B: Enable Primary/Secondary Program	J-10
Function Code 21B/41B: Disable Primary/Secondary Program	J-11
Function Code 22B: Set System Request Timeouts	J-12
Function Code 23B: Disable Asynchronous Interrupts	J-12
Function Code 25B: Inform DD*00 of Terminal Strap Change	J-13
Function Code 27B: Set User Request Timeouts	J-13
Function Code 42B: Get Character	J-14
Function Code 44B: Modify Configuration Word	J-14
Function Code 45B: Modify Trigger Character	J-15
DD*00 Status Request	J-15
Control Word CNTWD	J-15
STAT1 and STAT2 Status Parameters	J-15
STAT3 and STAT4 Status Parameters	J-16
Driver Parameter Areas	J-16
DD*00 Extended Status	J-17
CTU Device Driver DD*20	J-18
CTU Read/Write Conventions	J-18
DD*20 Read/Write Request	J-19
Control Word CNTWD	J-19
BUFR and BUFLN	J-19
A- and B-Register Contents	J-19
DD*20 Control Request	J-20
Control Word CNTWD	J-20
Function Code 00: Clear and Reset Device	J-21
Function Code 01B: Write End-of-File	J-21
Function Codes for Tape Positioning	J-21
Function Code 06B: Dynamic Status	J-22
Function Code 10B: Write End-of-File	J-23
Function Code 26B: Write End-of-Data	J-24
PRAM1 Optional Parameter	J-24
DD*20 Status Reporting	J-24
Control Word CNTWD	J-24
STAT1 and STAT2 Status Parameters	J-24
STAT3 and STAT4 Status Parameters	J-25
DD*20 Extended Status	J-25
IDM00 (HP 12040A) 8-Channel Asynchronous MUX Interface Driver	J-27
IDM00 Read Request	J-27
Control Word CNTWD	J-27
BUFR and BUFLN	J-27
PRAM3 Parameter	J-28
PRAM4 Optional Parameter	J-28
IDM00 Write Request	J-29
Control Word CNTWD	J-29
BUFR and BUFLN	J-29
IDM00 Status Request	J-30
IDM00 Control Requests	J-30
Control Word CNTWD	J-30
Function Code 6B: Dynamic Status	J-31
Function Code 23B: Control Program Scheduling	J-31
Function Code 26B: Flush Input Buffer	J-32
Function Code 30B: Set Port ID	J-32
Function Code 33B: Configure Driver Responses	J-34
Function Code 37B: Set Read Type	J-35

IDM00 Type-Ahead	J-36
IDM00 (HP 12040B/C) 8-Channel Asynchronous MUX Interface Driver	J-39
IDM00 Read Request	J-39
Control Word CNTWD	J-39
BUFR and BUFLN	J-40
PRAM3 Parameter	J-40
PRAM4 Optional Parameter	J-40
IDM00 Write Request	J-41
Control Word CNTWD	J-41
BUFR and BUFLN	J-42
IDM00 Status Request	J-42
IDM00 Control Requests	J-42
Control Word CNTWD	J-43
Function Code 6B: Dynamic Status	J-43
Function Code 23B: Control Program Scheduling	J-44
Function Code 26B: Flush Input Buffer	J-45
Function Code 30B: Set Port ID	J-45
Function Code 31B: Connect Line	J-48
Function Code 32B: Disconnect Line	J-50
Function Code 33B: Configure Driver Responses	J-51
Function Code 34B: Set Port Configuration	J-53
Function Code 36B: Set Read Length	J-54
Function Code 37B: Set Read Type	J-54
Function Code 50B: Loopback Test	J-55
Function Code 52B: Terminate Receive Buffer	J-56
IDM00 Type-Ahead	J-58
Buffer Overflow and Transmission Error	J-59
Type-Ahead and Terminal Status Request	J-60
Virtual Control Panel (VCP)	J-60
Modems	J-61
ID*00/ID*01 ASIC Interface Drivers	J-66
ID*00/ID*01 Read/Write Request	J-66
Control Word CNTWD	J-66
BUFR and BUFLN	J-67
PRAM3 Control Word Modifier	J-67
PRAM4 Optional Parameter, Read Requests Only	J-68
ID*00/ID*01 Card Control Word	J-69
ID*00/ID*01 Control Request	J-72
Function Code 00: Clear and Reset Card	J-73
Function Code 06B: Dynamic Status	J-73
Function Code 23B: Control Asynchronous Interrupts	J-76
Function Code 43B: Control Error Checking	J-76
ID*00/ID*01 Status Reporting	J-76
Control Word CNTWD	J-76
STAT1 and STAT2 Status Parameters	J-76
STAT3 and STAT4 Status Parameters	J-77
Driver Parameter Area	J-77
ID*00/ID*01 Extended Status	J-77
ID*01 Modem Capabilities	J-78
ID*01 Alarm Program Scheme	J-78
ID*01 Modem Control Requests	J-78
Function Code 31B: Activate for Modem Environment	J-79
PRAM1	J-79

PRAM2, PRAM3, PRAM4	J-80
Function Code 32B: Deactivate Modem Environment	J-82
ID*01 Error Message	J-82
ID*01 Manual Dial-Out Considerations	J-82
ID*01 Alarm Program MODEM	J-83
ID*01 Example of Modem Link Set-up	J-84

List of Illustrations

Figure 1-1	Device Table (DVT) Format	1-3
Figure 1-2	Interface Table (IFT) Table	1-3
Figure 1-3	ecode Format for Read/Write Request	1-5
Figure 1-4	cntwd Format for Read/Write Request	1-5
Figure 1-5	pram3/pram4 Format for Read/Write Request	1-8
Figure 1-6	A-Register Contents after Unbuffered Write Request	1-9
Figure 1-7	ecode Format for Control Request	1-12
Figure 1-8	cntwd Format for Control Request	1-12
Figure 1-9	pram3/pram4 Format for Control Request	1-14
Figure 1-10	ecode Format for Status Request	1-17
Figure 1-11	cntwd Format for Status Request	1-17
Figure 1-12	stat2 Format for EXEC 13 Request	1-18
Figure 1-13	Extended Status Format	1-19
Figure 2-1	DD*12 Write Request cntwd Format	2-2
Figure 2-2	DD*12 Control Request cntwd Format	2-4
Figure 2-3	DD*12 Status Request cntwd Format	2-5
Figure 2-4	stat1 and stat2 Status Words Format	2-6
Figure 2-5	DVT16 to DVT18 Extended Status Words	2-6
Figure 2-6	DVT18 Status Word Format	2-7
Figure 2-7	cntwd Format for Write Request	2-9
Figure 2-8	cntwd Format for a Control Request	2-12
Figure 2-9	pram1 Format to Change Character Set	2-14
Figure 2-10	pram1 for Print Mode Select Control Request	2-16
Figure 2-11	cntwd Format for Status Request	2-17
Figure 2-12	DVT6 Status Word (Read Into stat1)	2-17
Figure 2-13	stat3 and stat4 Formats for Z = 0 or 1	2-18
Figure 2-14	DDC12 Driver Parameters	2-18
Figure 2-15	xstat(1) to xstat(3) Status Words Format	2-19
Figure 2-16	DD*23 Control Request cntwd Format	2-22
Figure 2-17	DD*23 Control Request cntwd Format	2-23
Figure 2-18	DD*23 Status Request cntwd Format	2-24
Figure 2-19	stat1 and stat2 Status Words Format	2-24
Figure 2-20	DVT16 to DVT19 Extended Status Words	2-25
Figure 2-21	stat1 and stat2 Status Words Format	2-25
Figure 2-22	DD*24 Read/Write cntwd Format	2-27
Figure 2-23	DD*24 Control Request cntwd Format	2-28
Figure 2-24	DD*24 Status Request cntwd Format	2-30
Figure 2-25	DD*24 Status Request Parameters stat1 and stat2	2-30
Figure 2-26	DD*24 Driver Parameter Words DP4 and DP5	2-31
Figure 2-27	Extended Status Words DVT16 through DVT19	2-32
Figure 2-28	DD*36 Read/Write Request cntwd Format	2-46
Figure 2-29	DD*30 Status Request cntwd Format	2-48
Figure 2-30	DD*30 Status Request cntwd Format	2-48

Figure 2-31	DVT16 to DVT19 Extended Status Words	2-49
Figure 2-32	DVT18 Format for DD*30	2-50
Figure 2-33	DVT19 Format for DD*30	2-52
Figure 2-34	DDM30 Read/Write Request cntwd Format	2-54
Figure 2-35	DDM30 Status Request cntwd Format	2-56
Figure 2-36	stat1 and stat2 Status Words Format	2-56
Figure 2-37	DVT16 to DVT19 Extended Status Words	2-57
Figure 2-38	DVT18 Format for DDM30	2-58
Figure 2-39	DVT19 Format for DDM30	2-59
Figure 2-40	Read/Write Request cntwd Format	2-69
Figure 2-41	DD*33 Disk Control Request cntwd Format	2-70
Figure 2-42	DD*33 Status Request cntwd Format	2-71
Figure 2-43	stat1 and stat2 Status Words Format	2-71
Figure 2-44	stat3 and stat4 for Z=0	2-72
Figure 2-45	stat3 and stat4 for Z=1	2-72
Figure 2-46	DVT16 to DVT19 Status Words Format	2-73
Figure 2-47	CTD Read/Write Request cntwd Format	2-79
Figure 2-48	ID*27 Read/Write Request cntwd Format	2-92
Figure 2-49	ID*27 Status Request cntwd Format	2-94
Figure 2-50	ID*27 Status Request Parameters	2-94
Figure 2-51	ID*27 Status Parameters	2-95
Figure 2-52	DVT18 Format for ID*27	2-96
Figure 2-53	DVT19 Format for ID*27	2-97
Figure 2-54	Read/Write Control Word for ID*36	2-101
Figure 2-55	Status Request Control Word for ID*36	2-102
Figure 2-56	stat1 and stat2 Status Words Format for ID*36	2-103
Figure 2-57	DVT16 to DVT19 Extended Status Words Format for ID*36	2-103
Figure 2-58	Direct-Addressing/Auto-Addressing Mode Comparison	2-107
Figure 2-59	cntwd Format for an Auto-Addressed Read/Write	2-108
Figure 2-60	cntwd Format for an Auto-Addressed Control Request	2-109
Figure 2-61	DVT16 to DVT18 Extended Status Words Format	2-110
Figure 2-62	cntwd Format for a Direct I/O Read/Write Request	2-115
Figure 2-63	cntwd Format for a Direct I/O Control Request	2-116
Figure 2-64	DVT16 to DVT19 Extended Status Information	2-117
Figure 2-65	ID*37 Status Request cntwd Format	2-120
Figure 2-66	stat1 and stat2 Status Words Format	2-120
Figure 2-67	DVT16 to DVT19 Extended Status Words	2-121
Figure 2-68	IDR37 Read/Write Request cntwd Format	2-123
Figure 2-69	IDR37 Status Request cntwd Format	2-126
Figure 2-70	IDR37 Status Request Parameters	2-126
Figure 2-71	ID*50 Read/Write Request cntwd Format	2-128
Figure 2-72	ID*50 Control Request cntwd Format	2-129
Figure 2-73	DVT16 to DVT18 Extended Status Words	2-129
Figure 2-74	ID*50 Status Request cntwd Format	2-133
Figure 2-75	stat1 and stat2 Status Words Format	2-134
Figure 2-76	PIC Control Register Format	2-134
Figure 2-77	DVT18 Word Format for ID*50 Dynamic Status Request	2-136
Figure 2-78	ID*52 Read/Write Request cntwd Format	2-138
Figure 2-79	ID*52 Control Request cntwd Format	2-139
Figure 2-80	DVT16 to DVT18 Extended Status Words	2-140
Figure 2-81	ID*52 Status Request cntwd Format	2-142
Figure 2-82	stat1 and stat2 Status Words Format	2-142
Figure 2-83	PIC Control Register Format	2-143

Figure 2-84	DVT18 Format for ID*52 Dynamic Status Request	2-144
Figure 2-85	Serial Driver Read Request Control Word (cntwd)	2-151
Figure 2-86	Write Request Control Word cntwd	2-156
Figure 2-87	Control Request Control Word cntwd	2-158
Figure 2-88	ID400 or ID800 Revision Codes	2-164
Figure 2-89	Word Format for Code 17B Definable Terminator	2-166
Figure 2-90	Buffered Read Mode Algorithm	2-179
Figure 2-91	ENQ/ACK Handshaking	2-185
Figure C-1	Conceptual Block Diagram of Type-Ahead Mode and FIFO Mode	C-6
Figure E-1	Flowchart for Selecting Serial Interface Drivers	E-5
Figure G-1	State Diagram for Program Scheduling	G-1
Figure J-1	DD*00 Read/Write Request CNTWD Format	J-4
Figure J-2	DD*00 Write Request CNTWD for Hard Copy Terminal	J-5
Figure J-3	DD*00 Control Request CNTWD Format	J-7
Figure J-4	DD*00 Dynamic Status Format	J-8
Figure J-5	DVT18 Dynamic Status Word Format	J-8
Figure J-6	B-Register Format	J-14
Figure J-7	DD*00 Status Request CNTWD Format	J-15
Figure J-8	STAT1 and STAT2 Status Words Format	J-15
Figure J-9	Terminal Configuration Format in DVP1	J-16
Figure J-10	DD*20 Read/Write Request CNTWD Format	J-19
Figure J-11	DD*20 Control Request CNTWD Format	J-20
Figure J-12	Cartridge Tape Positioning	J-22
Figure J-13	Extended Status Words DVT16 to DVT19	J-22
Figure J-14	DD*20 Status Request CNTWD Format	J-24
Figure J-15	STAT1 and STAT2 Status Words Format	J-24
Figure J-16	DVT16 Status Word Format	J-26
Figure J-17	IDM00 Read Request Control Word CNTWD	J-27
Figure J-18	IDM00 Read Request Control Word CNTWD	J-39
Figure J-19	IDM00 Write Request Control Word CNTWD	J-41
Figure J-20	ID*00/ID*01 Read/Write Request CNTWD Format	J-66
Figure J-21	ID*00/ID*01 Card Control Word Default Format	J-68
Figure J-22	ASIC Control Word Format	J-69
Figure J-23	ID*00/ID*01 Control Request CNTWD Format	J-72
Figure J-24	DVT16 to DVT19 Status Words Format	J-73
Figure J-25	ASIC Status Format and Definition (DVT 18)	J-74
Figure J-26	ID*00/ID*01 Status Request CNTWD Format	J-76
Figure J-27	STAT1 and STAT2 Status Words Format	J-76
Figure J-28	DVT16 to DVT19 Status Words Format	J-77
Figure J-29	PRAM1 Format For Calling Sequence	J-79

Tables

Table 1-1	Read/Write Calling Sequence	1-4
Table 1-2	TR/BI Bit Combinations	1-7
Table 1-3	Control Request Calling Format	1-11
Table 1-4	Status Request Calling Format	1-16
Table 2-1	Paper Motion Request Codes	2-14
Table 2-2	Standard and Optional Language Identity Codes	2-15
Table 2-3	D- and F-Bit Error Codes	2-32
Table 2-4	Track and Logical Sector Ranges of Values	2-47
Table 2-5	DD*30 Driver Parameter Area	2-49
Table 2-6	Track and Logical Sector Ranges of Values	2-55
Table 2-7	DDM30 Driver Parameter Area	2-57
Table 2-8	DD*33 Driver Parameter Area	2-73
Table 2-9	EXEC Function and Subfunction Codes	2-81
Table 2-10	Complementary Command Array (icomp)	2-84
Table 2-11	Legal Track and Sector Values	2-93
Table 2-12	ID*27 Driver Parameters	2-95
Table 2-13	Error Codes from DVT16, Bits 0 to 5	2-96
Table 2-14	Status Codes from DVT18 (Bits 8 through 12)	2-97
Table 2-15	ID*36 Driver Parameter Area	2-104
Table 2-16	ID*37 and HP-IB Mnemonics	2-105
Table 2-17	HP-IB Universal Commands	2-122
Table 2-18	Legal Track and Sector Values	2-123
Table 2-19	IDR37 Driver Parameters	2-127
Table 2-20	Comparison of Original and Current Drivers	2-145
Table 2-21	Character Mode Transfers	2-152
Table 2-22	HP Block Mode	2-153
Table 2-23	Special Status Read Words	2-155
Table 2-24	DVT16 System Defined Errors	2-161
Table 2-25	DVT16 Driver Defined Errors	2-162
Table 2-26	DVT19 Interface Driver ID	2-163
Table 2-27	BRG Ranges	2-176
Table 2-28	Carriage Control Capabilities	2-181
Table 2-29	Device Type	2-181
Table A-1	Hewlett-Packard Character Set for Computer Systems	A-2
Table A-2	HP 7970B BCD-ASCII Conversion	A-6
Table J-1	DD*00 Driver Parameter Area	J-16
Table J-2	DVT18 Dynamic Status Word Format for DD*20 CTU Driver	J-23
Table J-3	VCP Terminal Baud Rates	J-60

I/O Request Conventions

This chapter explains the request conventions that application programs must follow in making EXEC I/O requests to RTE-A device and interface drivers. Call sequences and the parameters needed in them are described, as well as the status and error codes returned by read, write, control, and status requests.

In this manual, you will find references such as CN 11. This is a shorthand way to refer to an EXEC 3 call with subfunction 11 octal. It is described as a CN 11 call because both FMGR and CI allow you to generate EXEC 3 calls interactively by use of the CN command. For example, the CI command “CN 14 11B -1” would create and execute the EXEC call equivalent to:

```
CALL EXEC(3,ior(14,1100B),-1)
```

This is a convenient way to issue control calls without writing programs to do so. The serial drivers are commonly configured by means of CN commands in the Welcome file (WELCOME_{xx}) that is executed upon bootup.

Chapter 2 describes the RTE-A drivers in detail, and depend upon knowledge of the material in this chapter. It is important that you understand this chapter before reading the driver descriptions. This chapter describes the assembly language and FORTRAN EXEC calling sequences. In Chapter 2, only the FORTRAN calls are shown. Refer to this chapter to convert FORTRAN calls to their assembly language equivalents.

Device and Interface Driver Separation

There are two kinds of drivers: interface drivers and device drivers. Interface drivers are associated with interface cards in the A-Series computer. Device drivers are associated with peripheral devices connected to the interface cards. There is usually at least one device driver for each interface driver. Often, several device drivers share the same interface driver, because several kinds of peripheral devices can be connected to the same interface card.

Interface drivers usually process I/O requests from device drivers. Device drivers process requests from application programs and pass them on to an interface driver. If there is no device driver, or if special conditions require direct control of the interface, an application program can bypass the device driver to make requests directly to an interface driver. Whenever possible, device drivers should be used.

Device drivers, when present, format or interpret output buffers from application programs according to the requirements of a device. If there is no device driver, as in very small systems, the application program must ensure that the data buffers are already interpreted and prepared for the device before making a request to an interface driver.

Device Table (DVT)

The Device Table (DVT) is a variable-length table constructed by the system generator for each device in the system. It is the area from which the system communicates to the device driver information about the I/O request. The system uses the DVT as a storage area for list linkage words and the DVT status indicators and other system concerns. The device driver uses the DVT for device-dependant storage and as a communication area to the interface driver. Each LU used in an I/O request has an associated DVT internally identified by the operating system.

Figure 1-1 illustrates the Device Table used by a particular device, with the format of words that concern application programmers for future reference. The information in this table is posted and retrieved via system requests; there is never a need to access directly this table or the Interface Table in Figure 1-2 from an application program. Hewlett-Packard reserves the right to change the internal table constructs at any time; these tables are for convenience only. Fields are described through the following text.

Interface Table (IFT)

The Interface Table (IFT) is a variable-length table constructed by the generator for each I/O card in the system. It is primarily a storage area for system I/O concerns, although the interface driver may have a need to examine the contents.

Associated with the IFT can be one or more IFT extension words used for storage of all temporary data associated with a particular I/O card.

Figure 1-2 shows the Interface Table for a particular device, and the format of words of interest to application programmers.

Word: DVT1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DVT Link Word															
	⋮															
6	AV	Device Type						Status						E		
	⋮															
12	Device Driver Timeout Clock															
13	Interface Driver Timeout Value															
	⋮															
16	Request Parameter #1 / Error Code															
17	Request Parameter #2 / Transmission Log															
18	Request Parameter #3 / Extended Status #1															
19	Request Parameter #4 / Extended Status #2															
	⋮															
DVP1	Start of Driver Parameter Area															
	⋮															

Figure 1-1. Device Table (DVT) Format

Word: IFT1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2	Timeout List Linkage															
	Interface to Clock															
	⋮															
6	AV	Interface Type						X	I/O Select Code							
7	⋮															
IFTX	Start of IFT Extension (Storage)															
	⋮															

Figure 1-2. Interface Table (IFT) Table

Read/Write Parameter Definitions

The word formats and bit definitions of the read/write parameters are described in the following sections.

Read/Write Request Conventions

Table 1-1 contains the general calling sequences to read data from a device to a data buffer (read request), or to write data from a data buffer to a device (write request). Each EXEC parameter is described further in the following paragraphs; device-specific information is discussed in the appropriate driver sections of Chapter 2. Optional parameters (their use is dependent on the driver called) are defined in the Macro calling format comment lines and bracketed in the FORTRAN calling sequence line.

Table 1-1. Read/Write Calling Sequence

Macro Calling Format:	
EXT EXEC	
.	
JSB EXEC	
DEF NEXT	
DEF <i>ecode</i>	! EXEC Request code: 1 = READ; 2 = WRITE
DEF <i>cntwd</i>	! Control Word
DEF <i>bufr</i>	! User READ/WRITE buffer: 2 = WRITE
DEF <i>bufln</i>	! Buffer length
DEF <i>pram3</i>	! Optional request-dependent parameter
DEF <i>pram4</i>	! Optional request-dependent parameter
DEF ZERO	
DEF ZERO	! Optional: 0 = format placeholder
DEF <i>keywd</i>	! <i>keywd</i> - Locked LU keyword number
NEXT	.
.	
<i>ecode</i>	DEC <i>nn</i>
<i>cntwd</i>	DEC <i>nn</i>
<i>bufr</i>	DEC <i>nn</i>
<i>bufln</i>	DEC <i>nn</i>
<i>pram3</i>	DEC <i>nn</i>
<i>pram4</i>	DEC <i>nn</i>
<i>keywd</i>	DEC <i>nn</i>
ZERO	DEC 0
FORTRAN Calling Format:	
CALL EXEC(<i>ecode</i> , <i>cntwd</i> , <i>bufr</i> , <i>bufln</i> [, <i>pram3</i> [, <i>pram4</i> [, 0 , 0 , <i>keywd</i>]]])	

ecode

The *ecode* parameter is the Executive request code, a parameter to inform the system what type of request, read or write, is being made to an external device. The request word format is shown in Figure 1-3.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NA	NS	<i>ecode</i> (1 = Read, 2 = Write)													

Figure 1-3. *ecode* Format for Read/Write Request

An *ecode* of 1 specifies a read request; an *ecode* of 2 is a write request. Setting the NA (no-abort) bit to 1 allows error recovery if a non-severe error occurs; if NA is zero, an erroneous EXEC request will abort the calling program. See the section on EXEC Error Returns for more information.

If the NS (no-suspend) bit is set, the program is not suspended by a down device or a locked LU; however, a program can still be I/O or buffer limit suspended.

cntwd

The *cntwd* parameter is the control word that supplies information to EXEC identifying the device and supplies the control information as shown in Figure 1-4.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	x	TR	x	EC	x	BI	LU					

Figure 1-4. *cntwd* Format for Read/Write Request

BB: Identifies the device driver bypass bit; it causes the interface driver associated with the addressed LU to be called, bypassing the device driver entirely. This allows you to access directly the I/O interface programming details available at the interface driver. For most applications, this level of control is not necessary.

When writing programs that use the interface driver with no device driver defined for the LU, the BB-bit is ignored. However, if a device driver is defined for the LU being programmed, and you want to use the interface driver directly, the BB-bit must be set.

NB: Identifies the buffered/nonbuffered mode in effect. NB is 1 if nonbuffered mode is in effect, overriding normal device buffering. Programs issuing read/write requests in this mode are always suspended while the request executes. Using the nonbuffered mode allows the calling program to obtain any error and status information after the request has completed.

EXEC ignores the NB bit for a read request and forces nonbuffered operation. If a buffered read is desired, an EXEC Class I/O read request should be made. See the *RTE-A Programmer's Reference Manual*, part number 92077-90007, for information about Class I/O Read/Write and Control requests.

NB is 0 if normal device buffering is used. If a device normally uses buffered mode, the program will execute in parallel with the output operation.

UE: A user error bit.

If UE is 1, it informs the driver to not suspend a calling program if a device error occurs (such as a write request to a cartridge tape unit that does not contain a tape). After such a request, the driver will return status information in the A-Register as if the device were actually downed (see the section “A- and B-Register Contents”). The extended status words can then be examined to determine why the driver attempted to down the device. The device extended status will be discussed later in this chapter. If the UE bit is set, the calling program is expected to process the device errors that occur. The program should examine status, and error returns in the A-Register and extended status.

If UE is 0, the system provides normal error handling.

The UE bit is not functionally equivalent to the NS bit, which is described in the *RTE-A Programmer's Reference Manual*, part number 92077-90007. Setting the UE bit only instructs the system to return error information to the calling program; the program is expected to process the returned error information.

Z: is used in conjunction with optional parameters *pram3* and *pram4*.

Z is 1 if *pram3* and *pram4* are to be used to describe a control buffer name and buffer length to be passed to the driver. The buffer name (*pram3*) is an integer value specifying the location of the buffer to which the data, either words or characters, is transmitted. In FORTRAN, this is the buffer name itself. The buffer length (*pram4*) is an integer value specifying either the positive number of 16-bit words, or the negative number of 8-bit characters to be transmitted.

Z is 0 if *pram3* and *pram4* are either not defined at all, or if one or both of them will contain signed integer values to be used by the driver. Refer to the appropriate driver section for more information on how *pram3* and *pram4* are used when the Z-bit is zero.

x: is reserved for driver definition. Refer to the appropriate driver sections for detailed definitions of this bit.

TR: is 1 if transparency mode is in effect. Driver addition, or removal of special control characters is prohibited in transparency mode. TR is 0 if transparency mode is not in effect. The driver can add or delete special embedded control characters during input or output. TR/BI bit combinations are described more fully in Table 1-2.

EC: is 1 if echo mode is in effect to display keyboard input as the keys are struck. (This is the normal mode of operation for terminal devices.) This bit set to 0 suppresses echo mode.

BI: is 1 if binary data is to be transmitted, or 0 if ASCII data is to be transmitted. TR/BI bit combinations are described in Table 1-2.

LU: is the logical unit number (1-63) of the device from which data is to be read (*ecode* = 1), or to which data will be written (*ecode* = 2). If LU is specified as zero, the request is executed but no data is transferred.

TR/BI Bit Conventions: Table 1-2 lists the combinations of bits TR and BI that normally operate together for the handling of special characters and end-of-record (EOR) processing. Note that it is not necessary for all drivers to support the full set of combinations described. Some special characters are device specific and are defined in the appropriate driver section later in this manual.

Table 1-2. TR/BI Bit Combinations

		READ (<i>ecode</i> = 1)	WRITE (<i>ecode</i> = 2)
Normal ASCII			
TR 0	BI 0	<p>Process and remove special characters from input (such as DEL).</p> <p>Data transfer stops on the occurrence of CRLF in the data stream or upon an EOR signal.</p> <p>Neither the CRLF nor the EOR are read into the input buffer after they are processed.</p> <p>If an odd number of bytes is received, the right byte is filled with a space (ASCII) or null (binary).</p>	<p>If a device is not capable of enabling the EOR line, data transfer stops when the buffer limit is reached.</p> <p>A CRLF and/or EOR signal is added to the output data and processed unless the underscore character ‘_’ (137B) is the last character of the output buffer. CRLF is not added to the transmitted data in this case.</p> <p>Any other device-defined special character is processed and removed.</p> <p>If the underscore character (_) occurs anywhere else in the data before the buffer limit is reached, it is output to the device.</p>
Normal Binary			
TR 0	BI 1	<p>Special characters are not processed or removed. The driver ignores CRLF processing. It is read into the input buffer as data.</p> <p>Data transfer stops when the EOR line is enabled, where available.</p>	<p>The driver ignores processing of special characters, underscore (_) and/or CRLF; they are not removed from or added to the output buffer. If the EOR line can be enabled, it is sent on transmission of the last byte of data.</p>
Transparent ASCII			
TR 1	BI 0	<p>Process and remove the special characters CRLF from the input buffer. All other special characters are not processed or removed.</p> <p>Data transfer stops on occurrence of a CRLF or EOR signal.</p>	<p>Special character handling is provided, except for underscore (_) and CRLF processing. Therefore, CRLF is not added to the output data and EOR signaling is not provided to the device.</p>
Transparent Binary			
TR 1	BI 1	<p>Special characters, CRLF and EOR, are not processed or moved, but are passed to the input buffer.</p> <p>Data transfer stops when the buffer area is full.</p>	<p>Special character removal or processing is not provided, and CRLF or EOR signaling is not added to the output data.</p>

bufr

The *bufr* parameter is the buffer name if used in FORTRAN or Pascal; in Macro it is the address of the user buffer into which data is to be read for a read request (*ecode* = 1), or from which data is to be written for a write request (*ecode* = 2).

bufln

The *bufln* parameter is the length of the buffer, a variable defining the length of the data record to be transferred. Either a positive number is used to specify the length of the data record in words, or a negative number is used to specify the length of the data record in characters (8-bit bytes). The maximum value to which *bufln* can be set is either 16384 (words) or -32768 (characters).

If the data record contains REAL data, this must be allowed for in the *bufln* parameter. Remember that the number of words to use for *bufln* is dependent on the number of words in the data type being written. For example, integers can be either one or two words; reals can be two, three, or four words; complex can be either four, six, or eight words; and characters can be any length.

pram3* and *pram4

The *pram3* and *pram4* parameters are driver-defined optional parameters. The use of these parameters is dependent on the needs of a specific driver and on the state of the Z-bit, as shown in Figure 1-5. When the Z-bit is set to 1, *pram3* specifies the address of a control buffer and *pram4* contains the length of the buffer (positive number of words or negative number of characters). Refer to the related driver section for more information on the use of a control buffer.

	Z = 0	Z = 1
<i>pram3</i>	Optional 1-word variable	Control Buffer Name (Required)
<i>pram4</i>	Optional 1-word variable	Control Buffer Length (Required)

Figure 1-5. *pram3/pram4* Format for Read/Write Request

0 (Zero)

0 (Zero) is an optional placeholder. It must be supplied whenever the *keywd* parameter is used.

keywd

The *keywd* parameter is a keyword number, a resource number assigned by the system to locked LUs when an LU lock request is made via the LURQ system library call. A keyword number is returned after the call, and can then be supplied via the EXEC request so that a locked device can be accessed by a program other than the “owner” of the locked device. The “owner” does not have to supply the *keywd* to EXEC. See the *RTE-A Programmer’s Reference Manual*, part number 92077-90007, for more information on the LURQ system library call.

A- and B-Register Contents

End-of-operation information for read requests and unbuffered write requests is transmitted to the program in the A- and B-Registers. As shown in Figure 1-6, the A-Register contains word 6 of the device. The A-Register will be the same as *stat1* after the status request as shown in the “Status Request Conventions” section.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		Device Type						EOF	DB	EOM	SOM	SE	x	x	E

Figure 1-6. A-Register Contents after Unbuffered Write Request

- AV:** Bits 15 and 14 indicate current device availability and are used by the system for I/O control. The DS operator command can also be used to examine the availability.
- If AV is 00, the DVT is available for a new request to be initiated (the device is free to process a new request).
 - If AV is 01, the associated device has been set down by the driver or the operator. New requests will be suspended on the downed device.
 - If AV is 10, the device is busy processing an I/O request. New requests may be pending (that is, linked through word 2 of the DVT).
 - If AV is 11, the device is down, but busy with a request (such as an abort request).

Device Type:

Bits 13 through 8 are a logical 6-bit value describing the type of device associated with the current DVT. All device type values are initially established at generation. The default device types for the devices that may be included in a typical system are:

Type #	Device Type
00B	Non-HP (non-Enq/Ack) Terminals
05B	Terminals (HP 262X, 264X, 2382, 239X, 45610, Vectra)
06B	Printing Terminals
12B	Printers (HP 2608S, 2631, 2635, 256X, 268X, 293X, 2225)
20B	Cartridge Tape Unit (HP 264X)
23B	Magnetic Tape (HP 7970E)
24B	Streaming Tape Drive (HP 7974, 7978)
26B	Cartridge Tape Drive (CTD, CS/80 disks)
30B	Disk (HP 9133, 9134, 9895, 9121)
32B	Disk (HP 7906, 7920, 7925)
33B	Disk (HP 7908, 7911, 7912, 7914, 7946)
34B	Shortened-Cylinder MAC/ICD disks
35B	Multi-CPU MAC disks

Note Bits 7 through 0 may contain driver-dependent information that does not adhere to the following conventions.

- EOF:** is set to 1 if an end-of-file condition has been detected; used for mini-cartridge tapes.
- DB:** is set to 1 if the device is busy. The device is not available for other operations.
- EOM:** is set to 1 to indicate that the end of a medium has been reached, as in a disk write that requires two tracks when only one remains.
- SOM:** is set to 1 to indicate that positioning is at the start-of-medium of the recording area.
- SE:** the soft error indicator, is set to 1 if a recoverable error occurs before a device completes a tack. The E-bit is set only if the operation fails.
- x:** indicates device- or driver-dependent status bits.
- E:** the hard error indicator, is set to 1 when an error prevents a device from completing a request. When a hard error occurs, an error message is displayed on the system console, unless the UE-bit has been set (see the “Extended Device Status” section in this chapter).

If a hard error occurs, the application program can examine the AV-bit to determine whether the device has been set down or not. When the UE-bit is set in an I/O request, the E flag can also be examined by the application program. If the request was not buffered, the program can check the device status after an error calling RMPAR to retrieve DVT16, as described in the “Extended Device Status” section.

The B-Register contains the transmission log, which is the positive number of words or characters (depending upon the *bufln* specification) transmitted.

If *bufln* was negative, the B-Register contains the number of characters entered. If *bufln* was positive, the B-Register contains the number of words entered.

If the input data ends on an odd (left) byte, the last word is padded with a blank (octal 40) in ASCII mode or a null (octal 0) in binary mode. This byte is not included in the transmission log computation. In all cases, the contents of the user buffer beyond the last valid word as indicated by the transmission log is undefined. There is no guarantee that the buffer has not been altered by editing that may have occurred, neither should it be assumed that the terminator byte, if any, is there.

The A- and B-Registers are meaningless in output requests to a buffered device.

Refer to the section “EXEC Error Returns” in this chapter for a description of how to obtain the values in these registers and how to interpret the information.

Control Request Conventions

Various device driver control functions are accomplished via EXEC control requests. The general calling sequences to provide control requests to a device are shown in Table 1-2. All control requests require the first two parameters, *ecode* and *cntwd*, while the remaining parameters are optional, depending on the specific device addressed. Note that each parameter must be a single-word integer. The EXEC parameters are discussed further in the following paragraphs; specific driver information is included in the appropriate driver section.

Table 1-3. Control Request Calling Format

Macro Calling Format:	
EXT EXEC	
.	
.	
JSB EXEC	
DEF RTN	
DEF <i>ecode</i>	! EXEC Request code: 3 = CONTROL
DEF <i>cntwd</i>	! Control Word
DEF <i>pram1</i>	! Device-dependent parameter
DEF <i>pram2</i>	! Device-dependent parameter
DEF <i>pram3</i>	! Device-dependent parameter
DEF <i>pram4</i>	! Device-dependent parameter
DEF ZERO	! Optional placeholder
DEF ZERO	
DEF <i>keywd</i>	! Locked LU keyword number
RTN	.
.	
.	
<i>ecode</i>	DEC <i>nn</i>
<i>cntwd</i>	DEC <i>nn</i>
<i>pram1</i>	DEC <i>nn</i>
<i>pram2</i>	DEC <i>nn</i>
<i>pram3</i>	DEC <i>nn</i>
<i>pram4</i>	DEC <i>nn</i>
<i>keywd</i>	DEC <i>nn</i>
ZERO	DEC 0
FORTRAN Calling Format:	
CALL EXEC(<i>ecode</i> , <i>cntwd</i> [, <i>pram1</i> [, <i>pram2</i> [, <i>pram3</i> [, <i>pram4</i> [, 0 , 0 , <i>keywd</i>]]]]])	

Control Request Parameter Definitions

The word formats and bit definitions of the control request parameters are described in the following sections.

ecode

The *ecode* parameter is the Executive Request Code parameter to inform the system that a control request (*ecode* = 3) is being made to an external device. The request word format is shown in Figure 1-7.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NA	<i>ecode</i> = 3 for Control Request														

Figure 1-7. *ecode* Format for Control Request

An *ecode* of 1 specifies a read request; an *ecode* of 2 is a write request. Setting the NA (no-abort) bit to 1 allows error recovery if a non-severe error occurs; if NA is zero, an erroneous EXEC request will abort the calling program. See the section “EXEC Error Returns” for more information.

cntwd

The *cntwd* parameter is the control word. It supplies information to EXEC that identifies the device LU and the control action to be initiated on that LU. The word format is shown in Figure 1-8.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	Function Code						LU					

Figure 1-8. *cntwd* Format for Control Request

BB: (bit 15) is the device driver bypass bit.

If set to 1, it causes the interface driver associated with the addressed LU to be requested, bypassing the device driver entirely. This provides a means of directly accessing the I/O interface programming details available at the interface driver level. For most applications, this level of control is not necessary.

When writing programs that use the interface driver when no device driver is defined for the LU, the BB bit will be ignored. However, if a device driver is defined for the LU being programmed and you want to use the interface driver, the BB bit must be set.

NB: (bit 14) identifies the buffered/nonbuffered mode in effect.

If set to 1, it indicates that nonbuffered mode is in effect, overriding normal device buffering. Programs issuing control requests in this mode are suspended while the request executes.

If set to 0, normal device buffering is in effect. If a device normally uses buffered mode, the program will execute in parallel with the output operation.

UE: (bit 13) is the user-error bit.

If UE is 1, it informs the driver to not suspend a calling program if a device error occurs (such as a control request to a cartridge tape unit that does not contain a tape). After such a request, the driver will return status information in the A-Register as if the device were actually downed. The extended status words can then be examined to determine why the driver attempted to down the device. Extended status is discussed below.

UE is 0 to allow the system to provide normal error handling.

The UE bit is not functionally equivalent to the NS bit, which is described in the *RTE-A Programmer's Reference Manual*, part number 92077-90007. Setting the UE bit only instructs the system to return the error information to the calling program; the program is expected to process the returned error information.

Z: (bit 12) is used in conjunction with optional parameters *pram3* and *pram4*.

Z is 1 to signify that optional parameters *pram3* and *pram4* will be used as the buffer name and buffer length respectively for describing a control buffer to be passed to the driver.

The buffer length (*pram4*) is an integer value specifying either the positive number of 16-bit words, or the negative number of 8-bit characters (byte) to be transmitted. The buffer name (*pram3*) is an integer value specifying the location of the buffer to which the data is to be transferred. In a high-level language, this is the name of the buffer.

Z is 0 to signify that *pram3* and *pram4* are either not defined at all, or one or both of them contain signed integer values to be used by the driver. Refer to the appropriate driver section for more information.

Function Code:

is the field that determines the type of control request being made. The available general-function codes are listed below. Note that not every control function can be used for a particular driver; refer to the related driver section for specifics.

Function

Code Control Request Action in bits 11..6

00	Clear or reset device. Optional parameters may be used to indicate one of a variety of possible reset conditions.
01B	Write end-of-file.
02B	Backspace one record.
03B	Forward space one record.
04B	Rewind medium.
05B	Rewind standby.
06B	Dynamic status. Optional parameters might be given to indicate one of varying types of status. This request is always forced by the operating system for nonbuffered operation.
07B	Set end-of-medium condition.
10B	Set start-of-medium condition.
11B	Line spacing/form feed: <i>pram1</i> will indicate the number of lines to skip with a positive value, and a single form feed with any negative value.

- 12B Write gap.
- 13B Forward space one file.
- 14B Backward space one file.
- 15B Conditional form feed. Will issue top-of-form if not already at top of form.
- 16B Remote enable device.
- 17B Local enable device.
- 20B Enable primary program scheduling from driver. The optional parameters may be used to describe the program name and/or contain values to be passed to the scheduled program.
- 21B Disable primary program scheduling from driver.
- 22B Set device timeout. Optional parameter *pram1* is used to give the new timeout value for interface drivers and *pram2* contains the timeout value for device drivers, in tens of milliseconds.
- 23B Control asynchronous interrupt response. Optional parameter *pram1* is set to 0/1 for enable/disable condition.
- 24B Set the device address or subchannel. Optional parameter *pram1* describes the device address or subchannel.
- 25B Control local operation lockout. *pram1* is set to 0/1 for disable/enable condition.
- 26-27B Driver/device dependent.
- 30-37B Reserved.
- 40-57B Driver/device dependent control functions.

LU: is the logical unit number of the external device to which the control request is made. If LU is specified as zero, the request is executed but no data is transferred.

pram1* through *pram4

Parameters *pram1* through *pram4* are driver defined optional parameters. The use of these parameters is dependent on the needs of a specific driver and the control function code used. The format is shown in Figure 1-9.

When the Z-bit is set to 1, *pram3* and *pram4* describe a control buffer to be used by a driver. *pram3* is the buffer name, and *pram4* is the buffer length (positive number of words or negative number of characters). Refer to the related driver section for more information concerning the use of a control buffer.

	Z = 0	Z = 1
<i>pram3</i>	Optional 1-word variable	Control Buffer Name (Required)
<i>pram4</i>	Optional 1-word variable	Control Buffer Length (Required)

Figure 1-9. *pram3/pram4* Format for Control Request

0 (Zero)

Zero is an optional (formal) placeholder that must be supplied whenever the *keywd* parameter is used.

keywd

The *keywd* parameter is a keyword number, a resource number assigned by the system to a locked LU after the LU is locked via the LURQ system library call. This keyword number can be supplied in the EXEC request so a locked device can be accessed by other than the owner of the locked device (its owner does not have to supply the *keywd* to EXEC.) Refer to the *Programmer's Reference Manual* for more information on the LURQ call.

A- and B-Register Contents

End-of-operation status information for an unbuffered control request is returned in the A- and B-Registers.

The A-Register is set to word 6 of the device table (DVT) containing the device status. See Figure 1-6 and the previous section on A- and B-Register Contents (under Read/Write Requests) for the format and meaning of the A-Register.

The B-Register returns any information posted by the driver in DVT 17. For most control requests, this value will be the transmission log (zero).

Status Request Conventions

The status of I/O operations is maintained in device (DVT) and interface (IFT) tables for each device in the system. The status should be considered a snapshot of an I/O operation, since the state of I/O circumstances is constantly changing.

The general calling sequences for a status request to a device are shown in Table 1-4. All status requests require the first three parameters, *ecode*, *cntwd*, and *stat1*; the remaining parameters are optional, depending on the needs of the particular device addressed. Each EXEC parameter is described further in the following paragraphs. Driver-specific information is discussed in the related driver sections.

Table 1-4. Status Request Calling Format

Macro Calling Format:	
EXT EXEC	
.	
.	
JSB EXEC	
DEF RTN	
DEF <i>ecode</i>	! EXEC Request code: 13 = STATUS
DEF <i>cntwd</i>	! Control Word
DEF <i>stat1</i>	! Device status return from DVT word 6
DEF <i>stat2</i>	! Optional; status return from IFT word 6
DEF <i>stat3</i>	! Optional status
DEF <i>stat4</i>	! Optional status
RTN	.
.	
.	
<i>ecode</i>	DEC <i>nn</i>
<i>cntwd</i>	DEC <i>nn</i>
<i>stat1</i>	DEC <i>nn</i>
<i>stat2</i>	DEC <i>nn</i>
<i>stat3</i>	DEC <i>nn</i>
<i>stat4</i>	DEC <i>nn</i>
FORTRAN Calling Format:	
CALL EXEC(<i>ecode</i> , <i>cntwd</i> , <i>stat1</i> [, <i>stat2</i> [, <i>stat3</i> [, <i>stat4</i>]]])	

Status Parameter Definitions

The word formats and bit definitions of the status parameters are described in the following sections.

ecode

The *ecode* parameter is the executive request code, a parameter to request (*ecode*=13) status information about an external device. The request word format is shown in Figure 1-10.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NA	<i>ecode</i> = 13 for Status Request														

Figure 1-10. *ecode* Format for Status Request

An *ecode* of 1 specifies a read request; an *ecode* of 2 is a write request. Setting the NA (no-abort) bit to 1 allows error recovery if a non-severe error occurs; if NA is 0, an erroneous EXEC request will abort the calling program. Refer to the section on EXEC Error Returns for more information.

cntwd

The *cntwd* parameter is the control word that supplies the logical unit number of the device from which status is requested. The word format is shown in Figure 1-11.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0						LU					

Figure 1-11. *cntwd* Format for Status Request

- Z:** is used in conjunction with optional parameters *stat3* and *stat4*.
 Z is 1 if optional status parameters *stat3* and *stat4* are to be used to supply the address (or name) and length of a buffer to return the Driver Parameter Area.
 Z is 0 if the first word of the driver parameter area is to be returned in *stat3*, and/or the second word of the \$DVTP area is to be returned in *stat4*.
- LU:** is the logical unit number of the external device for which the status request is made. If LU is specified as zero, the LU 0 bit-bucket status is returned.

stat1

The *stat1* parameter is the device status parameter, the only required parameter. It will contain the device status from word 6 of the device table (DVT). See the explanation of Figure 1-6, A-Register Contents after Unbuffered Write Request.

Note *stat1* is a decimal number unless otherwise indicated. It is a system supplied status that applies to all drivers.

stat2

The *stat2* parameter is an optional interface status parameter that defines the characteristics of the interface card associated with device LU requested, obtained from word 6 of the interface table. The *stat2* word format is shown in Figure 1-12.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		Interface Type						x		I/O Select Code					

Figure 1-12. *stat2* Format for EXEC 13 Request

AV: indicates interface availability (the current status of the I/O interface) and is used by the system for I/O control:

00 = Interface is free; no operation is in progress.

01 = Interface is locked to a device driver for future operation (is not busy with request, but locked).

10 = Interface is busy with device driver request, but not locked.

11 = Interface is locked and is busy with a request.

Interface Type:

is a logical 6-bit value used to describe the I/O interface card to which a device or a set of devices are connected. All interface type values are initially established at generation.

Typical interface types are:

Type #	Interface Type
00	Asynchronous Serial Interface or MUX Card
01	Asynchronous Serial Interface or MUX with modem support
36	PROM Storage Device
37	HP-IB Interface Card
50	Parallel Interface Card
52	CPU-CPU Link

x: is reserved.

I/O Select Code:

is the select code by which the particular interface card is addressed. This is set on the card itself by switches and is specified at generation time.

stat3 and *stat4*

The *stat3* and *stat4* parameters return optional status. When these parameters are supplied in the status request sequence, and bit 12 (Z-bit) of *cntwd* is zero, the first word of the driver parameter area of the DVT is returned in *stat3* and the second word of the driver parameter area is returned in *stat4*.

If bit 12 *cntwd* is set to one, *stat3* and *stat4* supply the driver with the address and the length of a buffer to return the driver parameter area. Refer to the related driver section for a description of the contents of the driver parameter area.

Extended Device Status

In addition to the status information provided through the EXEC 13 request, an extended device status is also available, depending on the nature of the device driver. The extended status can be recovered by a call to RMPAR immediately after a nonbuffered Read, Write, or Control request. Refer to the *Programmer's Reference Manual* for the RMPAR calling sequence.

At the completion of a nonbuffered I/O request, DVT16 through DVT19 of the DVT are placed into words 2 through 5 of the calling program ID segment. RMPAR is thus able to retrieve these words to be placed in a buffer reserved for the extended status. Since words 2 through 5 of the ID segment are used as temporary storage, the validity of the information is maintained only as long as the calling program does not cause the temporary words to be disturbed (that is, until the next EXEC request). This can be guaranteed if RMPAR is called immediately following the I/O request. The calling sequence to retrieve the extended status is:

```

INTEGER XSTAT(5)
.
.
N = 1 or 2 or 3
CALL EXEC(N,CNTWD[ ,PRAM1[ ,PRAM2[ ,PRAM3[ ,PRAM4[ ,0,0,KEYWD]]]])
CALL RMPAR(XSTAT)
.
.
.

```

The extended status information returned in the 5-word array XSTAT is:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
XSTAT(1)	x	x	0						Error Code						DVT16		
XSTAT(2)	Transmission Log																DVT17
XSTAT(3)	Driver Defined Extended Status																DVT18
XSTAT(4)	Driver Defined Extended Status																DVT19
XSTAT(5)	Undefined																

Figure 1-13. Extended Status Format

x: indicates reserved bits.

Error Code:

is a 6-bit error indication describing the particular type of device error detected. Note that when any of these bits are set, the error bit (bit 0) of the device status word returned in the A-Register is also set. The error codes are:

Error	Meaning
00	No Error
01	Illegal Request
02	Not Ready
03	Timeout
04	End of Tape
05	Transmission Error (parity)

06	Write Protected
07	Addressing Error (HP-IB)
08	Serial Poll Failure (HP-IB)
09	Group Poll Failure
10	Fault (such as disk)
11	Data Communication Error
12	Generation Error (Check DVTX or DVTP)
13-19	Reserved
20-59	Driver Definable Error Condition (see the related driver manual)

Transmission Log:

is a value returned by the driver, indicating the positive number of words or characters actually transferred in a nonbuffered Read, Write, or Control request. (The next byte after the length in the transmission log should be ignored.) The value (words or characters) is determined by what was specified in the original EXEC request. Note that the same information is returned in the B-Register after any nonbuffered I/O request.

Extended Status:

is the device-dependent status information from a driver or device. Refer to the related driver section for details.

EXEC Error Returns

The following errors cause an unconditional program abort.

Error Code	Error Type
PE	CPU Memory Parity Error
MP	Memory Protect
RQ	Request Code
SR	EXEC call executed in a privileged subroutine
LD	Segment load failed

With the following errors, the programmer has the option of causing the program to abort or reporting the error and allowing the program to continue execution.

Error Code	Error Type
SC	Scheduling
LU	LU Lock
IO	Input/Output Error
RN	Resource Number
CL	Class I/O
OP	Option Error

Errors specific to I/O operations are:

IO00	Illegal class number.
IO01	Not enough parameters or illegal parameter in I/O call.
IO02	Illegal logical unit in I/O call.
IO04	Illegal buffer address in I/O call.
IO07	Driver has rejected call.
IO10	Illegal class GET.

If an error occurs, the A-Register is set to the ASCII error type (LU, SC, IO, RN, CL, OP) and the B-Register is set to the ASCII equivalent of the error number (ASCII 01, 02, 03, etc.).

In Assembly Language, the A- and B-Registers can be accessed directly to determine the error information. In high-level languages, use the ABREG subroutine to retrieve the contents of the A- and B-Registers after a non-severe EXEC error. The first parameter passed to ABREG returns the value of the A-Register; the second parameter returns the contents of the B-Register. The calling program can then implement the appropriate error processing.

If an EXEC request is successful, the A- and B-Registers contain information about the status of the call. ABREG also can be used to retrieve this information upon successful EXEC completion. Note that status information after successful EXEC requests is meaningful only for nonbuffered requests.

***ecode* No-Abort Bit**

The no-abort (NA) option alters the return point of the EXEC call, and is enabled by setting bit 15 of the *ecode* request code to 1. If an error occurs, the line immediately following the CALL EXEC line is executed (which must be a one-word instruction). If there is no error, the second line of code following the CALL EXEC is executed.

The following excerpt of a sample FORTRAN program demonstrates the use of the special error return:

```
      IMPLICIT INTEGER (A-Z)
      CALL EXEC (ECODE+100000B, ....)    ! set no-abort bit
      GO TO 777                          ! executed only if EXEC error
10    CONTINUE                          ! normal return point
      .
      .
      CALL EXEC (IOR(100000B,ECODE),...,*777)
      .
      .
*    exec error handling section
777  CALL ABREG (A, B)                   ! A & B contain error codes
      WRITE (1, '(A2," ",A2, " error in EXEC ", I2, " call.>")')
      >  A, B, ECODE
      .
      .
      .
```

To use the no-abort option with Pascal, add -32768 to *ecode*. The instruction following the EXEC call should be a call to a procedure without a formal argument list. Thus, the variables containing the A- and B-Register values should be global to the error-handling procedure and the program module in which EXEC was called.

An example of using the no-abort option from Pascal:

```
Program FRAGMENT;

CONST
    bell          = #7;
    noabort       = -32768;

TYPE
    ASCWD         = PACKED ARRAY [1..2] OF CHAR;
    EXEC_CALL     = 1..28;
    LU            = 0..63;
    STRING26      = PACKED ARRAY [1..26] OF CHAR;

VAR
    ecode :EXEC_CALL;
    crt   :LU;
    str   :STRING26;

procedure ABREG                               { Retrieve register contents }
    ( VAR a, b: ASCWD ); EXTERNAL;

procedure REPORT_EXEC_ERROR; $DIRECT$        { EXEC error handling }
LABEL 1;
VAR a, b: ASCWD;
BEGIN

    abreg (a,b);                               { get register values }
    str := '          EXEC error [    ].';      { build }
    str[20] := a[1]; str[21] := a[2];          { message }
    str[22] := b[1]; str[23] := b[2];          { and }
    str[26] := bell;                            { display }
    EXEC (2+noabort, crt, str, -26); GOTO 1; 1: ;
                                                { do nothing if another exec error }

END;

BEGIN { main program }
.
.
.
EXEC (ecode + noabort, ....); { set no-abort bit }
REPORT_EXEC_ERROR;           { executed only if error occurs }
.
.
.
END. { main program }
```

Device Model Numbers for System Generation

The device model numbers that can be specified during system generation for the drivers described in this manual are given in the *RTE-A System Generation and Installation Manual*, part number 92077-90034.

Device and Interface Drivers

This chapter describes the device and interface drivers supported on an RTE-A Operating System. The device drivers are listed first, followed by the interface drivers. Within each of these categories, the drivers are listed in numeric order. The only exception is that the Serial I/O drivers (DDC00, DDC01, ID800, ID801, ID400, ID100, and ID101) are documented under the Serial I/O Driver heading at the end of this chapter. (Appendix J documents the older Serial I/O drivers prior to Revision 4010: DD*00, DD*20, IDM00, ID*00, and ID*01.)

Specific programming considerations when using EXEC calls with each driver are covered here; for general programming considerations, refer to Chapter 1.

DDC12

HP-IB Line Printer Driver DD*12

DD*12 is a line printer device driver designed to drive the HP 2631 and HP 2932A/33A/34A Line Printers with an HP-IB Interface.

DD*12 provides the software interface for the device, while the HP-IB interface card driver, ID*37, performs the actual I/O instructions to the interface card. Access to DD*12 is achieved via standard EXEC requests as discussed in the following sections.

DD*12 Write Request

The call sequence for the write request is

```
CALL EXEC( 2 , cntwd , bufr , bufln )
```

Control Word *cntwd*

Figure 2-1 shows the format of the control word (*cntwd*) used by DD*12 for an EXEC read or write request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	0	TR	0	CC	0	LU						

Figure 2-1. DD*12 Write Request *cntwd* Format

NB, UE: the non-buffered and user error bits are as defined in the Read/Write Parameter section in Chapter 1.

TR: Transparency bit.

If the TR bit and BI bit are both set to zero, the driver processes data in transparent ASCII mode. In this mode, you must explicitly provide carriage return (CR, 15B), line feed (LF, 12B), and form feed (FF, 14B) controls in the output buffer. (A form feed embedded in the output buffer causes a carriage return before the form feed executes; a line feed (LF) embedded in the buffer does not cause a carriage return before the LF executes.)

The driver does not interpret the CC bit (bit 7) when in transparent ASCII mode.

Refer to the DD*00 Read and Write Modes section in this chapter for more information about normal and transparent ASCII write conventions.

CC: Carriage Control bit.

If this bit equals 1, single-spacing is used when the driver outputs the entire data buffer to the printer.

If this bit equals 0, the driver interprets the first character of the output buffer as a carriage control character; this character is not output to the printer. Output to the

printer starts with the second character. The recognized carriage control characters are:

Char	Action
0	Double Spacing
1	Eject One Page
*	Suppress Spacing (Overprint); causes the driver not to issue a line feed following the carriage return after printing the buffer contents. The default is to advance to the next line.

All other characters cause single spacing.

LU: as defined in the Read/Write Parameter section in Chapter 1.

BI (Binary data) bit 6 must be set to zero because the driver requires that the output buffer contain ASCII information.

bufr and bufln

The I/O buffer containing ASCII data used in the EXEC request is described by parameters *bufr* and *bufln* as described in Chapter 1 under the Read/Write Parameter Definitions section.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each nonbuffered EXEC write request. The format of word 6 is the same format as *stat1* returned after an EXEC 13 status request discussed below.

The B-Register will contain the positive number of words or characters that were transmitted during the last nonbuffered write request.

DD*12 Control Request

The call sequence for the control request is

```
CALL EXEC(3, cntwd[, pram1])
```

Control Word *cntwd*

Figure 2-2 shows the format of the control word (*cntwd*) used by DD*12 for a line printer control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	Function Code						LU					

Figure 2-2. DD*12 Control Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit definition) bits are defined in the Control Request Parameter Descriptions section in Chapter 1.

Function Code 00: Clear and Reset

Function Code 00 clears control to the HP-IB interface card. This causes the interface to terminate any printer operation and resets the printer parameters as:

- Print mode set to normal.
- Spacing set to the switch setting on the printer.
- Horizontal tabs cleared.
- Display functions off.
- View mode on.

Function Code 11B: Line Skipping and Form Feed

The value in *pram1* specifies a form feed or the number of lines to be skipped when Function Code 11B is given:

- | | |
|----------|--|
| negative | 1 Page eject/Top of form. |
| 0 | Suppress line feed on next write operation only. |
| 1-63 | Carriage return + 1 through 63 line feeds. |

If any Vertical Forms Control (VFC) channels must be controlled, the escape sequences controlling them may be passed to the driver via write requests, with the escape sequences included in the output buffer described by *bufnr* and *bufln*. Other printer control options may be passed in the same manner. Refer to the *HP 2631 Operator's Reference Manual* for the allowed escape sequences.

The following example will issue a top-of-form and reset, then clear the line printer at LU 6, space down 10 lines and set the line spacing to 3 lines per inch (LPI).

As described in the device operator's reference manual, setting the spacing to 3 lines per inch requires the sequence:

ESC&l3 where: ESC = 33B; 015400B in high byte.
l = 154B; 066000B in high byte.

The code to configure the output format, then, is

```

IMPLICIT INTEGER (A-Z)
DIMENSION BUFR (2)
.
.
C ISSUE TOP-OF-FORM TO THE LINE PRINTER AT LU 6.
  CALL EXEC ( 3, 6 + 1100B, -1 )
C
C RESET AND CLEAR LU 6
  CALL EXEC ( 3, 6 )
C
C SKIP 10 LINES
  CALL EXEC ( 3, 6 + 1100B, 10)
C
C SET LINE SPACING TO 3 LINES/INCH WITH ESC&13
BUFR (1) = 015400B + 1H&
BUFR (2) = 066000B + 1H3
BUFLN = -4 CALL EXEC ( 2, 6+2000B, BUFR, BUFLN)
.
.
END

```

DD*12 Status Request

The call sequence for the status request is

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4]]])
```

Control Word *cntwd*

Figure 2-3 shows the control word (*cntwd*) format for an EXEC status request driven by DD*12.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0						LU					

Figure 2-3. DD*12 Status Request *cntwd* Format

stat1 and stat2 Status Parameters

The format of the returned status parameters *stat1* and *stat2* is shown in Figure 2-4.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type = 12						0			EOM	0			E
<i>stat2</i>	AV		Interface Type = 37						0			I/O Select Code				

Figure 2-4. stat1 and stat2 Status Words Format

For *stat1*, the AV (device availability) bits and device type are as defined in the Status Request Parameter Definitions section of Chapter 1.

End-of-Media (EOM) is set to 1 if the printer runs out of paper, and the message “I/O NR LUxx D, F” is displayed at the system console. Load paper in the printer and put it back online. Do not reset the printer or the data in its buffer will be lost. The printer is automatically set up, and the printing continues.

Bit 0, the error flag bit, indicates an error code in DVT16.

For *stat2*, the interface type is 37B, corresponding to the HP-IB interface card. All other bits are defined in the Status Parameter Definitions section of Chapter 1.

stat3 and stat4 Status Parameters

If the Z-bit of the control word (*cntwd*) is zero, *stat3* returns the first word of the driver parameter area. If the Z-bit is set to 1, *stat3* names the buffer to return the driver parameter area, and *stat4* is the length of the *stat3* buffer. The only words defined in the driver parameter area are the HP-IB address of the line printer and a second word of all zeros.

DD*12 Extended Status

As discussed in the Extended Status section in Chapter 1, the extended status can be obtained by making a call to RMPAR after an unbuffered EXEC read/write or control request to determine the status of the request. The format of the returned status is shown in Figure 2-5.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	X		0						Error Code							
DVT17	Transmission Log = 0															
DVT18	Extended Status															

Figure 2-5. DVT16 to DVT18 Extended Status Words

For DVT16, bits 14 and 15 are used by the system. The octal error codes in bits 0 through 5 are:

- 0 = Normal request completion.
- 1 = Illegal request.
- 2 = Device not ready.
- 3 = Device has timed out.
- 5 = Transmission error/parity error.

DVT17 is the transmission log posted with the original length (or zero if error). DVT18 contains the extended status. The word format is shown in detail in Figure 2-6.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ONL	RDY	0	PE	X	PO	PFD	

Figure 2-6. DVT18 Status Word Format

The significance of each bit, when set, is

- ONL = Printer is online.
- RDY = Printer is ready to receive data.
- PE = Parity error.
- PO = Paper out.
- PFD = Printer powerfail detected.

HP-IB Printer Driver DDC12

DDC12 is the device driver for HP 2608S and 2563A/64A/65A/66A Line Printers equipped with HP-IB. DDC12 communicates with the printer through ID*37, the HP-IB interface driver.

These are medium to high-speed graphics printers with 16 available languages plus a graphics mode. They can be controlled by escape sequences or control codes within the data sent to the printer. To ensure that the HP 2608S printer responds properly to DDC12, the date code of the control board must be 2237 or later. Check the date code by pressing the self-test button on the front panel. The code is printed with the test output. If the date code is not 2237 or later, the printer can be updated by an HP service representative.

Programs that use driver DD*12 can also use DDC12, but with some exceptions. DD*12 drives the HP 2631 printer. When they have the same printing functions, the HP 2631 and 2608S printers use the same control codes and escape sequences, so data for either printer is compatible with the other if it uses functions common to both. Refer to the printer technical manual for a list of functions and the escape sequences and control codes that activate the printer. The drivers also interpret the *pram1* parameter in control request 11B differently. This difference is explained in the DDC12 Control Requests section (control request 11B).

Bringing the Printer Up Automatically

DDC12 automatically performs a system UP command as soon as the printer is turned on and ready. It is not necessary to issue the UP command in response to an IO-NR message. All that is required is to turn the printer on, make sure it is ready to run, and press the ONLINE switch. If an IO-TO occurs because of a power failure, DDC12 brings the printer UP as soon as power is restored. If the printer is disconnected from the system but not turned off, DDC12 cannot automatically bring it up. In this case, the UP command must be executed to bring the printer up.

Powerfail Recovery

DDC12 automatically restores the following conditions after the printer power fails and is restored, or after the printer is manually reset by pressing the ONLINE/OFFLINE and FORM ADJUST/FORM LENGTH buttons simultaneously.

1. Print mode with transparency on or off
2. Left margin definition
3. Primary and secondary character sets
4. 6 or 8 LPI selection

If any of the above four conditions is altered by escape sequences, control codes, or via the printer front panel, the driver cannot restore the condition correctly. All other printer conditions are set to the values selected by the POWER CONDITIONS switches on the power supply front panel. Refer to the printer technical reference manual for a list of the POWER CONDITIONS switch settings. Because of the automatic recovery feature, the above conditions cannot be cleared by manually resetting the printer. They can be cleared by any one of the following:

1. Reprogramming them with a control request
2. Issuing a clear control request
3. Rebooting the system

Up to five lines of text can be lost because of a power failure. During normal printing, the printer buffer accepts several complete records and signals that they have received before printing begins. If the printer power fails or if the printer is manually reset, the printing process stops immediately. The lines stored in the printer's internal buffer are not printed, and are therefore lost when the driver recovers from the reset or power loss.

DDC12 Write Request

The call sequence for a write request is:

```
CALL EXEC( 2 , cntwd , bufr , bufln )
```

Control Word *cntwd*

Figure 2-7 shows the format of *cntwd* used by DDC12 for a write request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	HON	VFC	0	V	0	LU						

Figure 2-7. *cntwd* Format for Write Request

BB, NB, UE: The bypass, non-buffered, and user error bits are as defined in the Read/Write Parameter section in Chapter 1.

HON: Honesty mode bit. The honesty bit turns driver carriage control on and off. It is called the honesty bit because when it is on, the driver “trusts” that the data sent to the printer contains escape codes or control codes for carriage control.

0 = Honesty mode off. The driver controls paper motion normally. Paper motion can also be controlled by escape sequences and control codes in *bufr*.

1 = Honesty mode on. The contents of *bufr* are printed without any carriage control by the driver. Paper motion must be controlled by escape sequences or control codes in *bufr*, or by paper motion control requests.

If the last character of *bufr* is an underscore (`_`) and the honesty mode is off, the underscore is interpreted as a line continuation request. The contents of *bufr* are sent to the printer, but are not printed until the printer receives the next *bufr*. The contents of both buffers are then printed as one line.

Carriage control is determined by one or more of the following:

1. The current print mode (refer to the DDC12 Control Request section, control code = 30B).
2. The auto page eject flag.
3. The honesty bit.
4. The V-bit and the first character in *buf*r.
5. The last character in *buf*r.
6. Escape sequences or control codes in *buf*r.
7. Paper motion control requests (refer to the DDC12 Control Request section, control code = 11B.)

VFC: Vertical Forms Control (VFC) Definition bit:

- 0 = Normal write request. Print the contents of *buf*r.
- 1 = VFC definition request. *buf*r contains VFC table definition data. Refer to the DDC12 VFC Definition Request section for more information.

V: The V-bit. Turns carriage control by the first character of *buf*r on and off. The V-bit is ignored in the graphics mode. Unless paper motion is specifically requested with a control request, the printer advances the paper one dot row after printing each row.

- 0 = Interpret the first character of *buf*r as a carriage control request, and print the rest of *buf*r. The first character is interpreted according to the following definitions:

Blank Single space. Print the contents of *buf*r, followed by a Vertical Forms Control (VFC) channel 3 request if the auto page eject flag is set, or a carriage return and line feed if it is not.

- 0 Double space. Perform a VFC channel 3 if the auto page eject flag is set, or a carriage return/line feed if it is not, then print the contents of *buf*r and perform another VFC channel 3 or a carriage return/line feed.

- 1 Eject current page with a VFC channel 1, then print the contents of *buf*r, followed by a VFC channel 3 if auto page eject is set, or a carriage return/line feed if it is not.

- * No paper motion before or after printing the contents of *buf*r (overprint).

- + Same as *, overprint.

Other Same as definition for blank character.

- 1 = Print the contents of *buf*r according to the current print mode, followed by either a VFC channel 3, if the auto page eject flag in the printer is set, or a carriage return/line feed if it is not.

LU: Logical unit number of the printer.

bufr* and *bufln

The *bufr* parameter is either the name of the data buffer to be printed, or the address of the first byte or word in the data buffer. Refer to the Read/Write Parameter Definitions section of Chapter 1.

The *bufln* parameter is the length of *bufr*. If *bufln* is:

- < 0 The length of *bufr* given in bytes (characters).
- = 0 Nothing is printed, but carriage control occurs according to the current print mode and the V-bit, honesty bit, and auto page eject flag settings.
- > 0 The length of *bufr* given in words.

The printer can print up to 132 standard-size or 66 double-size characters on a line. It can fit up to 115.5 bytes of graphics data on a row. *bufr* should not contain more characters or data than the printer can fit on a line or dot row. Extra characters or graphics data are lost.

If, however, *bufr* contains escape sequences or control codes for carriage control, *bufr* may contain more characters or data than can fit on a line. The printer buffer can receive up to 1024 bytes of data. Thirty bytes of the buffer are reserved for driver protocol data, so *bufr* may contain as many as 994 bytes of characters or graphics data with carriage control codes. DDC12 rejects write requests with *bufln* greater than 994 bytes (497 words).

The driver does not check or modify the print mode before printing the data in *bufr*. Your software must keep track of the current print mode to ensure that the output appears as desired. The print mode is selected by escape sequences in *bufr* or control requests.

DDC12 VFC Definition Request

When bit 9 (the VFC definition bit) of *cntwd* is set, the write request becomes a VFC definition request. The calling sequence is the same as the write request, but the contents of *bufr* are interpreted as described below. No printing occurs with a VFC definition request.

bufr* and *bufln

The *bufr* parameter is either the name of the buffer that contains the VFC table definition, or the address of the first word in the buffer. (Refer to the Read/Write Parameter Definitions section of Chapter 1.) The buffer must contain exactly the number of words specified by Bits 0 - 6 of *bufln*.

The VFC table is explained in the printer technical reference manual. Each word in *bufr* corresponds to one line of the VFC table. Bits 0 - 15 of each word correspond to VFC channels 1 - 16. Before attempting to define the VFC table, read and understand the applicable parts of the printer manual.

1. Primary and secondary character set languages
 2. 6 or 8 LPI line density
 3. Standard VFC
- e. The printer buffer is cleared, and the left margin is set to 0 (printing begins in column one).
 - f. The primary character set restored in step d is selected as the current printing language.
 - g. The paper is advanced to the top-of-form.

06B = Dynamic status request. This request causes the current printer status to be loaded into DVT18, the device table status word. The status word is explained in the DDC12 Status Request section. The printer status can be read by the RMPAR system routine.

11B = Paper motion request (requires *pram1*). Any of 77 different paper motion requests can be made with this request. The *pram1* values for each request are listed in Table 2-1. Software that calls DD*12 is compatible with DDC12, except that the drivers vary in their interpretation of *pram1* in this request. DD*12 interprets *pram1* values between 56 and 63 as requests to advance that many lines, while DDC12 interprets those values as VFC channel requests.

Table 2-1. Paper Motion Request Codes

<i>pram1</i>	VFC Channel	Description
< -2	1	Top of form
-2	0	Conditional top of physical page
-1	1	Top of form
0		Suppress space on next operation
1-55	3	Advance 1-55 lines
56	4	Skip to next single space line
57	5	Skip to next double line
58	6	Skip to next triple line
59	7	Skip to next half page
60	8	Skip to next quarter page
61	2	Skip to next tenth line
62	1	Skip to bottom of form
63		Skip to top-of-form
64		Set auto page eject mode (default)
65		Clear auto page eject mode
66	9	Skip to bottom of form
67	10	Skip to line before bottom of form
68	11	Skip to line before top of form
69	12	Skip to top of form
70	13	Skip to next seventh line
71	14	Skip to next sixth line
72	15	Skip to next fifth line
73	16	Skip to next fourth line
> 73		Not defined (ignored)

15B = Change character set languages (requires *pram1*). The primary and secondary character set languages can be programmed with this request. *pram1* specifies the languages for both character sets, as shown in Figure 2-9.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							Character Set 2					Character Set 1			

Figure 2-9. *pram1* Format to Change Character Set

Character Set 2 and Character Set 1 are the octal codes for the language selected from Table 2-2 as the secondary and primary character sets, respectively.

Table 2-2. Standard and Optional Language Identity Codes

Octal Code	Language Code
00	Standard Language Set USASCII
	Optional Language Set 001
01	ARABIC
02	CYRILLIC
03	KATAKANA
04	DRAW
	Optional Language Set 003
05	BLOCK
06	MATH
	Optional Language Set 002
07	APL
10	FRENCH
11	GERMAN
12	SWEDISH/FINNISH
13	DANISH/NORWEGIAN
14	SPANISH
15	BRITISH
16	JAPANESE ASCII
17	ROMAN EXTENSION SET
Any request for a language not installed in the printer defaults to USASCII.	

16B = VFC reset (requires *pram1*). The standard VFC is selected, and the line density is set to 6 or 8 LPI with this request. If the standard VFC is already selected, then this request sets the line density. If the VFC has been defined by a VFC definition, the standard VFC is restored, the line density is set to 6 or 8 LPI, and the current paper position becomes the TOF. *pram1* selects the line density:

pram1: 0 = 6 LPI
 1 = 8 LPI

20B = Self Test. Requests execution of the printer self-test, which takes about 25 seconds and prints a page of test output. Unless the printer fails the test, all operating conditions are restored as before the test. To read the test results from the device table, wait until the test finishes, and perform a dynamic status request, as described in the DDC12 Extended Status Request section.

21B = Left margin definition (requires *pram1*). This request defines a left margin of zero to 131 spaces. The decimal value of *pram1* determines the width of the margin, in standard-size columns. For example, if *pram1* = 10, the first ten columns are blank, and printing begins in column 11.

In the double-size mode, the margin is set to the same width as in the standard-size mode, except that if *pram1* is odd, the margin width is rounded up to the next even

value so that the double-size characters line up vertically. For example, if *pram1* = 9, the first ten columns are blank, and the first double-size character is printed in columns 11 and 12.

The margin setting is ignored in the graphics mode.

22B = Set timeout (requires *pram1* and *pram2*). This request sets both the interface timeout and the device timeout. Put the desired interface timeout in *pram1* and the desired device timeout in *pram2*, both values expressed in tens of milliseconds. For no timeout, set the parameter to zero.

Using certain functions on the HP 2563A and 2566A may require that the timeout be set larger than the generation-time default of five seconds. If the device timeout is too small, I/O not ready errors may occur, even though the printer is online and operating normally. If the largest device timeout (11 minutes) is not sufficient, then set the timeout to zero (no timeout).

Consult the operator's manual for the HP 2563A or 2566A for further information.

30B = Print Mode Select (requires *pram1*). This request sets the print mode and turns the transparency mode on or off. The driver restores the print mode setting if the printer power fails or if it is reset. The format of *pram1* is shown in Figure 2-10.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											TRN	Print Mode			

Figure 2-10. *pram1* for Print Mode Select Control Request

TRN = Transparency mode: 0 = off
1 = on

When transparency mode is on, no control codes or escape sequences sent to the printer are executed. Characters are printed normally, and escape sequences and control codes are displayed as special symbols.

Print Mode = 00B	Standard-size (default)
01B	Double-size
02B	Graphics
03B	Compressed size
04B - 17B	Reserved (defaults to standard)

Standard size. All characters are printed at 10 characters per inch. Up to 132 characters can be printed per line, at either 6 or 8 LPI.

Double-size. All characters are printed at twice the normal width and height. Up to 66 characters can fit on each line, at either 3 or 4 LPI. VFC channel requests are independent of the print mode.

Graphics mode. All data written to the printer, including escape sequences and control codes, are interpreted as dot-per-bit data. The data in *buf*r is interpreted as

binary data consisting of a 1 for every dot to be printed, and a 0 for every blank. Dots can be placed at any or all of the 924 positions per row (132 characters x 7 dots per character). *buf*r should contain 116 bytes or 58 words of data. The four least significant bits (0 - 3) of the 116th byte or 58th word, and any subsequent data, are ignored.

Compressed size. All characters are printed at 16.7 characters per inch. Up to 132 characters can be printed per line, at either 6 or 8 LPI.

DDC12 Status Request

The call sequence for a status request is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4] ] ] )
```

Control Word *cntwd*

Figure 2-11 shows the *cntwd* format for a status request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				Z				0				LU			

Figure 2-11. *cntwd* Format for Status Request

Z: The Z-bit determines how the optional parameters *stat3* and *stat4* are to be interpreted. Refer to the Status Parameter Definitions section of Chapter 1.

LU: Logical unit number of the printer.

stat1

Device status from DVT6 is returned to this required parameter. The format of DVT6 is shown in Figure 2-12.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		Device Type = 12					0		EOM	0			E		

Figure 2-12. DVT6 Status Word (Read Into *stat1*)

If EOM is set, the printer has run out of paper. The other bits are explained in the Status Parameter Definitions section of Chapter 1.

stat2

Interface card status from IFT6, the sixth word of the interface table. The bits of IFT6 are explained in Chapter 1, in the Status Parameter Definitions section.

stat3 and stat4

If the Z-bit is 0, then DVP1 and DVP2 are returned to *stat3* and *stat4*. If the Z-bit is 1, then *stat3* and *stat4* supply the address and length of a buffer to return the driver parameter area. *stat3* and *stat4* are shown in Figure 2-13.

if Z=0:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat3</i>	First Word of Driver Parameter Area (DVP1)															
<i>stat4</i>	Second Word of Driver Parameter Area (DVP2)															
if Z=1:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat3</i>	Buffer to Return the Driver Parameter Area															
<i>stat4</i>	Length of Buffer Identified by <i>stat3</i>															

Figure 2-13. stat3 and stat4 Formats for Z = 0 or 1

The driver parameters returned to *stat3* and *stat4* (or to the buffer that *stat3* identifies) are shown in Figure 2-14.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP1	HP-IB Address (Used by ID*37)															
DVP2	0															
DVP3	TRN	1										Print Mode				
DVP4	APE	Driver Defined Extended Status													SS	
DVP5	PRG								Character Set 2				Character Set 1			
DVP6	0										Line Density					
DVP7	0							Left Margin Definition								
DVP8	Device Timeout															

Figure 2-14. DDC12 Driver Parameters

DVP1: HP-IB address of the Printer (0-7)

DVP2: This driver parameter is always 1 for DDC12

DVP3: TRN = Transparency Mode: 0 = off
1 = on

Print Mode =	00B	Standard-size
	01B	Double-size
	02B	Graphics
	03B - 17B	Reserved

- DVP4: APE = Auto Page Eject: 0 = On
 1 = Off
- SS = Suppress Spacing: 0 = Normal spacing after printing
 1 = Suppress spacing (overprinting)
- DVP5: PRG = Character sets programmed: 1 = yes
 0 = no
- Char Set 2 = Secondary character set language code from Table 2-2.
 Char Set 1 = Primary character set language code from Table 2-2.
- DVP6: Line density: 6 = 6 LPI
 (decimal) 8 = 8 LPI
 0 = Line density not programmed
- DVP7: Left margin definition: 0 - 131 = width of left margin (decimal)
- DVP8: Device timeout in tens of milliseconds

DDC12 Extended Status Request

The extended status is read by making a call to RMPAR. A five-word array *xstat* is returned by the RMPAR call. *xstat*(1) through *xstat*(3) contain the device table entries DVT16 through DVT18. They are shown in Figure 2-15. *xstat*(4) and *xstat*(5) are undefined.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>xstat</i> (1)	x		0						Error Code							
<i>xstat</i> (2)	Transmission Log															
<i>xstat</i> (3)	PR	x						OL	RDY	x			STF	NP	PF	

Figure 2-15. *xstat*(1) to *xstat*(3) Status Words Format

xstat(1) = DVT16:

x = reserved bits

Decimal Error Codes:

- 1 IO-RQ Illegal Request
- 2 IO-NR Not ready
- 3 IO-TO Timeout
- 5 IO-TE Transmission error
- 12 IO-GE Generation error (also reported as IO-12)
- 21 IO-21 Printer failure
- 63 IO-63 Power failure with automatic restart

xstat(2) = DVT17: The value of the transmission log indicates the number of characters or words transferred in the last write, VFC definition or control request.

xstat(3) = DVT18:

- PR = A protocol error has occurred from which the driver has not yet recovered. This error should never occur under normal operating conditions.
- X = Undefined.
- OL = Printer is online.
- RDY = Printer is ready to receive data. This bit is 0 momentarily after most write requests, because the printer can receive only one line at a time. The user software can still initiate new requests while it is 0, because the driver waits until it is 1 before acting on the request. If the user software waits until it is 1 before initiating new requests, the printing process is slowed considerably.
- STF = Self-test failure.
- NP = Not printing. The printing action is being held up by one of the following:
 - Paper out
 - Platen gate open
 - Paper jam
 - Ribbon malfunction
- PF = A powerfail has occurred from which the driver has not yet recovered.

DDC12 Error Information

Any of the following errors may occur during a printing operation.

I/O device error on LU *xx*

The reason is: I/O request error

This illegal request error is reported if the driver detects any of the following:

1. A read request
2. A write request with *bufln* greater than 994 bytes (497 words)
3. A VFC definition request with *bufln* = 0

The printer is not brought down, and the request is flushed.

I/O device error on LU *xx*

The reason is: Device not ready

This error indicates that the printer is not ready to print because it is offline or because something is preventing it from printing; it may have run out of paper or the ribbon may be jammed. Add paper, replace the ribbon or press the ONLINE button on the front panel. The printer is brought down, and the request is not flushed.

I/O device error on LU *xx*

The reason is: Device timed out

A timeout error occurs when there is no response from the printer, indicating that it is disconnected or turned off. Turn it on or reconnect it. The printer is brought down, and the request is not flushed.

I/O device error on LU *xx*

The reason is: Transmission error

If the printer reports that a parity or protocol error has been detected on the HP-IB, the driver retries the transmission three times. After three failures, the driver exits with a transmission error. The printer is brought down, and the request is not flushed.

I/O device error on LU xx

The reason is: Device information specified at generation time is wrong

This error indicates that the driver extension area or parameter area was not sized large enough at system generation time. If this happens, the driver rejects all requests. If the defaults found in %DDC12 are used for generation, the IO-GE error will not occur. The printer is brought down, and the request is flushed.

I/O device error on LU xx

The reason is: Special driver defined error = 21

This catastrophic error should never occur under normal operating conditions. It indicates a hardware failure or recurrent power failure. The printer is brought down, and the request is not flushed.

HP-IB Magnetic Tape Driver DD*23

DD*23 drives the HP 7970E Magnetic Tape under HP-IB control.

Device driver DD*23 works with the HP-IB interface card driver, ID*37. DD*23 processes data buffers for the Magnetic Tape drive, while ID*37 performs the I/O instructions to the interface card.

DD*23 Read/Write Request

The call sequence for a read or write request is:

READ Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln*)

WRITE Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln*)

Control Word *cntwd*

Figure 2-16 shows the format of the control word (*cntwd*) used by DD*23 for an EXEC read or write request.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	0	NB	UE	0	0	TR	0	0	BI	LU						

Figure 2-16. DD*23 Control Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are defined in the Read/Write Request Parameter Definitions section of Chapter 1. Bit 6 (BI) indicates ASCII data if 0 or Binary data if 1. If set, bit 10 (TR) indicates that Transparency Mode is in effect. When the transparency mode is not in effect, the tape LU is set down when more than two forward motion requests are attempted at or beyond the EOT marker. When transparency is in effect, there is no restriction on the number of forward motion requests processed at or beyond EOT.

bufr and *bufln*

The *bufr* and *bufln* parameters describe the data buffer for ASCII data in EXEC requests, as defined in the Read/Write Request Parameter Definitions section of Chapter 1. If *bufln* is negative, it indicates the number of characters in the buffer; if it is positive, it indicates words. A write request with *bufln* = 0 completes immediately. On input, only as much data as will fit in the user buffer is transmitted into that buffer. If *bufln* = 0 on a binary read, the magnetic tape skips forward one record. If *bufln* = 0 on an ASCII read, the request completes immediately.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each nonbuffered EXEC write request. The format of word 6 is the same as that returned to *stat1* after an EXEC 13 status request, as described below.

The B-Register will contain the positive number of words or characters that were transmitted during the last nonbuffered write request.

DD*23 Control Request

The call sequence for the DD*23 control request is:

```
CALL EXEC(3, cntwd)
```

Figure 2-17 shows the format of the control word (*cntwd*) used by DD*23 for a magnetic tape control request.

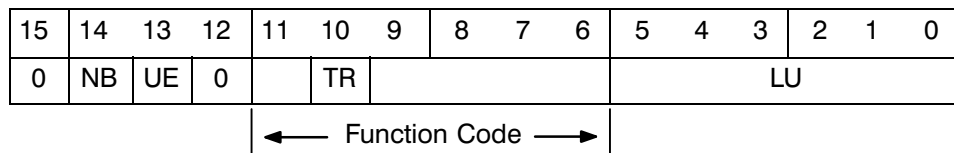


Figure 2-17. DD*23 Control Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are as defined in the Control Request Parameter Definitions section of Chapter 1. The following control functions are available through DD*23:

Code (TR off)	Code (TR on)	Function
00	20B	Rewind
01	21B	Write EOF
02	22B	Backspace one Record
03	23B	Forward Space one Record
04	24B	Rewind (same as 00)
05	25B	Rewind/Standby
06	26B	Dynamic Status
12B	32B	Write Gap
13B	33B	Forward Space one File
14B	34B	Backspace one File

DD*23 Status Request

The call sequence for the DD*23 status request is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4] ] ] )
```

Control Word *cntwd*

Figure 2-18 shows the control word format for a DD*23 status request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				Z		0				LU					

Figure 2-18. DD*23 Status Request *cntwd* Format

stat1 and *stat2* Status Parameters

The format of the returned status parameters, *stat1* and *stat2*, is shown in Figure 2-19.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type = 23					EOF	DB	EOM	BOM	SE	WP	OF	E	
<i>stat2</i>	AV		Interface Type = 37					0		I/O Select Code						

Figure 2-19. *stat1* and *stat2* Status Words Format

For *stat1*, the AV (device availability) bits and device type are as defined in the Status Request Parameter Definitions section of Chapter 1. The status codes returned in bits 0 through 7 are defined as:

EOF indicates tape is currently positioned at an end-of-file mark.

DB indicates device busy. Tape is performing a function that prevents other operations from starting (that is, tape rewind).

EOM indicates end-of-medium. Tape is currently positioned at the end-of-tape.

BOM indicates the tape is currently positioned at the load point (beginning of medium).

SE indicates a soft error (read/write error - recovered).

WP indicates that the tape is write protected (no write ring installed on tape).

OF indicates that the tape unit is offline.

E error indicator set by system if driver sets any error code in DVT 16.

stat2, the interface type, is 37B to correspond with the HP-IB interface card. All other bits are as defined in Chapter 1 under Status Request Parameter Definitions.

stat3 and *stat4* Status Parameters

If the Z-bit is zero, *stat3* returns the first word of the driver parameter area, and *stat4* returns the second word of the driver parameter area. If Z is 1, *stat3* is the buffer to return the driver parameter to, and *stat4* is the length of the *stat3* buffer. The only words defined in the driver parameter area are the HP-IB address and unit number of the tape unit.

DD*23 Extended Status

As discussed in the Extended Device Status section of Chapter 1, the extended status can be obtained by making a call to RMPAR after an unbuffered EXEC request to determine the status of the request. The format of the returned status is shown in Figure 2-20.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	X		0								Error Code					
DVT17	Transmission Log = 0															
DVT18	Extended Status #1															
DVT19	Extended Status #2															

Figure 2-20. DVT16 to DVT19 Extended Status Words

For DVT16, bits 14 and 15 are used by the system. The error codes returned to bits 0 through 5 are:

- 00 = Normal request completion
- 02 = Device not ready
- 03 = Device has timed out
- 04 = Device is at end-of-medium
- 05 = Transmission error/parity error (MTE)
- 06 = Tape Unit is Write Protected
- 77B = Driver request restart

DVT17 is the transmission Log (0 on an error), posted with the original length.

DVT18 and DVT19 are extended status words. The word formats are shown in Figure 2-21.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT18	EOF	BOT	EOT	STE	CR	WP	MTE	OL				OV	TR	RW	TB	IB
DVT19	RES		PWR	CPE	RES			RET								

Figure 2-21. *stat1* and *stat2* Status Words Format

EOF indicates the tape is at end-of-file mark.

BOT indicates that the tape is at load point (beginning of tape).

EOT indicates that the tape is at end-of-tape mark.

STE indicates a single track error (corrected by drive, soft error).
CR indicates command rejected. A command was sent but rejected by tape unit.
WR indicates the tape unit is write protected (write ring not installed).
MTE indicates a multiple track error (uncorrectable data error).
OL indicates the tape unit is online.
RES reserved by the system.
OV indicates a data error (overflow/underflow occurred on the bus).
TR indicates tape runaway (no data detected on 25 feet of tape).
RW indicates that the tape is rewinding.
TB indicates that the tape unit is busy (or sick).
IB indicates that the tape interface is busy.
PWR indicates that power has just been restored.
CPE indicates a command parity error.
RET indicates number of retries before error-free completion.

HP-IB Magnetic Tape Driver DD*24

Device driver DD*24, along with HP-IB interface driver ID*37, provides the software interface between RTE-A and tape drives HP 7974A, 7978A/B and 7980. The device driver can be accessed through standard EXEC read, write and control requests. The following subsections show the form of the standard EXEC calls.

DD*24 can also be accessed through class EXEC calls. The class EXEC calls are described in the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

DD*24 Read/Write Request

The call sequence for a read or write request is:

Read request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln*)
Write request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln*)

Control Word *cntwd*

Figure 2-22 shows the format of control word (*cntwd*) used by DD*24 for an EXEC read or write request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	NB	UE	x	SR	TR	x	x	x	BI	LU					

Figure 2-22. DD*24 Read/Write *cntwd* Format

In Figure 2-22, x indicates a bit not used by DD*24.

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

Bit 6 (BI) indicates binary data if set to 1, or ASCII data if set to 0.

Bit 10 (TR) indicates transparency mode if set to 1; however, DD*24 remains in transparency mode at all times, ignoring bit 10. DD*24 places no restrictions on operations beyond the EOT marker and as such does not look at bit 10.

Bit 11 (SR) indicates a streaming mode request when set. In streaming mode the driver makes use of the immediate response function of the controller.

bufr* and *bufln

The *bufr* parameter is the address of the user-defined read/write buffer.

The *bufln* parameter is the length of *bufr* (in words if positive, in characters if negative) as defined in Chapter 1. The HP 7974A and HP 7978A tape drives limit the record size to a maximum of 16K bytes.

A write request with a length of zero will not cause any data or record information to be written to the tape; in other words, a zero-length buffer requires no device action and completes immediately on the driver. On input, only as much data as will fit in the buffer is transmitted into that buffer. A zero-length binary read causes a forward space of one record, while a zero length ASCII read immediately completes the request; again, no device action required.

A- and B-Register Returns

For standard (EXEC 1 and 2) nonbuffered requests, the A-Register will contain word 6 of the device table (DVT6) after each request. The format of DVT6 is the same as that of *stat1* returned after an EXEC 13 request and is described in the subsection on Status Requests. For nonbuffered requests, the B-Register contains the positive number of words or characters (depending on program specification) actually transferred.

DD*24 Control Request

The call sequence for the DD*24 control request is:

```
CALL EXEC(3, cntwd[, pram1[, pram2[, pram3[, pram4]]]])
```

Control Word *cntwd* and Parameters

Figure 2-23 shows the format of control word *cntwd* used by DD*24 in an EXEC control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	NB	UE	Z	SR	TR	Function Code				LU					

Figure 2-23. DD*24 Control Request *cntwd* Format

In Figure 2-23, x indicates a bit not used by DD*24.

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are as defined in the Control Request Parameter Definitions section of Chapter 1.

The Z-bit, when set to 1, specifies that *pram3* and *pram4* are used as optional buffer and buffer-length parameters. The diagnostic requests (function code 17B, below) are the only requests that use *pram3* and *pram4* as buffer and buffer length.

Bit 11 (SR) indicates a streaming mode request when set. In streaming mode the driver makes use of the immediate response function of the controller.

Bit 10 (TR) indicates transparency mode if set to 1; however, DD*24 remains in transparency mode at all times, ignoring bit 10. DD*24 places no restrictions on operations beyond the EOT marker and as such does not look at bit 10.

Bits 6 through 9 contain the *cntwd* function code. *cntwd* codes 0 through 14 (octal) have the following functions:

Code (octal)	Function
00	Rewind.
01	Write file mark.
02	Backspace one record.
03	Forward space one record.
04	Rewind (same as 0).
05	Rewind and go offline.
06	Dynamic device status. A control request with a function code of 6 causes DD*24 to read the current drive status and update the status information in the DVT and driver parameter area. On return, the A-Register contains a copy of the DVT6 status. A dynamic device status request can be made after each streaming mode write request to catch errors that may have occurred.
12	Write gap (erases about 3.5 inches of tape).
13	Forward space one file.
14	Backspace one file.

Function code 15B selects density according to the value in *pram1* as follows:

pram1 = 6250 for 6250 bpi GCR (HP 7978A, 7978B, and 7980).

pram1 = 1600 for 1600 bpi PE.

pram1 = 800 for 800 bpi NRZI.

Function code 17B is a diagnostics request. The Z-bit must be set for these functions:

pram1 = 0 for device clear and read ID bytes into DP4.

pram1 = 1 for loopback test, with:

pram2 = unused,

pram3 = a 257-word buffer, with:
 words 1-128 = data to send (256 bytes)
 words 129-256 = data returned
 word 257 = DSJ returned

pram4 = buffer length (257 words)

pram1 = 2 for self-test, with:

pram2 = self-test number (1-255)

pram3 = two-word buffer for returned self-test status and DSJ; two bytes of self-test status are returned in the first word, and the DSJ value is returned in the high byte of the second word.

pram4 = buffer length (2 words)

pram1 = 3 for read status log, with:

pram2 = unused

pram3 = buffer for returned status log

pram4 = buffer length (2064 words)

A- and B-Register Returns

The characteristics of the A-Register contents on return from a control request are the same as those for the read and write requests.

The B-Register on return from a control request is undefined.

DD*24 Status Request

The Status Request is an EXEC call using code 13. The call has the following form:

```
CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]])
```

The status returned by the EXEC 13 Status Request reflects the information found in the DVT, IFT, and driver parameter area at the time the request is made.

Control Word *cntwd*

Figure 2-24 shows the format of control word (*cntwd*) in the Status Request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-24. DD*24 Status Request *cntwd* Format

The Z-bit indicates the format of the *stat3* and *stat4* parameters, as described below. LU is the logical unit number of the tape drive.

Parameters *stat1* and *stat2*

Figure 2-25 shows the format of parameters *stat1* and *stat2* of the EXEC 13 Status Request.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type = 24B				EOF	0	EOT	BOT	RE	WP	OF	E		
<i>stat2</i>	AV		Interface Type = 37B				0		I/O Select Code							

Figure 2-25. DD*24 Status Request Parameters *stat1* and *stat2*

stat1 is DVT6 for the associated LU, which is the same as the A-Register return for an EXEC read, write, or control request. *stat2* is IFT6 of the associated interface.

Referring to Figure 2-25, the AV and Device Type bits are as defined in the Status Request Parameter Definitions section of Chapter 1. The Interface Type is 37B, to correspond to the HP-IB interface card. The status codes returned in bits 0 through 7 are defined as follows:

- EOF = End of file, set by driver when a request has caused the drive to sense an EOF mark.
- EOT = End of tape, set by driver when drive has sensed that tape is positioned beyond the EOT marker.
- BOT = Beginning of tape, set by driver when the tape is positioned at the BOT marker.
- RE = Recovered error (soft error).
- WP = Write-protected tape.
- OF = Drive offline.
- E = Error indicator, set by system if there is an error code in DVT16.

Parameters *stat3* and *stat4*

The format of *stat3* and *stat4* is dependent on the value of the Z-bit in *cntwd*. If Z is zero, then *stat3* and *stat4* contain the first and second words of the driver parameter area, respectively. If Z is 1, then *stat3* is the address of a buffer where information from the driver parameter area is to be returned and *stat4* is the length of the buffer.

The driver parameter area for DD*24 is five words long and has the following format:

- DP1 = HP-IB address.
- DP2 = Default density:
6250 for 6250 bpi GCR (HP 7978A, 7978B, 7980)
1600 for 1600 bpi PE
800 for 800 bpi NRZI
- DP3 = Identify bytes returned from the tape drive:
564B for the HP 7974A
570B for the HP 7978A/B
600B for the HP 7980A
620B for the HP C1511B DDS
- DP4 = Additional device status (see below).
- DP5 = Additional device status (see below).

Figure 2-26 shows the format of the additional device status contained in DP4 and DP5.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP4	DO	0	IR	CR Class			Reserved (0)									
DP5	Controller Error Code								Queued Request Count							

Figure 2-26. DD*24 Driver Parameter Words DP4 and DP5

The fields in DP4 and DP5 are defined as follows:

DO indicates that the tape-path door is open.

IR indicates immediate response mode.

CR Class is the command reject error class. This status corresponds to bits 5-7 of status register 4 in the controller.

Controller Error Code is the specific error encountered. This error code corresponds to status register 5 in the controller.

Queued Request Count is the number of commands pending at the time of the error. Used only during immediate response mode. This count corresponds to status register 6 in the controller.

DD*24 Extended Status

The status information returned in the A-Register after a standard nonbuffered EXEC 1, 2 or 3 request is a subset of the status information found in the Extended Status (described below), and in most cases is all the status information a program will need.

Extended status can be obtained by making a call to RMPAR after a standard nonbuffered EXEC request. The extended status will contain DVT16 through DVT19 in that order, as represented in Figure 2-27.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DVT16	D	F	0						Error Code								
DVT17	Transmission Log = 0																
DVT18	EOF	BOT	EOT	RE	CR	WP	HE	ON	GCR	UD	EP	ET	TR	0	0	0	
DVT19	PE	NRZ	PWR	TE	POS	FE	SE	CE	Retry Count								

Figure 2-27. Extended Status Words DVT16 through DVT19

In word DVT16, D is the down bit, set by the driver to not down the device for the error condition. F is the flush bit, set by the driver to flush the request associated with the error code. The Error Code is the request error code. The possible error codes and the setting of the D- and F-bits are listed in Table 2-3.

Table 2-3. D- and F-Bit Error Codes

Code	D	F	Description
0	–	–	Normal request completion.
1	1	1	Illegal request. The illegal request may be due to the request length being greater than 16384 bytes, an undefined function for a control request, or unknown parameter values for a control request.
2	0	0	Device not ready. The tape drive is offline and cannot be accessed until it is placed back online.
3	0	0	Device has timed out. An operation did not complete in the allotted time. The tape drive may be powered down, disconnected from the HP-IB, or at the wrong HP-IB address.
5	0	0	Transmission error or parity error. The error may be due to HP-IB parity errors, tape parity errors, or drive hardware errors. If parity errors are occurring, replace the tape with a new one and, if the errors persist, run diagnostics to check the hardware.
6	0	0	Tape write-protected. A write operation was attempted on a write-protected tape.
10	0	0	Drive fault. An unrecoverable device error detected.
12	0	1	Generation error. The generation did not allocate enough table area for the DVT extension and driver parameter area.
21	0	1	Driver internal error. The driver detected an internal error.
22	1	1	Density error. May be due to an attempted reading of a tape that is blank or has a density unrecognizable to that drive or may be due to an attempted write of a density not available on the drive. Default density DP2 may be incorrectly set.

Word DVT17, the transmission log, is the number of words or characters transferred in a nonbuffered read, write, or control request.

The status bits in words DVT18 and DVT19 are defined as follows.

EOF = End of file.
BOT = Beginning of tape.
EOT = End of tape.
RE = Recovered error (soft error).
CR = Device command reject.
WP = Tape is write protected.
HE = Hard error. Uncorrectable tape error.
ON = Drive online (opposite to DVT6 offline status).
GCR = 6250 bpi GCR density selected.
UD = Unknown density on currently loaded tape.
EP = Internal data parity error.
ET = Data timing error. Data overrun or underrun occurred.
TR = Tape runaway.
PE = 1600 bpi PE density selected.
NRZ = 800 bpi NRZI density selected.
PWR = Power has just been restored to device.
TE = HP-IB command parity error.
POS = Tape position unrecoverable.
FE = Formatter error.
SE = Servo error.
CE = Controller error.

SCSI Tape Device Driver DDQ24

The following sections describe tape device driver DDQ24, including read and write request calls, Z-Buffer calls, and control request calls.

DDQ24 Driver Read and Write Calls

The calling sequences for the DDQ24 read and write calls are as follows:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln*)

Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln*)

The following paragraphs define the read and write call parameters.

The *cntwd* parameter, which is declared as a one-word integer, has the following format:

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	0			SD	BI	LU Number						

BB: Bypass bit. If set (= 1), this bit allows you to bypass the tape device driver and call the interface driver directly. We strongly recommend that you *do not do this* unless you are familiar with the consequences of this function.

NB: No buffer bit. If set, there is no buffer bit to inhibit output buffering. This has no effect on the driver.

UE: User error bit. If this bit is set, the user interprets errors and takes action to correct them.

Z: Z-Buffer bit, which is a dual buffer bit. If set, this bit indicates that the call is a Z-Buffer read or write, instead of an ordinary read or write. It also adds two buffers, one for the Z-read/write command and one for the Z-read/write command length. See the section “Z-Buffer Calls”.

If this bit is 0, it indicates an ordinary read or write.

SD: Read sense data bit. To read the last SCSI command and sense data after a “check condition” status has been returned by the SCSI device, use the read request with this bit set and the request length equal to 11 words.

BI: Binary data bit. If set, this bit indicates a binary data read or write operation. If not set, it indicates an ASCII data read or write operation.

LU Number: Normally, the LU number can be any number from 0 (zero) to 63, inclusive. However, if you use an XLUEX call instead of a normal EXEC call, you can use LU designations from 0 to 255, inclusive. For information about XLUEX calls, refer to the *RTE-A Programmer’s Reference Manual*, part number 92077-90007.

The *bufr* parameter indicates the user buffer address to read or write. This parameter is valid only for read and write calls (including Z-Buffer calls), not for control calls. The buffer address can be any legal value, as enforced by the operating system, just like any other EXEC call buffer.

The *bufln* parameter is the request length that specifies the number of words or bytes to be transferred. A positive value indicates that the number is the number of words. A negative value indicates that the number equals the number of bytes, except for the special case of -32768 (100000B) which indicates 32,768 words. This parameter is valid only for read and write calls (including Z-Buffer calls), not for control calls.

DDQ24 Z-Buffer Calls

Z-Buffer read and write calls enable you to construct bus transactions that are outside the range of the usual read and write transfers, such as tape eject requests.

When the Z-bit (bit 12) is set, the form of the EXEC call changes to:

Z-Read Request: CALL EXEC(1 , cntwd , bufr , bufln , command , commandln)
Z-Write Request: CALL EXEC(2 , cntwd , bufr , bufln , command , commandln)

The difference between the usual read/write calls and Z-read/Z-write calls is that the Z-Buffer calls include two extra parameters to specify the command and the command's length. All Z-read/Z-write driver calling sequences must include these two parameters.

The *command* parameter specifies the Command Descriptor Block (CDB), and is used only in Z-read and Z-write calls. This parameter contains the SCSI command you want to send to the device. The device driver and interface driver do not modify the command buffer. It is your responsibility to ensure that the format of the command request is correct. If the command request format is not correct, an error message appears.

A Z-write call sends the specified command along with the data in the user buffer. A Z-read sends the specified command, and reads the response from the device.

The *commandln* parameter specifies the length of the Command Descriptor Block sent with Z-Buffer calls in bytes or words. This parameter is used only in Z-read and Z-write calls. The parameter's value equals the number of words in the command.

Read SCSI Command and Sense Data Request After Check Condition

To read the last SCSI command and sense data after a "check condition" status has been returned by the SCSI device, use the following call:

```
CALL EXEC( 1 , 200B+lu , bufr , 11 )
```

Upon return, the first byte of *bufr* indicates the last SCSI command, and words 2 through 11 of *bufr* contain the sense data.

Control Calls

You specify the control calls described in this section as the second parameter of a control request. The subfunction bits in the control word (*cntwd*) identify the control commands. The calling sequence for the DDQ24 control call is as follows:

```
Control Request    CALL EXEC( 3 , cntwd[ , parm1[ , parm2] ] )
```

The parameters *parm1* and *parm2* are optional parameters. These parameters contain data specific to a particular control call. The parameters you can specify with each control call are described with the control calls. Some control calls do not have any parameters.

The *cntwd* parameter, which is declared as a one-word integer, has the following format:

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	0	UE	0	SubFunction						LU Number					

BB: Bypass bit. If set (= 1), this bit allows you to bypass the tape device driver and call the interface driver directly. We strongly recommend that you *do not do this* unless you are familiar with the consequences of this function.

UE: User error bit. If this bit is set, the user interprets errors and takes action to correct them.

Subfunction: These bits indicate the control command subfunction. DDQ24 supports the following control requests:

<u>Octal Value</u>	<u>Definition</u>
0B	Rewind
1B	Write <i>n</i> File Mark(s)
2B	Backward space <i>n</i> record(s)
3B	Forward space <i>n</i> record(s)
4B	Rewind
5B	Rewind and go offline (unload tape)
6B	Dynamic Status Request
7B	Write <i>n</i> Set Mark(s)
10B	Backward space <i>n</i> Set Mark(s)
11B	Forward space <i>n</i> Set Mark(s)
13B	Forward space <i>n</i> file(s)
14B	Backward space <i>n</i> file(s)
15B	Set tape density, enable/disable compression
16B	Enable/disable driver request sense after check condition

where *n* is a positive integer from 1 to 32767, inclusive.

The following paragraphs describe each control command in detail.

LU Number: Normally, the LU number can be any number from 0 (zero) to 63, inclusive. However, if you use an XLUEX call instead of a normal EXEC call, you can use LU designations from 0 to 255, inclusive. For information about XLUEX calls, refer to the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

Control 0B Rewind

Control 4B Rewind

These two commands are exactly the same. Both of these rewind commands rewind the tape drive, but leave the tape online and loaded. There are no parameters to use with these commands.

The calls for control commands 0B and 4B, respectively, are as follows:

```
CALL EXEC( 3, 000B+lu )    (No parameters)
CALL EXEC( 3, 400B+lu )    (No parameters)
```

For example, if you want to rewind the tape on a device that has LU number 5, using the 4B Rewind command, execute the following call:

```
CALL EXEC( 3, 405B )
```

The 3 denotes that the call is a control command, and the 405B is 400B (the octal control command designation) plus 5B (the LU number).

Control 1B Write File Mark

This command writes one or more file marks to the tape at the current tape location. It has one optional parameter which indicates the number of file marks to write at the particular location.

The call for control command 1B is

```
CALL EXEC( 3, 100B+lu , [ number_filemarks ] )
```

where the optional parameter *number_filemarks* is a positive integer that indicates the number of file marks to write at the current tape location. If you do not use this parameter, the default number of file marks is 1. If you enter zero in this field, the driver converts it to a 1.

For example, if you want to write three file marks to identify the current location on a tape on LU 20B, execute the following call:

```
CALL EXEC( 3, 120B, 3 )
```

The first 3 denotes that the call is a control command, the 120B is 100B (the octal control command designation) plus 20B (the LU number), and the second 3 specifies three file marks.

You may wish to write more than one file mark at a location in order to discriminate between different marked locations on the tape.

Note that this command sets the end of file bit (bit 7 in DVT 6) to write the file mark. For more information on this, see the section titled "Status" in this chapter.

Control 2B Backward Space Record

This command rewinds the tape one or more records from the current tape location. It has one optional parameter to specify the number of records to rewind.

The call for control command 2B is:

```
CALL EXEC(3,200B+lu,[num_records])
```

where the optional parameter *num_records* is a positive integer that indicates the number of records to rewind. If you do not use this parameter, the default number of records to rewind is 1. If you enter zero in this field, the driver converts it to a 1.

For example, if you want to rewind six records from the current location on a tape on LU 7B, execute the following call:

```
CALL EXEC(3,207B,6)
```

where 3 denotes that the call is a control command, 207B is 200B (the octal control command designation) plus 7B (the LU number), and 6 is the number of records to rewind.

Control 3B Forward Space Record

This command forwards the tape one or more records from the current tape location. It has one optional parameter to specify the number of records to forward.

The call for control command 3B is:

```
CALL EXEC(3,300B+lu,[num_records])
```

where the optional parameter *num_records* is a positive integer that indicates the number of records to forward. If you do not use this parameter, the default number of records to forward is 1. If you enter zero in this field, the driver converts it to a 1.

For example, if you want to go forward nine records from the current location on a tape on LU 3B, execute the following call:

```
CALL EXEC(3,303B,9)
```

where 3 denotes that the call is a control command, 303B is 300B (the octal control command designation) plus 3B (the LU number), and 9 is the number of records to forward.

Control 5B Rewind and Unload Tape

This command rewinds the tape drive and unloads the tape. There are no parameters to use with this command.

The call for control command 5B is as follows:

```
CALL EXEC(3,500B+lu) (No parameters)
```

For example, if you want to rewind and unload the tape on a device that has LU number 6, execute the following call:

```
CALL EXEC(3,506B)
```

The 3 denotes that the call is a control command, and the 506B is 500B (the octal control command designation) plus 6B (the LU number).

Control 6B Dynamic Status

This command returns the status of the specified LU, extended transaction status, and error codes. Status is returned in DVT 6 and DVT 16 through DVT 19 (refer to the section titled “Status” in this chapter for more information).

The calls for control command 6B are as follows:

```
CALL EXEC( 3 , 600B+lu )
CALL ABREG( a , b )
CALL RMPAR( PARMS )
```

The first call is the command call which specifies the command and LU. The second call is to get the A- and B-Registers, and the third call is to get the RAM parameters.

For detailed information about the A and B Registers and the RMPAR command and its parameters, refer to the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

Control 7B Write Set Mark

This command writes one or more set marks to the tape at the current tape location. It has one optional parameter, which indicates the number of set marks to write at the particular location.

The call for control command 7B is

```
CALL EXEC( 3 , 700B+lu , [ number_setmarks ] )
```

where the optional parameter *number_setmarks* is a positive integer that indicates the number of set marks to write at the current tape location. If you do not use this parameter, the default number of set marks is 1. If you enter zero in this field, the driver converts it to a 1.

For example, if you want to write two set marks to identify the current location on a tape on LU 5B, execute the following call:

```
CALL EXEC( 3 , 705B , 2 )
```

The first 3 denotes that the call is a control command, the 705B is 700B (the octal control command designation) plus 5B (the LU number), and the 2 specifies three set marks.

You may wish to write more than one set mark at a location in order to discriminate between different marked locations on the tape.

Note that this command sets the end of file bit (bit 7 in DVT 6) to write the set mark. For more information, see the section titled “Status” in this chapter.

Control 10B Backward Space Set Mark

This command rewinds the tape one or more set marks from the current tape location. It has one optional parameter to specify the number of set marks to rewind.

The call for control command 10B is

```
CALL EXEC(3,1000B+lu,[num_setmarks])
```

where the optional parameter *num_setmarks* is a positive integer that indicates the number of set marks to rewind. If you do not use this parameter, the default number of set marks to rewind is 1. If you enter zero in this field, the driver converts it to a 1.

For example, if you want to rewind two set marks from the current location on a tape on LU 3B, execute the following call:

```
CALL EXEC(3,1003B,2)
```

where 3 denotes that the call is a control command, 1003B is 1000B (the octal control command designation) plus 3B (the LU number), and 2 is the number of set marks to rewind.

Control 11B Forward Space Set Marks

This command forwards the tape one or more set marks from the current tape location. It has one optional parameter to specify the number of set marks to forward.

The call for control command 11B is

```
CALL EXEC(3,1100B+lu,[num_setmarks])
```

where the optional parameter *num_setmarks* is a positive integer that indicates the number of set marks to forward. If you do not use this parameter, the default number of set marks to forward is 1. If you enter zero in this field, the driver converts it to a 1.

For example, if you want to go forward four set marks from the current location on a tape on LU 6B, execute the following call:

```
CALL EXEC(3,1106B,4)
```

where 3 denotes that the call is a control command, 1106B is 1100B (the octal control command designation) plus 6B (the LU number), and 4 is the number of set marks to forward.

Control 13B Forward Space File

This command forwards the tape one or more files from the current tape location. It has one optional parameter to specify the number of files to forward.

The call for control command 13B is

```
CALL EXEC(3,1300B+lu,[num_files])
```

where the optional parameter *num_files* is a positive integer that indicates the number of files to forward. If you do not use this parameter, the default number of files to forward is 1. If you enter zero in this field, the driver converts it to a 1.

As an example, if you want to go forward eight files from the current location on a tape on LU 9B, execute the following call:

```
CALL EXEC(3,1309B,8)
```

where 3 denotes that the call is a control command, 1309B is 1300B (the octal control command designation) plus 9B (the LU number), and 8 is the number of files to forward.

Control 14B Backward Space File

This command rewinds the tape one or more files from the current tape location. It has one optional parameter to specify the number of files to rewind.

The call for control command 14B is

```
CALL EXEC(3,1400B+lu,[num_files])
```

where the optional parameter *num_files* is a positive integer that indicates the number of files to rewind. If you do not use this parameter, the default number of files to rewind is 1. If you enter zero in this field, the driver converts it to a 1.

As an example, if you want to rewind six files from the current location on a tape on LU 7B, execute the following call:

```
CALL EXEC(3,1407B,6)
```

where 3 denotes that the call is a control command, 1407B is 1400B (the octal control command designation) plus 7B (the LU number), and 6 is the number of files to rewind.

Control 15B Set Tape Density, Enable/Disable Compression

This command sets the selected density for the HP 7980S Magnetic Tape Drive in driver parameter 2 (DVP2). It has two parameters to specify tape density and enable or disable compression. Data compression can also be enabled/disabled for DDS (DAT) drives. For DDS (DAT) drives, the *density* parameter is ignored but a value must still be supplied.

The call for control command 15B is

```
CALL EXEC(3,1500B+lu,density,compression)
```

where the parameter *density* is either 800, 1600, or 6250. The parameter *compression* is either 0 (use the device default), 1 (enable compression), or -1 (disable compression).

As an example, if you want to set the tape density to 6250 bpi and enable compression, execute the following call:

```
CALL EXEC(3,1507B,6250,1)
```

where 3 denotes that the call is a control command, 1507B is 1500B (the octal control command designation) plus 7B (the LU number), 6250 is the tape density, and 1 enables compression.

Control 16B Enable/Disable Driver Request Sense After Check Condition

This command enables/disables the driver to issue the “request sense” command if a “check condition” occurs for subsequent read, write, and control requests. This only has effect if the UE bit is set in subsequent read, write, or control requests. If the UE bit is not set, then the driver will send the “request sense” command before returning to the user process.

The call for control command 16B is

```
CALL EXEC(3,1600B+lu[,parm1])
```

If the optional parameter *parm1* is 1, it prevents (disables) the driver from issuing the “request sense” command.

Driver Parameter Table Use in DDQ24

Driver Parameter Table usage in DDQ24																	
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP 01	D	Reserved, set to 0													SCSI Addr		
DVP 02	Tape Density																

D: Disconnect. When this bit is set, the driver does not disconnect until the transaction is complete.

This bit should be set for devices that do not retry after a reselection timeout. Setting this bit can impact performance, but may be required for some devices.

Tape Density: 0 (default, no density);
800;
1600; or
6250.

Status

DVT (Device Table) word 6 and DVT words 16 through 19 contain various types of status:

- DVT word 6 returns the LU status.
- DVT word 16 returns DDQ24 driver errors.
- DVT word 17 returns the length of the transferred data (the transmission log).
- DVT word 18 returns SCSI status.
- DVT word 19 returns both SCSI sense key and additional sense code.

For more detailed information about status and status calls, refer to the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

Logical Unit Status (DVT Word 6)

DVT word 6 contains the LU's status. LU status is returned in the A-Register after any request call, or when a Status request asks for it. The DVT word 6 bits have the following functions:

DVT word 6 Bits:																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
AV		Device type = 24B						EOF	DB	EOT	BOT	R	WP	OF	E	

- AV:** These two bits determine the availability of the tape device as follows:
- 0 = Driver is free to process a new request
 - 1 = Driver or user has set down the device
 - 2 = Driver currently processing I/O request
 - 3 = Device is down, but busy with a request (this normally occurs only when you down an active device).
- Device Type:** These bits set the device type. During system generation, they are set to 24B.
- EOF:** End-of-file bit. The driver sets this bit when a request causes it to sense an EOF mark.
- DB:** Device busy bit. If set, it indicates that the device is performing a function that prevents other operations from beginning (for example, a tape rewind).
- EOT:** End-of-tape bit. The driver sets this bit when the drive senses that the tape is positioned beyond the EOT mark.
- BOT:** Beginning-of-tape bit. The driver sets this bit when the tape is positioned at the BOT mark.
- R:** Recover bit. If set, the driver recovered from a soft error.
- WP:** Write protect bit. If set, this bit indicates that the tape is write-protected.
- OF:** Offline bit. If set, this bit indicates that the drive is offline.
- E:** Error bit. The system sets this bit if there is an error code in DVT word 16.

Driver Error (DVT Word 16)

DVT word 16, bits 5 through 0 contain error codes for DDQ24. The error codes are as follows:

<u>Dec.</u>	<u>Octal</u>	<u>Error and Solution</u>
0	0B	= Normal request completion.
1	1B	= Illegal request. Check the syntax of your EXEC request call. The EXEC call could have illegal parameters or specify an unused LU or an unimplemented command. Retry the request after you check it.
3	3B	= Device timeout. Use the TO command to increase the timeout period.
6	6B	= Device write protected. Remove the tape's write protection.
7	7B	= Address error. Ensure that the SCSI address is correct.

10	12B	=	Tape fault. Check the tape for any problems.
12	14B	=	Insufficient driver table space generated. Generate more driver table space.
22	26B	=	Incompatible cartridge. Use the correct type of tape cartridge.
23	27B	=	Positioning error detected. First try ejecting, then reloading the tape. If this does not solve the error, a hardware error is indicated.
24	30B	=	Hardware error. Check all connections, and ensure that tape drive is functioning correctly.
25	31B	=	Unknown error. This message should not appear. If it does, decode the error by checking DVT 19 which contains specific SCSI error codes.
26	32B	=	End of Data.
62	76B	=	Device busy.
63	77B	=	Driver retry request. This message can appear only if the Control Word user error bit (UE bit 13) is set. Control Word bit 13 enables the user to interpret errors and take action on them instead of allowing the system to handle errors. If you receive this message, check your request's syntax and try the request again.

All codes of 28 decimal (34B octal) or higher are interface driver error codes, except for 63 (77B), which indicates a driver retry request.

Transmission Log (DVT Word 17)

DVT word 17 contains the transmission log which is the length of the data transferred.

SCSI Status and Transaction Status (DVT Word 18)

DVT word 18 contains the SCSI status and transaction status of device driver DDQ24. The SCSI status returned in DVT18 (bits 7 – 0) is as follows:

xx00000xB = No error.
 xx00001xB = Check condition.
 xx00010xB = Condition met.
 xx00100xB = Busy.
 xx01000xB = Intermediate.
 xx01010xB = Intermediate condition met.
 xx01100xB = Reservation conflict.
 xx10001xB = Command terminated.
 xx10100xB = Queue full.

The transaction status returned in DVT18 (bits 15 – 8) is as follows:

0 = No error.
 1 = SCSI error. Check the sense code and additional sense code to identify the specific error.
 2 = Selection/reselection timeout. Check to ensure that the device address is correct.
 3 = Data error. This indicates a hardware error.
 4 = Bus parity error. This indicates a hardware error.
 5 = Reset.
 6 = Illegal bus free. This indicates a hardware error.
 7 = Abort request. The driver has sent an abort request.
 8 = Illegal request. This indicates an illegal script type.
 9 = Firmware error.

SCSI Sense Key and Additional Sense Code (DVT Word 19)

DVT word 19 contains the sense key and additional sense code. The sense code is returned in the first byte (bits 0 through 7) and the additional sense code is returned in the second byte (bits 8 through 15).

DDQ24 Driver Communication Word (DVT Word 20)

DVT word 20 has the following bit definitions:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I	x	S	x	x	1	R	R	D	P	DVT priority					

- I:** Initial flag. This bit is set by the generator, then cleared on the first entry to the driver.
- x:** Don't care.
- S:** Serial bit. This bit is set to indicate that this is a serial device, not a block device.
- R:** Retry bit. This is the retry counter. It counts up to three retries if the device is busy.
- D:** Delay bit. This bit is set if the driver is delaying and the next timeout is expected.
- P:** Powerfail retry bit. During a powerfail, a protocol failure may occur. This bit is set during the second attempt at the I/O request.

Disk Device Driver DD*30

DD*30 is the disk device driver designed to drive the following disk devices:

HP 9121	HP 9134A/B/XV
HP 9133A/B/XV	HP 9895

DD*30 provides the software interface to handle any device concerns, while the HP-IB interface card driver ID*37 controls the actual I/O instructions to the interface card.

DD*30 Read/Write Request

The call sequence for the DD*30 read and write requests is:

Read Request:	CALL EXEC (1 , <i>cntwd</i> , <i>bufr</i> , <i>bufln</i> , <i>track</i> , <i>sector</i>)
Write Request:	CALL EXEC (2 , <i>cntwd</i> , <i>bufr</i> , <i>bufln</i> , <i>track</i> , <i>sector</i>)

Control Word *cntwd*

Figure 2-28 shows the format of the control word (*cntwd*) used for a read or write request to DD*30.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	UE	Z	77B						LU					

Figure 2-28. DD*36 Read/Write Request *cntwd* Format

The UE (User Error) and LU (Logical Unit identifier) bits are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

Bit 12, the Z-bit, must be set to zero because *track* and *sector* (256 bytes) are integer variables, not control buffer descriptions.

bufr and *bufln*

The I/O buffer used in the EXEC request is described by parameters *bufr* and *bufln*, which are defined in the Read/Write Request Parameter Definition section in Chapter 1 of this manual.

***track* and *sector* Parameters**

The *track* and *sector* parameters are integer decimals that describe the track and logical sector positions to which the data buffer is written, or from which data is read. *sector* must be even for all read or write requests. The legal ranges of values for *track* and *sector* are shown in Table 2-4.

Table 2-4. Track and Logical Sector Ranges of Values

	HP 9895	HP 9133/34 Winchester	HP 9121
Track	0-134	0-154	0-64
Sector	0-58	0-58	0-30

Every disk has a fixed number of physical sectors, or blocks. The number of physical sectors determines the density of information that can be packed on the disk. Each physical sector contains 256 bytes. Physical sectors are twice as long as the logical sectors (128 bytes) used in program calls.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each EXEC read/write request to a disk. The format of word 6 is the same as that of *stat1* returned after an EXEC 13 status request, as described in the following section.

The B-Register will contain the number of words (if *bufln* was positive in the request) or characters (if *bufln* was negative) that were transmitted during the last read/write request.

As an example, the following FORTRAN sequence will read from track 100, sector 32 of an HP 7906 disk with LU 12. The data will be read into the buffer called BUFR.

```
IMPLICIT INTEGER (A-Z)
DIMENSION BUFR (128) (BUFLN = 128)
TRACK = 100
SECTOR = 32
CNTWD = 12 + 7700B
CALL EXEC(1, CNTWD, BUFR, BUFLN, TRACK, SECTOR)
```

DD*30 Control Request

Unlike other peripheral I/O devices that are controlled through EXEC requests, there are no control requests available to disk devices via DD*30. A control request to the disk will result in an IO-RQ error and the offending program will abort.

DD*30 Status Request

The call sequence for the status request is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4] ] ] )
```

Control Word *cntwd*

Figure 2-29 shows the control word (*cntwd*) format for an EXEC status request for devices driven by DD*30. The returned status information, the status of the last accessed disk drive, is discussed in the following paragraphs.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-29. DD*30 Status Request *cntwd* Format

stat1 and *stat2* Status Parameters

The format of the returned status parameters *stat1* and *stat2* is shown in Figure 2-30.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type = 30-32				0	0	EM	SE	0	0	E			
<i>stat2</i>	AV		Interface Type = 37				0	I/O Select Code								

Figure 2-30. DD*30 Status Request *cntwd* Format

In *stat1*, the AV (device availability) and Device Type bits are defined in the Status Request Parameter Definitions section of Chapter 1. The status code returned in bits 0 through 7 is:

- EM:** (End-of-Medium): A request was made for more sectors than are available.
- SE:** (Soft Error): At least one, but no greater than three, retries were made before the request was completed successfully.
- E:** (Error exists): Set by the system if the driver reports any error conditions in word 16 of the device table (DVT16). See the Extended Status section for the description and format of DVT16.

In *stat2* the interface type (always 37B) corresponds to the HP-IB Interface Card. All other bits are defined in Chapter 1 under Status Parameter Definitions.

stat3 and *stat4* Status Parameters

If the Z-bit is zero, *stat3* returns the first word of the driver parameter area, and *stat4* returns the second word of the driver parameter area.

If the Z-bit is 1, the driver parameter area is returned in the *stat3* buffer, and *stat4* is the length of the *stat3* buffer.

Driver Parameter Area

Table 2-5 contains the definitions of the words in the driver parameter area for DD*30.

Table 2-5. DD*30 Driver Parameter Area

Parameter	Parameter Description
+1	HP-IB Address
+2	Unit Number
+3	Starting Head Number
+4	Starting Cylinder
+5	Number of Spares
+6	Number of Tracks
+7	Physical Sectors/Track
+8	Number of Surfaces

DD*30 Extended Status

As discussed in the Extended Device Status section of Chapter 1, the extended status can be obtained by making a call to RMPAR after an EXEC read/write request to determine the status of the request. The format of the returned status words is shown in Figure 2-31.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>xstat(1)</i> / DVT16	X		0						Error Code							
<i>xstat(2)</i> / DVT17	Transmission Log = 0															
<i>xstat(3)</i> / DVT18	Extended Status #1															
<i>xstat(4)</i> / DVT19	Extended Status #2															

Figure 2-31. DVT16 to DVT19 Extended Status Words

In DVT16 (*xstat(1)*), bits 14 and 15 are used by the system. The error codes returned to bits 0 through 5 are:

- 0 = Normal request completion.
- 1 = Illegal request (such as track or sector address error).
- * 2 = Device not ready.
- * 3 = Device has timed out.
- * 5 = Transmission error (see error section for more information).
- * 6 = Device write protected.
- *10B = Device fault (hardware error).
- *77B = Media failure. Extended status has the disk controller status (see Figure 2-46).

* Codes not seen by the caller unless the UE (User Error) bit 13 was set.

DVT17 (*xstat(2)*) is the transmission log, always zero.

DVT18 (*xstat(3)*) describes the error in DVT16. Normally the first extended status word (*xstat(1)*/DVT16) will be set to zero. However, if there is some type of device error, and the device is able to return its status, the format of the word will be returned into DVT18. See Figure 2-32 for the format of this word.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI	PT	DT	Status				Unit # of Disk								

Figure 2-32. DVT18 Format for DD*30

SPI: if set, indicates that a spare track is in active use (HP 7906 only).

PT: if set, indicates a protected track (HP 7906 only).

DT: indicates a defective track.

DVT19 (*xstat(4)*) is similar to DVT18/*xstat(3)*. As with the first extended status word, the second extended word will normally be zero. The format of the word, if a status is available on the device error, is returned in DVT19. Figure 2-33 shows the format of this word.

The Status information in bits 8 through 12 of DVT18 is:

Status	Meaning
0	Normal completion of request. This status is transmitted whenever a command has been fully executed without an error.
1	Illegal opcode. Last command contained an unrecognizable opcode, which usually indicates an error in transmission caused by noise on the bus. Retries will be made.
3	Illegal drive type. An unknown drive type has been set in the drive type field for the disk.
7	Cylinder compare error. The address from the disk does not match the cylinder address from the controller (unsuccessful seek). Retries will be made.
10B	Unrecognizable Data error. A disk read type operation was terminated because of a data error. Retries will be made.
11B	Head/Sector compare error. Could not find an ID field on the track and/or on the target I/O field. Error could be caused by a hardware or media difficulty. Retries will be made.
12B	I/O program error. Illegal HP-IB sequence received by device.
14B	End of cylinder.
16B	Overrun. Data rate of the controller exceeds that of the DMA channel. Retries will be made. Note that this error should not occur for the HP 7902 since all data transfers are internally buffered.
20B	Illegal access to a spare track (HP 7906 only). Disk access was to a spare track that was in active use.
21B	Illegal access to a track marked defective. The request will be terminated.
22B	Hardware error. An internal overrun has occurred for the HP 9895, indicating a hardware failure. Retries will be made.
23B	The controller has detected some error condition with the disk drive. The extended status in DVT19 should be examined to determine the reason for the error.
26B	Track requested is marked protected (HP 7906 only).
27B	The request was terminated because the indicated unit number was outside the range available (0 through 3) for the HP 9895 disk. No retries will be made.
37B	Driver requests attention. The controller will report this status whenever a disk drive requests attention (for example, seek completion, drive becomes ready or not ready and seek check).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I	Status Word 2						ATN	WP	0	FLT	FS	SE	DNR	0	

Figure 2-33. DVT19 Format for DD*30

I: indicates that one or all bits (0, 1, 2, or 4) are set.

Status Word 2: is the upper byte of the second status word returned by the device. The value is device dependent. Therefore, refer to the appropriate hardware user documentation for the status information.

ATN: indicates that the disk drive is requesting attention.

WP: indicates that the HP 9895 diskette is write protected.

FLT: indicates a fault. Some type of hardware failure has occurred; retries will be made.

FS: is set if this is the first time DVT19 has been requested since a diskette has been inserted into the drive (HP 9895 only). Bit is zero if no disk is in the drive.

SE: (Seek check). Set if the cylinder, head, or sector is out of bounds, or nonexistent track was requested. Retries will be made.

DNR: indicates drive not ready. For the HP 9895 the drive is not connected (HOT=0) or no disk in drive (HOT=1). For the HP 7906, the drive is not up to speed or the disk is not loaded.

DD*30 Error Information

For unsuccessful data transmissions, the system will return the following error messages:

I/O - RQ @ LUxx

An illegal I/O request at LU xx for one of the following reasons:

1. Specified a negative track or sector value.
2. Specified an odd valued sector number.
3. Specified a sector or track number greater than the maximum allowed value.
4. An attempt was made to write more sectors than are available on disk.

I/O - NR @ LUxx,D

LU xx is not ready (status = 23) due to one of the following reasons:

1. No disk.
2. Drive not connected.
3. Door open.

I/O - TE @ LUxx

Transmission error due to one of the following conditions:

1. Device timeout.
2. Transfer terminated prematurely with no error indication from the disk controller.

I/O - WP @ LUxx

The device is write protected.

I/O - FA @ LUxx

Indicates a drive fault.

***TRK xx LU xx STAT=ZZ**

A media-related error as indicated by status shown in DVT18. Occurs after a failure of three retries. The disk will not be downed and the request will be flushed.

Transmission errors and errors indicating possible media failures (that is, cylinder or head/sector compare errors, and so forth.) are treated as follows:

The driver will attempt 3 retries. If the retry attempts are unsuccessful, a message is issued and the transmission log is set to zero. Except for the *TRK error, the disk will be set down.

Disk Device Driver DDM30

DDM30 is the disk device driver designed to drive the following disk devices:

HP 7906M
HP 7920M
HP 7925M

DDM30 provides the software interface to handle any device concerns, while the HP-IB interface card driver ID*37 controls the actual I/O instructions to the interface card.

DDM30 Read/Write Request

The call sequences for the DDM30 read and write requests are:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)
Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)

Control Word *cntwd*

Figure 2-34 shows the format of the control word (*cntwd*) used for a read or write request to DDM30.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	UE	Z	77B						LU					

Figure 2-34. DDM30 Read/Write Request *cntwd* Format

The UE (User Error) and LU (Logical Unit identifier) bits are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

Bit 12, the Z-bit, must be set to zero because *track* and *sector* (256 bytes) are integer variables, not control buffer descriptions.

bufr and *bufln*

The I/O buffer used in the EXEC request is described by parameters *bufr* and *bufln*, which are defined in the Read/Write Request Parameter Definition section in Chapter 1.

***track* and *sector* Parameters**

The *track* and *sector* parameters are integer decimals that describe the track and logical sector positions to which the data buffer is written, or from which data is read. *sector* must be even for all read or write requests. The legal ranges of values for *track* and *sector* are shown in Table 2-6.

Table 2-6. Track and Logical Sector Ranges of Values

	HP 7906M	HP 7920M	HP 7925M
Track	0-410	0-822	0-822
Sector	0-94	0-94	0-126

Every disk has a fixed number of physical sectors, or blocks. The number of physical sectors determines the density of information that can be packed on the disk. Each physical sector contains 256 bytes. Physical sectors are twice as long as the logical sectors (128 bytes) used in program calls.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each EXEC read/write request to a disk. The format of word 6 is the same as that of *stat1* returned after an EXEC 13 status request, as described in the following section.

The B-Register will contain the number of words (if *bufln* was positive in the request) or characters (if *bufln* was negative) that were transmitted during the last read/write request.

As an example, the following FORTRAN sequence will read from track 100, sector 32 of an HP 7906 disk with LU 12. The data will be read into the buffer called BUFR.

```
IMPLICIT INTEGER (A-Z)
DIMENSION BUFR (128)
:
BUFLN = 128
TRACK = 100
SECTOR = 32
CNTWD = 12 + 7700B
CALL EXEC(1, CNTWD, BUFR, BUFLN, TRACK, SECTOR)
```

DDM30 Control Request

Unlike other peripheral I/O devices that are controlled through EXEC requests, there are no control requests available to disk devices via DDM30. A control request to the disk will result in an IO-RQ error and the offending program will abort.

DDM30 Status Request

The call sequence for the status request is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4] ] ] )
```

Control Word *cntwd*

Figure 2-35 shows the control word (*cntwd*) format for an EXEC status request for devices driven by DDM30. The returned status information, the status of the last accessed disk drive, is discussed in the following paragraphs.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-35. DDM30 Status Request *cntwd* Format

stat1 and *stat2* Status Parameters

The format of the returned status parameters *stat1* and *stat2* is shown in Figure 2-36.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type = 32,34,35				0	0	EM	SE	0	0	E			
<i>stat2</i>	AV		Interface Type = 37				0	I/O Select Code								

Figure 2-36. *stat1* and *stat2* Status Words Format

In *stat1*, the AV (device availability) and Device Type bits are defined in the Status Request Parameter Definitions section of Chapter 1. The status code returned in bits 0 through 7 is:

- EM:** (End-of-Medium) A request was made for more sectors than are available.
- SE:** (Soft Error) At least one, but no greater than three, retries were made before the request was completed successfully.
- E:** (Error exists) Set by the system if the driver reports any error conditions in word 16 of the device table (DVT16). See the Extended Status section for the description and format of DVT16.

In *stat2*, the interface type (always 37B) corresponds to the HP-IB Interface Card. All other bits are defined in Chapter 1 under Status Parameter Definitions.

stat3 and *stat4* Status Parameters

If the Z-bit is zero, *stat3* returns the first word of the driver parameter area, and *stat4* returns the second word of the driver parameter area.

If the Z-bit is 1, *stat3* is the buffer to which to return the driver parameter area, and *stat4* is the length of the *stat3* buffer.

Driver Parameter Area

Table 2-7 contains the definitions of the words in the driver parameter area for DDM30.

Table 2-7. DDM30 Driver Parameter Area

Parameter	Parameter Description
+1	HP-IB Address
+2	Unit Number
+3	Starting Head Number
+4	Starting Cylinder
+5	Number of Spares
+6	Number of Tracks
+7	Physical Sectors/Track
+8	Number of Surfaces

DDM30 Extended Status

As discussed in the Extended Device Status section of Chapter 1, the extended status can be obtained by making a call to RMPAR after an EXEC read/write request to determine the status of the request. The format of the returned status words is shown in Figure 2-37.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>xstat(1)</i> / DVT16	X		0						Error Code							
<i>xstat(2)</i> / DVT17	Transmission Log = 0															
<i>xstat(3)</i> / DVT18	Extended Status #1															
<i>xstat(4)</i> / DVT19	Extended Status #2															

Figure 2-37. DVT16 to DVT19 Extended Status Words

In DVT16 (*xstat(1)*), bits 14 and 15 are used by the system. The error codes returned to bits 0 through 5 are:

- 0 = Normal request completion.
- 1 = Illegal request (such as track or sector address error).
- *2 = Device not ready.
- *3 = Device has timed out.
- *5 = Transmission error (see error section for more information).
- *6 = Device write protected.
- *10B = Device fault (hardware error).
- *77B = Media failure. Extended status has the disk controller status (see Figure 2-38).

* Codes not seen by the caller unless the UE (User Error) bit 13 was set.

DVT17 (*xstat*(2)) is the transmission log, always zero.

DVT18 (*xstat*(3)) describes the error in DVT16. Normally the first extended status word (*xstat*(1)/DVT16) will be set to zero. However, if there is a device error, and the device is able to return its status, the format of the word is returned in DVT18. See Figure 2-38 for the format of this word.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI	PT	DT	Status				Unit # of Disk								

Figure 2-38. DVT18 Format for DDM30

DVT19 (*xstat*(4)) is similar to DVT18/*xstat*(3). As with the first extended status word, the second extended word is normally zero. The format of the word, if status is available on the device error, is returned in DVT19. Figure 2-39 shows the format of this word.

SPI, if set, indicates that a spare track is in active use (HP 7906 only).

PT, if set, indicates a protected track (HP 7906 only).

DT indicates a defective track.

Status information in bits 8 through 12 of DVT 18 is:

Status	Meaning
0	Normal completion of request. This status is transmitted whenever a command has been fully executed without an error.
1	Illegal opcode. Last command contained an unrecognizable opcode, which usually indicates an error in transmission caused by noise on the bus. Retries will be made.
3	Illegal drive type. An unknown drive type has been set in the drive type field for the disk.
7	Cylinder compare error. The address from the disk does not match the cylinder address from the controller (unsuccessful seek). Retries will be made.
10B	Unrecognizable Data error. A disk read type operation was terminated because of a data error. Retries will be made.
11B	Head/Sector compare error. Could not find an ID field on the track and/or on the target I/O field. Error could be caused by a hardware or media difficulty. Retries will be made.
12B	I/O program error. Illegal HP-IB sequence received by device.
14B	End of cylinder.
16B	Overrun. Data rate of the controller exceeds that of the DMA channel. Retries will be made.
17B	Possibly correctable error.
20B	Illegal access to a spare track (HP 7906 only). Disk access was to a spare track that was in active use.
21B	Illegal access to a track marked defective. The request will be terminated.

- 22B Access not ready during data operation. Head motion detected during data transfer. Will retry.
- 23B The controller has detected some error condition with the disk drive. The extended status in DVT19 should be examined to determine the reason for the error.
- 26B Track requested is marked protected (HP 7906 only).
- 27B The request was terminated because the indicated unit number was outside the range available (0 through 7). No retries will be made.
- 37B Driver requests attention. The controller will report this status whenever a disk drive requests attention (for example, seek completion, drive becomes ready or not ready, and seek check).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
I	Status Word #2						ATN	WP	0	FLT		SE	DNR	0		

Figure 2-39. DVT19 Format for DDM30

I indicates that one or all bits (0, 1, 2, or 4) are set.

Status Word #2 is the upper byte of the second status word returned by the device. The value is device dependent. Refer to the appropriate hardware user documentation for the status information.

ATN indicates that the disk drive is requesting attention.

WP indicates that the HP 7906 is write protected.

FLT indicates a fault. Some type of hardware failure has occurred; retries will be made.

SE (Seek check). Set if the cylinder, head, or sector is out of bounds, or non-existent track was requested. Retries will be made.

DNR indicates drive not ready. For the HP 7906, the drive is not up to speed or the disk is not loaded.

DDM30 Error Information

For unsuccessful data transmissions, the system will return the following error messages:

I/O - RQ @ LUxx

An illegal I/O request at LU xx for one of the following reasons:

1. Specified a negative track or sector value.
2. Specified an odd valued sector number.
3. Specified a sector or track number greater than the maximum allowed value.
4. An attempt was made to write more sectors than are available on the disk.

I/O - NR @ LUxx,D

LU xx is not ready (status = 23) due to one of the following reasons:

1. No disk.
2. Drive not connected.
3. Door open.

I/O - TE @ LUxx

Transmission error due to one of the following conditions:

1. Device timeout.
2. Transfer terminated prematurely with no error indication from the disk controller.

I/O - WP @ LUxx

The device is write protected.

I/O - FA @ LUxx

Indicates a drive fault.

***TRK xx, LU xx Status = xx octal (DDM30)**

A media-related error as indicated by status shown in DVT18. Occurs after a failure of ten retries. The disk will not be downed and the request will be flushed.

TRK xx, LU xx*Error Correction invoked (DDM30)****xxx Corrections since last boot**

Indicates the error correction algorithm had to be used to correct marginal data.

Transmission errors and errors indicating possible media failures (that is, cylinder or head/sector compare errors, and so forth.) are treated as follows:

The driver will attempt 10 retries. If the retry attempts are unsuccessful, a message is issued and the transmission log is set to zero. Except for the *TRK error, the disk will be set down.

SCSI Disk Device Driver DDQ30

The following sections describe disk device driver DDQ30, including Read/Write request calls and Z-Buffer calls.

DDQ30 Read and Write Calls

The calling sequences for the DDQ30 read and write requests are as follows:

Read request: CALL EXEC(1 , cntwd , bufr , bufln , track , sector)
Write request: CALL EXEC(2 , cntwd , bufr , bufln , track , sector)

The following paragraphs define the read and write call parameters.

The second parameter is the *cntwd* parameter, which is declared as a one-word integer. You use the subfunction bits (11 through 6) of the *cntwd* parameter differently for read and write requests than you do for control requests. All driver calling sequences must include this parameter. The control word bits have the following functions:

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	NT	0					LU Number					

- BB:** Bypass bit. This bit allows the interface driver to be called directly. We strongly recommend that you *do not do this* unless you are familiar with the consequences of this function. The DDQ30 driver translates the track and sector addresses into block addresses before sending those addresses to the interface driver IDQ35. This is necessary because the SCSI standard uses block addresses, not track and sector addresses. In addition, DDQ30 adds an address offset that corresponds with the logical disk address offset. If you bypass DDQ30 and communicate directly with the interface driver, the track and sector-to-block address translation does not occur. Also, the address offset is not added. In other words, you would be talking to the disk in “raw” (bit-by-bit) mode. Again, we recommend that you do not bypass DDQ30 to talk directly to the interface driver.
- NB:** No buffer bit. This bit inhibits output buffering. No effect on driver.
- UE:** User error bit. If this bit is set, the user interprets errors and takes action to correct them.
- Z:** Dual buffer bit. 0 = normal read or write. For Z bit = 1, see the section titled “Z-Buffer Calls” in this chapter.
- NT:** No timeout bit. If this bit is set, issue the read or write request without timeout.
- LU Number:** Used by RTE-A to translate to a Device Table (DVT); not passed to the driver.

bufr: Buffer address, any legal value as enforced by the operating system.

bufln: The request length specifying the number of words or bytes to be transferred. Positive values indicate the number of words and negative values indicate the number of bytes, except for the special case of -32768 (100000B), which indicates 32,768 words. HP recommends that you use values that are a multiple of 128 words (256 bytes) because the disk driver always writes a complete block of 128 words. If you request a lesser value, the rest of the sector is written with whatever data that happens to follow the requested buffer.

track: The track address where the data is stored. The range of values is 0 to 65534; limited by the size of the disk for which the call is intended.

sector: The sector address where the data is stored. For backward compatibility, sectors are defined as 64 words each. There are two sectors in an RTE block. The driver rejects calls that start on an odd sector boundary because every request must begin on an RTE block boundary.

The length of the request plus the starting address as given by the track and sector values must remain within the defined size of the disk.

Z-Buffer Calls

The Z-Buffer is used to construct bus transactions that are outside the range of the usual read and write transfers, such as disk formatting requests.

When the Z-bit (bit 12) is set, the form of the EXEC call changes to:

Read request: CALL EXEC(1 , *cntwd* , *bufr* , *bufln* , *zbuf* , *zlen*)

Write request: CALL EXEC(2 , *cntwd* , *bufr* , *bufln* , *zbuf* , *zlen*)

The *zbuf* and *zlen* parameters describe a user Z-Buffer. The driver places no restrictions on the content of the Z-Buffer or the data buffer. The concept of track and sector does not apply to these calls.

Control Calls

The calling sequence for the DDQ30 control request is as follows:

Control request: CALL EXEC(3 , *cntwd* , *p1* , *p2* , *p3* , *p4*)

The *cntwd* (control word) bits are as follows:

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	0	UE	0	Function Code						LU Number					

Control 16B Enable/Disable Driver Request Sense After Check Condition

This command enables/disables the driver to issue the “request sense” command if a “check condition” occurs for subsequent read, write, and control requests. This only has effect if the UE bit is set in subsequent read, write, or control requests. If the UE bit is not set, then the driver will send the “request sense” command before returning to the user process.

The call for control command 16B is:

```
CALL EXEC(3,1600B+lu[,parms])
```

If the optional parameter *parms* is 1, it prevents (disables) the driver from issuing the “request sense” command.

Control 76B Alter Driver Parameter Table (RTE Block Information)

This command alters the starting RTE block number, the number of tracks, and the number of blocks per track (DVP04 through DVP07).

To execute a control 76B using the CN command:

```
CI> CN,lu,76B,p1,p2,p3,p4    where: p1    = number of tracks on the LU
                                p2    = number of 128-word blocks/track
                                p3,p4 = starting block number (double
                                        integer)
```

For example, to alter the RTE block information of LU 10 such that it has 400 blocks using 64 blocks per track and starting at block 65536, enter the following:

```
CI> cn,10,76b,400,64,1,0
```

Note that this command changes the track map information in memory only; if the system is rebooted, the track map information will be defined by the values given during system generation.

Driver Parameter Table Use in DDQ30

Driver Parameter Table usage in DDQ30																	
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP 01	D	Reserved, set to 0													SCSI Addr		
DVP 02	D	Reserved, set to 0				Unit No.			Reserved, set to 0								
DVP 03	Pass-Through Fence						Reserved, set to 0										
DVP 04	High Order word of starting block number																
DVP 05	Low order word of starting block number																
DVP 06	Number of Tracks on LU																
DVP 07	Reserved, set to 0											# of RTE blocks/track					
DVP 08	S	E	Reserved													BF	

- D:** Disconnect. When this bit is set, the driver does not disconnect until the transaction is complete.
 This bit should be set for devices that do not retry after a reselection timeout. Setting this bit can impact performance, but may be required for such devices as the C1701C Magneto-Optical Disk drive when it is connected to the same bus as a DAT drive.
 The first time a device is accessed, a logical OR is performed on the D-bits in DVP 01 and DVP 02 and the result stored in the D-bit of DVP 01. The D-bit of DVP 02 is then cleared.

Pass-Through Fence:

RTE block number (default is 24) to indicate *pass-through mode*. If the number of RTE blocks to be transferred (to or from device) is greater than or equal to this number, then enable pass-through mode. Pass-through mode allows data to be transferred directly between the HP 1000 and the SCSI device. Memory on the SCSI interface card is not used for data transfers in pass-through mode.

- S:** If set, the driver manages spin-up and spin-down.
E: If set, the disk is ejected on last dismount.
BF: Blocking factor minus 1: 0 = 256 bytes/sector
 1 = 512 bytes/sector
 2 = 768 bytes/sector
 3 = 1024 bytes/sector

Logical Unit Status (DVT Word 6)

DVT word 6 contains the status of the LU. LU status is returned in the A-Register after any request call or status request call. The DVT word 6 bits have the following functions:

DVT word 6 Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									WP			R		B	E

- WP:** If set, this bit indicates that the tape is write-protected.
- R:** If set, the driver recovered from a soft error.
- B:** If set, this bit indicates that the device is busy.
- E:** This bit is an error indicator. The system sets this bit if there is an error code in DVT word 16.

Driver Error (DVT Word 16)

DVT word 16, bits 5 through 0 (zero) contain error codes for the DDQ30 device driver. The error codes are as follows:

<u>Dec.</u>	<u>Octal</u>	<u>Error and Solution</u>
0	0B	= Normal request completion.
1	1B	= Illegal request. Check the syntax of your EXEC request call. The EXEC call could have illegal parameters or specify an unused LU or an unimplemented command. Retry the request after you check it.
2	2B	= Device not ready. The drive is offline, but media is present in the drive. If this occurs, first try unloading, then reloading the tape. Second, send a SCSI Command Descriptor Block (CDB) to load the tape. If this does not work, ensure that the device is correctly connected, not busy, and correctly generated into the system.
3	3B	= Device timeout. Use the TO command to increase the timeout period.
5	5B	= Transmission error.
6	6B	= Device write protected. Remove the tape's write protection.
7	7B	= Address error. Ensure that the SCSI address is correct.
10	12B	= Disk fault. Check the disk for any problems.
12	14B	= Insufficient driver table space generated. Generate more driver table space.
21	25B	= No disk in drive.
22	26B	= Incompatible cartridge. Change cartridge.
23	27B	= Medium uninitialized or format corrupted.
24	30B	= No spares available.
25	31B	= Automatic re-allocation failed.
26	32B	= Defect list update failed.
27	33B	= Defect list not available.
29	35B	= Illegal Logical Block Address.

All codes of 30 decimal (36B octal) or higher are interface driver error codes.

Transmission Log (DVT Word 17)

DVT word 17 contains the transmission log which is the length of the data transferred.

SCSI Status and Transaction Status (DVT Word 18)

DVT word 18 contains the SCSI status of device driver DDQ30. The SCSI status returned in DVT18 (bits 7 – 0) is as follows::

xx00000xB = No error.
xx00001xB = Check condition.
xx00010xB = Condition met.
xx00100xB = Busy.
xx01000xB = Intermediate.
xx01010xB = Intermediate condition met.
xx01100xB = Reservation conflict.
xx10001xB = Command terminated.
xx10100xB = Queue full.

The transaction status returned in DVT18 (bits 15 – 8) is as follows:

0 = No error.
1 = SCSI error. Check the sense code and additional sense code to identify the specific error.
2 = Selection/reselection timeout. Check to ensure that the device address is correct.
3 = Data error. This indicates a hardware error.
4 = Bus parity error. This indicates a hardware error.
5 = Reset.
6 = Illegal bus free. This indicates a hardware error.
7 = Abort request. The driver has sent an abort request.
8 = Illegal request. This indicates an illegal script type.
9 = Firmware error.

SCSI Sense Key and Additional Sense Code (DVT Word 19)

DVT word 19 contains the sense key and additional sense code. The Error Messages appendix in the *HP 12016A SCSI Installation and Reference Manual*, part number 12016-90001, describes DVT word 19 and gives an example program to access it. The sense code is returned in the first byte (bits 0 through 7) and the additional sense code is returned in the second byte (bits 8 through 15).

Driver Communication Word (DVT Word 20)

DVT Word 20 Bits:														
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0														
DVT 20	I	L	0	M	S	1	R	R	D	P	DVT priority			

I = Initial Flag	Set by generator, cleared on first entry to driver.
L = Locked	Eject button has been disabled.
M = Mount	Set if disk has been mounted.
S = Spinning	Set if disk is spinning.
R = Retry Counter	Counts up to three retries if the disk is busy.
D = Delay	Set if driver is delaying, next timeout is expected.
P = Powerfail Retry	During powerfail, a protocol failure may occur. This bit is set during the second attempt at an I/O request.

CS/80 Disk Device Driver DD*33

DD*33 is the driver designed to execute in an RTE-A Operating System for driving the following CS/80 devices:

HP 7908	16-Mb fixed disk with CTD*
HP 7911	28-Mb fixed disk with CTD*
HP 7942	24-Mb fixed disk with CTD*
HP 7946	55-Mb fixed disk with CTD*
HP 7912	65-Mb fixed disk with CTD*
HP 7914	128-Mb fixed disk with CTD*

* Cartridge Tape Drive (CTD) is integrated with this disk drive.

HP 9133D	15-Mb fixed disk with 630 kb microfloppy
HP 9133H	20-Mb fixed disk with 630 kb microfloppy
HP 9133L	40-Mb fixed disk with 630 kb microfloppy
HP 9134D	15-Mb fixed disk
HP 9134H	20-Mb fixed disk
HP 9134L	40-Mb fixed disk
HP 7941	24-Mb fixed disk
HP 7907	40-Mb fixed/removable disk
HP 7945	55-Mb fixed disk
HP 9144	Standalone CTD
HP 7933	404-Mb fixed disk
HP 7935	404-Mb removable media disk
HP 9122D	Floppy disk

DD*33 provides the software interface to handle device concerns, while the HP-IB interface card driver, ID*37, performs the actual I/O instructions to the interface card. Access to DD*33 is achieved via standard EXEC requests, as discussed in the following sections.

DD*33 Disk Read/Write Request

The call sequences for the DD*33 read and write request are:

Read Request:	CALL EXEC (1 , cntwd , bufr , bufln , track , sector)
Write Request:	CALL EXEC (2 , cntwd , bufr , bufln , track , sector)

Read/Write Request Control Word *cntwd*

Figure 2-40 shows the format of the control word (*cntwd*) used for a disk read or write request to DD*33.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	UE	0	77B						LU					

Figure 2-40. Read/Write Request *cntwd* Format

The UE (User Error) and LU (Logical Unit assignment) bits are as defined in the Read/Write Request Conventions section of Chapter 1.

Bit 12, the Z-bit, must be set to zero because *track* and *sector* are integer variables, not control buffer descriptions.

bufr* and *bufln

The I/O buffer used in the EXEC request is described by parameters *bufr* and *bufln*, and are as defined in the Read/Write Request Conventions section of this manual.

track* and *sector

The *track* and *sector* parameters are integer decimal values that describe the track and logical sector positions to which the data buffer is written, or from which data is read.

Note that *sector* must be an even integer value for all read or write requests.

Because the CS/80 disks use block addressing, DD*33 must convert track and sector addresses to block addresses. Using the track and sector values, the driver computes the block address using the following formula:

$$\text{Address} = (\text{Trk No.} * \text{No. of sectors/track}) + (\text{sector No./2}) + \text{block No.}$$

where: Address is the physical block address and block No. is the starting block in the driver parameter area.

The sector number passed in the EXEC call is a logical value that must be converted to the physical sector value. Therefore, the logical sector value passed must be even.

The spare command can be used only with CS/80 disks, including the CTD disk cache memory area. For additional information, refer to the *RTE-A Utilities Manual*, part number 92077-90004.

Logical/Physical Sector Correspondence

The number that identifies the density of information that can be packed on a disk cartridge is the “physical” sector value. This value is fixed for each disk. Each physical sector holds 256 bytes of information. A physical sector is sometimes called a block (256 bytes). A physical sector, or a block, is twice the length of a “logical” sector (128 bytes) used in program calls.

A- and B-Register Contents

The A-Register will contain word 6 of the device table (DVT) after each EXEC read/write request to a disk. The format of word 6 is the same as that of *stat1* returned after an EXEC 13 status request, to be discussed later in this section.

The B-Register will contain the positive number of words (if *bufln* > 0) or characters that were transmitted during the last read/write request (if *bufln* < 0). The following FORTRAN sequence will read from track 100, sector 32 of a HP 7908 disk with LU 12. The data will be read into the buffer called BUFR.

```

:
IMPLICIT INTEGER (A-Z)
DIMENSION BUFR (128)
:
BUFLN = 128
TRACK = 100
SECTOR = 32
CNTWD = 12 + 7700B
CALL EXEC(1, CNTWD, BUFR, BUFLN, TRACK, SECTOR)
:
END

```

Note: If track = -1 and sector = -1,
 BUFLN = -20 returns the last bad status
 BUFLN = -40 returns the last bad status and reads the status again from the disk.

DD*33 Disk Control Request

The call sequence for the DD*33 disk control request is:

```
CALL EXEC(3, cntwd[, ,pram1[, ,pram2[, ,pram3[, ,pram4]]]])
```

Control Request *cntwd*

The control word (*cntwd*) format for DD*33 disk Control request is shown in Figure 2-41. The LU bits are as defined in Chapter 1. The *pram1* and *pram2* parameters of the EXEC call are used with the available function codes as defined in the following paragraphs.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Function Code						LU					

Figure 2-41. DD*33 Disk Control Request *cntwd* Format

Function Code 26B = Set User Request Timeouts

Function Code 26B changes the timeout and enables/disables timeout retry control on a device. For example:

```
CI> CN,lu,26B,p1,p2      where: p1 = timeout in tens of milliseconds, must be
                             greater than the interface timeout.
                             p2 =  1  to enable timeout retry,
                             -1  to disable timeout retry,
                             0   to indicate no change.
```

Substituting values:

```
CI> CN,30,26B,2000,-1  changes the timeout of LU 30 to 20 seconds and
                             disables timeout retry.
```

Function Code 76B = Alter Driver Parameter table

Function Code 76B changes the track map information in the DVT for a disk device. For example:

```
CI> CN,lu,76B,p1,p2,p3,p4 where: p1 = number of tracks
                             p2 = number of 128-word blocks/track
                             p3,p4 = starting block number (double
                             integer)
```

Note that Function Code 76B changes the track map information in memory only; if the system is rebooted, the track map information will be defined by the values given during system generation.

DD*33 Status Request

The call sequence for the status request is:

```
CALL EXEC(13,cntwd,stat1[,stat2[,stat3[,stat4]]])
```

Status Request Control Word *cntwd*

Figure 2-42 shows the control word format for an EXEC status request for devices driven by DD*33.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-42. DD*33 Status Request *cntwd* Format

stat1 and *stat2* Status Parameters

The format of the returned status parameters, *stat1* and *stat2* is shown in Figure 2-43.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type = 26, 33					Status Bits								
<i>stat2</i>	AV		Interface Type = 37				0	I/O Select Code								

Figure 2-43. *stat1* and *stat2* Status Words Format

Device Type: disk=33B, CTD=26B

Status Bits:

- 0 Severe Error.
- 1 Channel Errors. Set for any Reject error, or DMA length error. If this bit is set, the severe bit (bit 0) is also set.
- 2 Not ready (unit not ready for access). If this bit is set, severe bit is also set.
- 3 Fault (a Fault error has occurred). If this bit is set, severe bit is also set.
- 4 Uninitialized Medium. If this bit is set, severe bit is also set.
- 5 EOF/EOV. If this bit is set, severe bit is also set.
- 6 Unrecoverable Data/(recoverable data), set for uncorrectable or marginal errors. Severe bit set only for unrecoverable data.
- 7 Write protected volume. Severe bit set only if write failed.

STAT3 and STAT4 Status Parameters

The formats of status parameters *stat3* and *stat4* for Z = 0 and 1 are shown in Figures 2-44 and 2-45.

For Z=0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat3</i>	First Word of the Driver Parameter Area															
<i>stat4</i>	Second Word of the Driver Parameter Area															

Figure 2-44. *stat3* and *stat4* for Z=0

For Z=1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat3</i>	Buffer to Return the Driver Parameter Area															
<i>stat4</i>	Length of Buffer Described by STAT3															

Figure 2-45. *stat3* and *stat4* for Z=1

DD*33 Driver Parameter Area

Table 2-8 summarizes the words defined in the driver parameter area for DD*33.

Table 2-8. DD*33 Driver Parameter Area

	Disk Parameter Description				CTD Parameter Description				
	15	8	7	0	15	8	7	0	
DP1	HP-IB Address				0	HP-IB Address			
DP2	Unit Number		Volume Number		Unit Number		Volume Number		
DP3	MSB				C	Unit Number Disk		Volume Number	
DP4	Starting Block Address 3 Words				Starting Block Address 2 Words				
DP5	LSB								
DP6	Number of Tracks				Address of First Cache Block				
DP7	Number of 128-Word Sector/Block								
DP8	Reserved				Reserved				
					C = 1 if Cartridge Tape is Cached				

DD*33 Extended Status

As discussed in the Extended Device Status section of Chapter 1, the extended status can be obtained by making a call to RMPAR after an EXEC read/write request to determine the status of the request. The format of the returned status words is shown in Figure 2-46.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>xstat</i> (1) / DVT16	X		0						Error Code							
<i>xstat</i> (2) / DVT17	Transmission Log = 0															
<i>xstat</i> (3) / DVT18	Extended Status #1 (Range 1-4)															
<i>xstat</i> (4) / DVT19	Extended Status #2															

Figure 2-46. DVT16 to DVT19 Status Words Format

DVT16 bits 14 and 15 are system defined and are not significant to the user. The error codes returned in bits 0 through 5 are:

- 0 = Normal request completion.
- 1 = Illegal request (for example, track or sector address error).
- *2 = Device not ready.
- *3 = Device has timed out.
- *5 = Transmission error (see error section for more information).

- *6 = Device write protected.
- *10B = Device fault. Hardware error.
- *77B = Driver Request Retry.

* Codes not seen by the caller unless UE (bit 13) was set.

DVT17, the transmission log, is set to 0.

DVT18 defines the error class: Class 1, Reject Errors; Class 2, Fault Errors; Class 3, Access Errors, and Class 4, Information Errors.

DVT19 defines the status for the error class shown in DVT18.

A special EXEC call is provided to retrieve all of the status information returned by the controller. This information contains only the target address after a normal completion. Note that it is possible for another request affecting status to be received after the faulty operation and before the status request. DD*33 automatically requests and preserves status after an error return from the controller. To obtain a valid status for a CTD device, do a clear cache call before the status request.

The request requires a length of at least 10 words to retrieve the status in the driver, or a length of at least 20 words for both the status in the driver and the current status words in the controller.

The call sequence for the status request is:

```
CALL EXEC(code , cnwd , bufr , bufln , opt1 , opt2)
```

where:

- code* = Request code = 1 for read
- cnwd* = Control word Bits 0-5 = Logical unit
- bufr* = Buffer for status data
- bufln* = 10 for status length (status in driver)
20 for status length (driver status and current drive status)
- opt1* = -1
- opt2* = -1

CS/80 devices return 10 words of status information. This information is ordered by severity, so that the first words of status show the most serious errors. The 10 words are:

- word 1 = Identification field, unit and volume of status information.
- 2 = Reject errors = most serious errors.
- 3 = Fault errors.
- 4 = Access errors.
- 5 = Information errors.
- 6-10 = Parameter area for words 2-5.

The parameter area is a five-word parameter field in decimal representation. Interpretation of the parameter field for all errors, unless noted otherwise in the specific error field definition, is as follows:

1. Convert decimal values in each field to a hexadecimal value. Arrange, for convenience, the new values as shown:

```

AAAAAA = H0H1
BBBBBB = H2H3
CCCCCC = H4H5
DDDDDD = H6H7
EEEEEE = H8H9

```

2. H0 through H9 are the hexadecimal CS/80 parameters P1 through P10 respectively.
3. P1 through P6 form the new target block address on the disk drive. This information can usually be disregarded except when noted differently in the specified error field description.
4. P7 through P10 are the four most recent errors from the CS/80 disk drive.

The identification field format is:

```
VVVUUUUSSSSSSSS
```

where:

```

VVVV = volume number for this status
UUUU = unit number for this status
SSSSSSSS = other unit requesting service ( -1 for none)

```

Reject Errors Field (Class 1)

The reject errors field format is:

x	x	1	x	x	2	3	4	5	6	7	x	8	x	x	x
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

- | | | |
|----|---|--|
| 13 | 1 | Channel Parity Errors (20000B) |
| | | An channel command was received without odd parity. Generally indicates a programming error. |
| 10 | 2 | Illegal Opcode (2000B) |
| | | An unrecognizable opcode was received, most likely a programming error. |
| 9 | 3 | Module Addressing (1000B) |
| | | Illegal Unit or Volume was received. |
| 8 | 4 | Address Bounds (400B) |
| | | The target address has exceeded the bounds for this device. |
| 7 | 5 | Parameter Bounds (200B) |
| | | A parameter (unit, volume or address) is out of bounds for this device. |

- | | | |
|---|---|--|
| 6 | 6 | <p>Illegal Parameter (100B)</p> <p>A parameter field has the wrong length for the opcode preceding it.</p> |
| 5 | 7 | <p>Message Sequence Error (40B)</p> <p>The message sequence has been violated. This may be a programming error, that is an execution message was required, but none was sent, or something was lost and the request should be retried.</p> |
| 3 | 8 | <p>Message Length Error (10B)</p> <p>The length of the execution message does not match the length command. A programming error, or a bus fault has occurred.</p> |

Fault Errors Field (Class 2)

The fault errors field format is:

x	1	x	2	x	x	3	x	4	x	5	6	7	x	8	9
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

- | | | |
|----|---|--|
| 14 | 1 | <p>Cross Unit Error (40000B)</p> <p>An error has occurred during a Copy Data. The offending units are listed in the parameter area.</p> |
| 12 | 2 | <p>Controller Fault (10000B)</p> <p>A hardware controller fault has occurred. Run the diagnostic or call your local Hewlett-Packard Customer Engineer.</p> |
| 9 | 3 | <p>Unit Fault (1000B)</p> <p>A hardware unit fault has occurred. Run the diagnostic or call your local Hewlett-Packard Customer Engineer.</p> |
| 7 | 4 | <p>Diagnostic Result (200B)</p> <p>The hardware failed a diagnostic test, indicated in the parameter field as follows:</p> <p>P1–P2 – contain most probable and second most probable cause of the error.
 P3–P4 – contain the errors associated with P1–P2, respectively.
 P5–P6 – are not used.
 P7–P10 – contain error information as usual.</p> |
| 5 | 5 | <p>Release Required for Operator Request (40B)</p> <p>The drive wishes to release to service an operator request, that is, load, unload. Programs should grant or deny this request as they see fit.</p> |
| 4 | 6 | <p>Release Required for Diagnostic (20B)</p> <p>Release is required for diagnostics initiated from the front panel. This request must be granted to complete a CTD load, but may be denied if necessary.</p> |

- 3 7 Release Required for Internal Maintenance (10B)
Release is required so that the drive can update its logs. If release is not granted, error information may be lost.
- 1 8 Powerfail (2B)
Power to the drive failed, or state was lost.
- 0 9 Retransmit
The drive has returned from automatic release.

Access Errors Field (Class 3)

The access errors field format is:

1	2	3	4	5	6	x	x	7	8	x	9	10	x	x	x
---	---	---	---	---	---	---	---	---	---	---	---	----	---	---	---

- 15 1 Illegal Parallel Operation (100000B)
The operation requested cannot be executed while others are in progress. Parallel operations are not supported on RTE-A.
- 14 2 Uninitialized Media (40000B)
Unformatted or unusable medium has been loaded. The medium should be formatted before further access is attempted.
- 13 3 No Spares Available (20000B)
No spares are left for the spare block command. The medium should be replaced or the drive should be serviced to determine if more spares can be recovered.
- 12 4 Not Ready (10000B)
The selected unit is not ready for access. The medium may not be loaded or the device may not be on line yet.
- 11 5 Write Protect (4000B)
The device is write protected.
- 10 6 No Data Found (2000B)
A block accessed during read has not been written. This usually occurs on a tape that has not been initialized.
- 7 7 Unrecoverable Data Overflow (200B)
The last transaction had more than one unrecoverable data error. Address in parameter area is first error.
- 6 8 Unrecoverable Data error (100B)
The controller was unable to read data, and the data cannot be totally recovered. Retries may succeed in recovering it partially.

- | | | | |
|---|----|---------------------|--|
| 4 | 9 | End of file (20B) | |
| | | | An end of file mark was encountered during the read. The address in the parameter area is the address of the block after the end of file mark. |
| 3 | 10 | End of Volume (10B) | |
| | | | The last access attempted to access past the end of the volume. The parameter area contains the address of the last block on the volume. |

Information Errors Field (Class 4)

The information field format is:

1	2	3	4	5	x	x	6	x	7	8	9	x	10	x	x
---	---	---	---	---	---	---	---	---	---	---	---	---	----	---	---

- | | | | |
|----|---|---|---|
| 15 | 1 | Release for Operator request (100000B) | |
| | | | The controller requests release to allow load/unload, save/restore. Release may be ignored or denied if necessary. System grants release during idle time if it sees a release request. |
| 14 | 2 | Release for diagnostic request (40000B) | |
| | | | Release for diagnostic generated by front panel or self test. If ignored, system grants release during idle time. |
| 13 | 3 | Release for Internal Maintenance (20000B) | |
| | | | Request release to update error logs or perform maintenance. If release is not granted, system will release during idle time. |
| 12 | 4 | One Spare Left (10000B) | |
| | | | There is only one spare block left on this volume. The media should be backed up and replaced or serviced to see if spares can be recovered. |
| 11 | 5 | Data Overrun (4000B) | |
| | | | During a transfer a latency occurred when the system could not accept data as fast as the device needed to. No data was lost. |
| 8 | 6 | Auto Sparing Invoked (400B) | |
| | | | A defective block was automatically spared. No data was lost. |
| 6 | 7 | Recoverable Data Overflow (100B) | |
| | | | The last request generated more than one recoverable data error. The first error is in the parameter area. |
| 5 | 8 | Marginal Data (40B) | |
| | | | The last request generated an error which was recovered with difficulty. The block should be spared before it becomes uncorrectable. |

- 4 9 Recoverable Data Error (20B)
 The last request generated an error which was recovered by retry or error correction. No data was lost.
- 2 10 Maintenance Track Overflow (4B)
 The maintenance track on the device has overflowed. The error and fault logs are full. The drive should be serviced to recover this information before it is lost. The Parameter Field contains the parameters appropriate to the most serious error seen. This area contains the target or fault address except for the following errors:
- Spare Block – After the spare block the parameter area contains the address of the beginning of the affected area in the first 3 words. The last 2 words contain the length in bytes of the affected field.
- Cross Unit – The parameter area is a list of units which experienced errors. Each unit is 1 byte long, and the list is terminated by a byte of 377B.
- Diagnostic Results – The first 3 words of the parameter area contain the diagnostic numbers which failed. Each number is 1 byte long.

DD*33 CTD Read/Write Request

The call sequences for the CTD read/write requests are:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln* , *hiblk* , *loblk*)
 Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln* , *hiblk* , *loblk*)

Figure 2-47 shows the format of the control word (*cntwd*) used for a CTD read or write request to DD*33. Required parameters are discussed in the following paragraphs.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	UE	Z	Subfunction						LU					

Figure 2-47. CTD Read/Write Request *cntwd* Format

The UE (User Error) and LU (Logical Unit assignment) bits are as defined in the Read/Write Request Conventions section of Chapter 1. Bit 12, the Z-bit, must be set to zero because *hiblk* and *loblk* are integer variables, not control buffer descriptions. Set bits 6 through 11 to 1 for a non-cached request and to 2 for a request using the disk cache. See the DD*33 CTD Control Request section for a description of the CTD cache.

bufr* and *bufln

The I/O buffer used in the EXEC request is described by parameters *bufr* and *bufln*, and are as defined in the Read/Write Request Conventions section of Chapter 1.

hiblk* and *loblk

The block address of the block of data to be written or read is given as a two-word integer with the most significant 16 bits in *hiblk* and the least significant 16 bits in *loblk*. The tape begins at block 0. Each block on the tape consists of 512 words, whereas a disk block consists of 128 words.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each EXEC read/write request to the CTD. The format of word 6 is the same format as *stat1* returned after an EXEC 13 status request, discussed earlier in this section.

The B-Register will contain the positive number of words (if *bufln* > 0) or characters (if *bufln* < 0), that were transmitted during the read/write request.

DD*33 CTD Control Requests

Driver DD*33 provides CTD control requests for such functions as writing file marks and unloading tape. Also, if the CTD is integrated with the disk drive, control requests can implement cached data transfers with significant improvement in the transfer rate. The following subsections cover control requests, including cached data transfer.

There are two types of CTDs (listed at the beginning of this section). They are both CS/80 devices, but one is integrated with the disk drive while the other is a standalone device. Because the integrated CTD shares a common controller with the disk drive, it can use cached mode, transferring data in large blocks through a cache area on its associated disk.

The standalone CTD cannot use a disk cache. If the driver receives a cached read or write request for a standalone CTD, it will treat the request as uncached. Similarly, cache control requests for a standalone CTD, such as close cache, will be ignored. Thus, programs written for cached operation will work with an uncached device.

Using the Cartridge Tape Drive

The cartridge tape drive is a high capacity device. It requires that data be transferred in a continuous streaming mode, with or without buffering.

Nonbuffered control, read, and write requests go directly to the CTD. Buffering involves creating a disk buffer called the disk cache at generation time. The disk cache is 64K bytes long in an area reserved on the associated disk drive. This area is used only for the CTD disk buffering. Buffering improves transfer times to the CTD by at least an order of magnitude for total transfers over 8K bytes on a tape. Buffering is recommended because it maximizes tape usage. Programs which access the CTD should use cached requests whenever possible to increase performance. How the disk buffer (cache) works is explained below.

If a program uses the CTD, it should lock the LU of the CTD to itself to insure the integrity of the disk buffer and tape. At this point it is necessary to initialize the disk buffer and cache pointers to a known state. This is accomplished by issuing EXEC calls to the CTD LU, clearing the cache. This will clear all the cache pointers and check the CTD for write-protected and certified tapes. Now it is possible to read and write to the cache.

The cache is set up for forward reference. This means that the blocks in the cache are a duplicate of a contiguous set of blocks on the CTD beginning with the first block accessed. This set of blocks is accessed from the disk buffer until the buffer is overflowed, closed, or has a file mark written to it.

At this point, the buffer will be posted if it has been written into and a new buffer will be brought in, if necessary. If an end of file mark is encountered on a read, it will show up when that block is

accessed. This process of reading and writing continues until all the desired data is transferred. The program should then close the cache to ensure that the last set of blocks written is posted and the cache is reset. The program also has the option of writing an end-of-file mark on the tape, and it may also programmatically unload the CTD. The close, write file mark, and unload functions are all implemented as EXEC control calls.

Clear Cache

Cache clearing is accomplished with a subfunction code of 1 in an EXEC control call. The subfunction codes are explained in Table 2-9. This control call allows a program to clear the CTD cache before the CTD is accessed. If the cache is not cleared, it is possible to write random data on the CTD. This call also checks the CTD for write-protected and certified cartridges. This status is returned in the A-Register regardless of whether or not caching is used.

Table 2-9. EXEC Function and Subfunction Codes

Function	Subfunction	Description
1	0	<i>hiblk</i> = -1 and <i>loblk</i> = -1 Dynamic Status Request 10 or 20 words in <i>bufr</i> , depending on length
1	1	Block Addressing
1	2	Block Addressing, Cached Requests (CTD only)
1	21B	Return Describe Information
2	23B	Direct Protocol Control mode
2	1	Block Addressing
2	2	Block Addressing, Cached Requests (CTD only)
	3	Write EOF, post and Close Cached (CTD only)
2		Write File Mark at address in IADD1 and IADD2
3	23B	Direct Protocol Control mode
	1	Clear Cache (CTD only)
3		Resets cache and checks write protect
		Post and Close Cache (CTD only)
3		Writes cache to tape if necessary
	3	Unload tape (CTD only). Causes tape to rewind and unload

Close Cache

Cache closing is accomplished with a subfunction code of 2 in the EXEC control call. This call is used to terminate cleanly the use of the disk buffer for the CTD. This call posts the disk buffer to the CTD if the buffer has been written and leaves the cache in a clear state. This call guarantees that blocks written into the disk buffer are moved to the CTD. If no cache is present, driver DD*33 takes no action and returns to the program.

Unload Tape

Unloading tape from a cartridge tape drive can be done programmatically by using a subfunction code of 3 in an EXEC control call. This call rewinds the tape so that the tape can be removed. It can be used for both cached or uncached CTDs. Then DD*33 returns to the program.

An example program using the CTD control calls is shown below.

```
INTEGER CTD, disk, TRACK, SECTOR, LENTH
INTEGER BLOCK(2), BUFFR(8192), SIZE(2)
INTEGER*4 DBLCK, DSIZE
EQUIVALENCE (DBLCK, BLOCK(1)), (DSIZE, SIZE(1))
.
C
C SET UP THE CTD, DISK LUS, AND INITIALIZE LENGTH AND BLOCK
C
    CTD = 10
    DISK = 20
    LENGTH = 8192
    DBLCK = 0
    SECTOR = 0
C
C FIND OUT HOW LONG THE TAPE IS (BUFFR(17) AND BUFFR(18))
C REFER TO THE CS/80 REF. MANUAL FOR DESCRIBE FIELDS
C
    CALL EXEC (1, CTD+2100B, BUFFR, -37, 0, 0)
    SIZE(1) = BUFFR(17)
    SIZE(2) = BUFFR(18)
C
C CLEAR THE CTD CACHE
C
    CALL EXEC (3, CTD+100B)
C
C MOVE DATA FROM DISK TO CTD BY WAY OF THE CACHE
C
    DO 10 TRACK = 0,100
    CALL EXEC (1, disk, BUFFR, LENTH, TRACK, SECTOR)
    CALL EXEC (2, CTD+200B, BUFFR, LENTH, BLOCK(1), BLOCK(2))
    10 DBLCK = DBLCK + LENTH/512
C
C CLOSE THE CACHE NOW AND WRITE A FILE MARK
C
    CALL EXEC (2, CTD+300B, BUFFR, 0, BLOCK(1), BLOCK(2))
C
C UNLOAD THE TAPE FOR THE USER
C
    CALL EXEC (3, CTD+300B)
.
.
```


DD*33 Direct Disk Control

Direct disk control can be done through a group of routines contained in a command library, \$DTCLB. The control functions available and the associated routines are listed below.

Command	Routine
Cancel	XCNCL
Channel Independent Clear	XCICL
Cold Load Read	XCOLD
Complementary Commands	XCOMP
Copy Data	XCOPY
Describe	XDESC
Initialize Media	XINMD
Initiate Diagnostic	XDIAG
Initiate Utility	XUTIL
Locate & Read	XLCRD
Locate & Verify	XL CVF
Locate & Write	XL CWR
Read Loopback Test	XRLPB
Release	XRELS
Release denied	XRELD
Request status	XRQST
Selected Device Clear	XSDCL
Spare Block	XSPRE
Unload	XUNLD
Write File Mark	XFMRK
Write Loopback Test	XWL PB

Calling Format

The calling format for the direct disk control routines is

```
CALL routine (lu , iaddr , icode , ibuf)
```

where:

- lu* Target LU. Gets routine to the correct driver.
- iaddr* HP-IB Address of CS/80 drive.
- icode* A 24-word array describing parameters for all the possible complementary commands. *icode* is a static buffer, not changed by \$DTCLB routines. The meaning of each word is described in Table 2-10.
- ibuf* An array. The first 60 words are reserved for use by the routine in building a command. The use of words 61 on is dependent upon the particular routine. On return, words 58 through 60 contain the following:
 - ibuf*(58) A-Register contents
 - ibuf*(59) B-Register contents
 - ibuf*(60) QSTAT

If *ibuf*(60) equals 1, the full status is returned in *ibuf*(2) through *ibuf*(11). The QSTAT associated with the request status command is returned in *ibuf*(1).

Table 2-10. Complementary Command Array (*icomp*)

Word	Meaning
1	Unit number
2	Volume
3	Address Mode 0 → Block 1 → 3 vector
4	Target address
5	Target address
6	Target address
7	Set Block Displacement -1 → No block displacement 1 → Valid block displacement in words 8 and 9
8	Block displacement
9	Block displacement (Note: Block displacement is a signed 6-byte integer. The first two byte will be filled with the sign bit)
10	Set length flag -1 → No length 1 → Valid length in 11 and 12
11	Length
12	Length
13	Burst size (Set Bit 15 to tag bursts with EOI)
14	RPS time 1
15	RPS Time 2
16	Number of read retries
17	Set status mask flag -1 → No status mask 1 → Valid status mask in 18–22
18	Status mask
19	Status mask
20	Status mask
21	Status mask
22	Set release -1 → No Set release
23	Set address return mode 0 → Block 1 → 3 vector
24	Set device specific options

To use a complementary command, set the appropriate words in *icomp*. All other words should be set to -1. Not all complementary commands are compatible with all of the commands. The use of word 61 or greater in *ibuf* is routine-dependent; such use is described below.

Routine	Use
XCNCL	Not used
XCICL	Not used
XCOLD	Return area for read data
XCOMP	Not used
XCOPY	Source: Destination: <i>ibuf</i> (61) Unit <i>ibuf</i> (67) Unit <i>ibuf</i> (62) Vol <i>ibuf</i> (68) Vol <i>ibuf</i> (63) Addr Mode <i>ibuf</i> (69) Addr Mode <i>ibuf</i> (64) Addr <i>ibuf</i> (70) Addr <i>ibuf</i> (65) Addr <i>ibuf</i> (71) Addr <i>ibuf</i> (66) Addr <i>ibuf</i> (72) Addr
XDESC	Return area for Describe data
XFMRK	Not used
XFORM	<i>ibuf</i> (61) = Option <i>ibuf</i> (62) = Interleave Factor
XINTP	<i>ibuf</i> (61) = Option
XDIAG	<i>ibuf</i> (61) = Not used <i>ibuf</i> (62) = Not used <i>ibuf</i> (63) = Length in bytes of param string <i>ibuf</i> (64) – <i>ibuf</i> (66) Parameter string left justified
XLCRD	Return area for read data
XLCVF	Not used
XLCWR	Source area for write data
XRELS	Not used
XRELD	Not used
XRLPB	<i>ibuf</i> (61 – 62) Loopback Length in bytes On Return: Return area for Loopback data
XRQST	Return area for Status information
XSDCL	Not used
XSPRE	Parameter field
XUNLD	Not Used
XUTIL	<i>ibuf</i> (61) Utility number <i>ibuf</i> (62) Type of execution message 0 ==> No execution message 1 ==> Device receives mesg 2 ==> Device sends mesg <i>ibuf</i> (63) Length in bytes of param string <i>ibuf</i> (64)– <i>ibuf</i> (67) Parameter string left justified. <i>ibuf</i> (68) Length of execution mesg in bytes <i>ibuf</i> (69)– <i>ibuf</i> (??) Execution mesg if <i>ibuf</i> (62)=1 On return: <i>ibuf</i> (61)– <i>ibuf</i> (??) Contains execution message if execution message was sent or received
XWLPB	<i>ibuf</i> (61 – 62) = Loopback Length in bytes <i>ibuf</i> (63 – ?) Source for Loopback data

The following are examples using the direct disk control routines XLCRD, XLCWR, XLCVF, and XSPRE.

Assuming you have an LU and HP-IB address, the following is a FORTRAN calling sequence to read one block from a disk starting at block 100 for a length of 2 (disk) blocks (512 bytes) and write them to block 200 of the same disk. A verify is then issued to ensure that the data at block 200 can be read.

```
C          CLEAR ICOMP ARRAY
          DO 10 I=1,24
             ICOMP(I)=-1
10 CONTINUE
          :
          :
C
C          DO THE READ
          ICOMP(1)=0           !UNIT 0
          ICOMP(2)=0           !VOLUME 0
          ICOMP(3)=0           !BLOCK ADDRESSING
          ICOMP(4)=0           !3 WORD
          ICOMP(5)=0           !BLOCK ADDRESS
          ICOMP(6)=100         !OF 100
          ICOMP(10)=1          !INDICATE LENGTH IS SET
          ICOMP(11)=0          !DOUBLE WORD
          ICOMP(12)=512        !NUMBER OF BYTES
          CALL XLCRD(LU,IADDR,ICOMP,IBUF)
C
C          DO WRITE
          ICOMP(6)=200         !NEW BLOCK NUMBER
          CALL XLCWR(LU,IADDR,ICOMP,IBUF)
C
C          VERIFY BLOCK WRITTEN
          CALL XLCVF(LU,IADDR,ICOMP,IBUF)
          :
          :
```

Note IBUF must be at least 188 words long (60 words for XLCRD and 128 words for the data from the disk).

Note that in the example above, only those values that change for the next call need to be changed in the ICOMP array. In the call to XLCWR, the only change is in the block number from the previous call. In the call to XLCVF, the same area that was just written is also being verified, so no change is necessary in the ICOMP array.

If a series of blocks is to be read, equate ICOMP(5) and ICOMP(6) to a 4-byte integer and keep incrementing the integer by one (or more) for each successive read. Using four of the available six bytes is sufficient to cover the maximum block address possible. The spare command is very similar to the locate and read command in terms of calling sequence. To spare block 100, you would go through the same sequence as for XLCRD above. Setting the length in ICOMP would have no effect, however, on the spare command.

XINMD

To format disk unit 0, volume 0, retaining all factory and field spares, and setting a block interleave factor of 1:

```
DO I=1,24
    ICOMP(I)=-1
ENDDO
.
.
.
ICOMP(1)=0          !UNIT 0
ICOMP(2)=0          !VOLUME 0
IBUF(61)=0          !KEEP FACTORY AND FIELD SPARES
IBUF(62)=1          !INTERLEAVE
C
CALL XINMD(LU,IADDR,ICOMP,IBUF)
```

XRELS/XRELD

To release unit 0:

```
DO I=1,24
    ICOMP(I)=-1
ENDDO
.
.
.
ICOMP(1)=0 !UNIT 0
CALL XRELS(LU,IADDR,ICOMP,IBUF)
```

To deny release, the sequence is exactly the same, but the call would be to XRELD.

XRQST/XDESC

Request status and describe have similar calling sequences, with the exception that the set volume command has no effect on the request status command. To get the status of unit 0, the sequence would be as follows:

```
DO I=1,24
    ICOMP(I)=-1
ENDDO
ICOMP(1)=0          !UNIT 0
CALL XRQST(LU,IADDR,ICOMP,IBUF)
```

Status is returned (in packed format) in IBUF(61) onwards. IBUF(59) has the contents of the B-Register which should be the number of bytes transferred (this should be 20).

To issue a describe command, you would also set ICOMP(2) to the volume number. The description is returned (in packed format) in IBUF(61) onwards. IBUF(59) contains the number of bytes returned.

XCOPY

To copy 10 blocks (2560 bytes) on unit 0, volume 0 starting at block 100, to unit 1, volume 0, block 200, you could use the following sequence.

```
DO I=1,24
    ICOMP(I)=-1
ENDDO
.
.
.
ICOMP(1) =15      !TALK TO CONTROLLER UNIT
ICOMP(10)=1      !INDICATE LENGTH IS SET
ICOMP(11)=0      !DOUBLE WORD
ICOMP(12)=2560   !NUMBER OF BYTES
C
C  SET FROM UNIT/VOL/ADDR
.
IBUF(61)=0       !UNIT 0
IBUF(62)=0       !VOLUME 0
IBUF(63)=0       !ADDR MODE IS BLOCKS
IBUF(64)=0       !6 BYTE
IBUF(65)=0       !BLOCK
IBUF(66)=100     !ADDRESS
C
C  SET DEST UNIT/VOL/ADDR
C
IBUF(67)=1       !UNIT 1
IBUF(68)=0       !VOLUME 0
IBUF(69)=0       !ADDR MODE IS BLOCKS
IBUF(70)=0       !6 BYTE
IBUF(71)=0       !BLOCK
IBUF(72)=200     !ADDRESS
C
C  COPY DATA
C
CALL XCOPY(LU,IADDR,ICOMP,IBUF)
```

Note that the XCOPY routine returns no data to the user (except, of course, the QSTAT and A- and B-Register returns). What is placed in IBUF(61) onwards remains intact after the call. If you wish to do a series of copy commands, you could increment the block addresses in IBUF(70) through IBUF(72) and IBUF(64) through IBUF(66) by the appropriate number, then go back and call XCOPY again.

XCOMP

XCOMP sends only complementary commands in the command message. This has the effect of permanently changing the disk parameters, as opposed to sending the complementary commands with another command, which causes them to be in affect only for that command sequence. The following sequence illustrates the use of XCOMP.

```
      DO I=1,24
        ICOMP(I)=-1
      ENDDO
      .
      .
      .
C
C  CHANGE SELECTED DISK PARAMETERS
C
      ICOMP(7)=1           !FLAG FOR BLOCK DISP
      ICOMP(8)=0           !DOUBLE WORD
      ICOMP(9)=10          !BLOCK DISP OF 10 BLOCKS
      ICOMP(13)=1024       !SET BURST SIZE TO 1024 BYTES
      CALL XCOMP(LU,IADDR,ICOMP,IBUF)
```

The parameters set in the above example will remain set for all subsequent commands to the disk that do not send their own values in complementary commands. The parameters will remain set until another call to XCOMP or until the disk is powered on or cleared.

DD*33 Error Information

For unsuccessful data transmissions, the system will return the following error messages.

I/O -NR @ LUxx,D

LUxx not ready due to one of the following:

1. No disk.
2. Drive not connected.
3. Tape not inserted.

I/O -TE @ LUxx

Disk transmission error.

I/O -WP @ LUxx

Device write protected.

I/O -FA @ LUxx

Drive fault.

I/O -TO @ LUxx

Drive not connected.

In addition to the above error messages, when an error occurs, DD*33 displays the following message:

```
*LU xx {RE, FA, TE, IN, or WN} ST = xxxxx
ERROR = yyyyyy yyyyyy yyyyyy yyyyyy yyyyyy
PARAM = zzzzzz zzzzzz zzzzzz zzzzzz zzzzzz
```

where:

xx is the LU number.

RE, FA, TE, IN is the class of errors described in the DD*33 extended status section.

RE = Reject Error (Class 1)

FA = Fault Errors (Class 2)

TE = Access Errors (Class 3)

IN = Information Errors (Class 4)

WN indicates that the driver is attempting to retry the current request. This is a warning only. Note that the status, error, and parameter fields are not valid for the WN message.

xxxxxx is the status for the error class.

yyyyyy are fields for (from left to right): identification, reject, fault, access, and information. Refer to the CS/80 extended status section for more information.

zzzzzz are the five parameter fields. Refer to the CS/80 extended status section for more information.

RETRY

To provide increased reliability during adverse conditions such as an auto-restart following a powerfail, the driver will retry certain requests after a timeout. This allows the driver to work with a wide variety of drives which have different startup and self-test characteristics. When this occurs, the driver prints the following messages:

```
warning CS/80 timeout
Warning: retry, abort or powerfail recovery.
*LU xxx WN ST=xxxxxx
ERROR=yyyyyy yyyyyy yyyyyy yyyyyy yyyyyy
PARAM= zzzzzz zzzzzz zzzzzz zzzzzz zzzzzz
```

where:

xxx is the LU

xxxxxx contains a driver address for factory use only.

The yyyyyy and zzzzzz fields are not reliable during retries due to the timeout and should not be interpreted as the error and parm fields.

Note that if an attempt is made to access a drive that is turned off or not connected, a series of messages will be printed to the system console indicating that the driver is having difficulty communicating with the disk drive. After 16 retries or if the driver otherwise decides that retries are inappropriate, the following messages will be printed to the system console:

```
warning CS/80 timeout
timeout taken.
I/O device error on LU xx      The reason is:
Device time out
Device has been downed (use UP,lu to try recovery)
xx is the LU.
```

Bad Tape Indicator

A special case of the above occurs when a bad tape is inserted into a CTD integrated with a CS/80 disk. In this case, the following message is displayed:

```
*LU xx  FA ST= 200 PARM= 1536 9472 0 0 0
```

where:

`xx` is the LU number of the CS/80 LU being accessed when the tape driver discovered that the tape was bad. The LU can be any LU on the CS/80 drive.

`FA` indicates a fault error (Class 2).

`200` indicates diagnostic results are being returned in the parameter fields.

The numbers 1536 and 9472 are internal diagnostics performed by the tape drive and indicate that the tape drive mechanism is not working as expected. This is usually caused by a faulty tape; you should replace the tape.

Note DD*33 retries the last operation after encountering a diagnostic result fault.

Disk Interface Driver ID*27

ID*27 is the disk interface driver for the following disks:

- HP 248x Integrated 10 Mbyte Hard disk
- HP 248x Integrated 270 Kbyte Microfloppy
- HP 248x Integrated 630 Kbyte Microfloppy
- HP 248x Integrated 15 Mbyte Hard disk
- HP 248x Integrated 20 Mbyte Hard disk

ID*27 is the software interface between the operating system and the disk interface card. ID*27 processes I/O instructions for the interface card.

ID*27 Read/Write Request

The call sequences for the ID*27 read and write requests are:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)
Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)

Control Word *cntwd*

Figure 2-48 shows the format of the control word (*cntwd*) for a read or write request to ID*27.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		UE	Z	77B						LU					

Figure 2-48. ID*27 Read/Write Request *cntwd* Format

The UE (User Error) and LU (Logical Unit identifier) bits are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

Bit 12, the Z-bit, must be set to zero because *track* and *sector* are integer variables, not control buffer descriptions.

Bits 6 through 11 must be set to 77B for a read or write request to a disk, or EXEC will reject the request and issue an IO01 error message (not enough parameters in the I/O call). If bits 6 through 11 are set to 77B for a read or write request to any other type of device, EXEC will reject the request and issue an IO01 error message.

bufr and *bufln*

The EXEC request I/O buffer is described by the parameters *bufr* and *bufln*, which are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

track and *sector* Parameters

The *track* and *sector* parameters identify the track and logical sector positions to which data is written, or from which it is read. *sector* must be an even integer value for all read or write

requests. The legal values of *track* and *sector* are shown in Table 2-11. The values in the table represent the full range for each disk. In most systems, disk space is divided between several LUs, so the legal track and sector values for each LU is a subset of the full ranges shown in the table.

Table 2-11. Legal Track and Sector Values

	10 Mbyte Hard disk	270 Kbyte Microfloppy	15 Mbyte Hard disk	20 Mbyte Hard disk
Track	0 - 1219	0 - 65	0 - 1835	0 - 2448
Sector	0 - 30	0 - 15	0 - 30	0 - 30

The physical sector value is a fixed number for a disk that describes how many 256-byte physical sectors, or blocks, can be stored on the disk. Each physical sector is twice as long as the logical sector (128 bytes) used in program calls.

A- and B-Register Contents

The A-Register contains word 6 of the device table after completion of each EXEC read or write request to a disk. The format of word 6 is the same as the format of *stat1*, described in the ID*27 Status Request section of this chapter.

The B-Register contains the positive number of words or negative number of characters that were transmitted during the last read/write request.

Control Word Read Request Example

The following FORTRAN program reads data from track 100, sector 32 of an HP 248x disk at LU 12, into a buffer variable called BUFR.

```

FTN7X Program Read
      IMPLICIT INTEGER (A-Z)
      DIMENSION BUFR (128)
      :
      BUFLN = 128
      TRACK = 100
      SECTOR = 32
      CNTWD = 12 + 7700B
      CALL EXEC(1, CNTWD, BUFR, BUFLN, TRACK, SECTOR)
      end

```

ID*27 Control Request

ID*27 does not support control requests for disk devices. The driver reports an IORQ error and aborts the offending program if it receives a control request.

ID*27 Status Request

The status request call sequence is:

```
CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]])
```

Control Word *cntwd*

Figure 2-49 shows the control word (*cntwd*) format for an EXEC status request to ID*27. The driver returns the status of the last disk accessed.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-49. ID*27 Status Request *cntwd* Format

The Z-bit is described below in the *stat3* and *stat4* Status Parameters section.

stat1 and *stat2* Status Parameters

The returned status parameters *stat1* and *stat2* are shown in Figure 2-50.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Driver Type = 30B				0		EM	SE	0		E			
<i>stat2</i>	AV		Interface Type = 27B				0		I/O Select Code							

Figure 2-50. ID*27 Status Request Parameters

AV (device availability) and **Driver Type** of *stat1* are defined in the Status Request Parameter Definitions section of Chapter 1.

The status code returned to *stat1* bits 0 through 7 is:

- EM End-of-Medium. More sectors than are available were requested.
- SE Soft Error. One or two retries were necessary to successfully complete the request.
- E Error exists. Driver has reported an error to the system via word 16 of the device table (DVT16). Refer to ID*27 Extended Status for a description of DVT16.

stat2 is the interface type (27B) of the disk interface card. The **AV** and **I/O Select Code** are defined in the Status Parameter Definitions section of Chapter 1.

stat3 and *stat4* Status Parameters

If the Z-bit is zero, *stat3* and *stat4* return the first two words of the driver parameter area.

If the Z-bit is 1, *stat3* is the name of a buffer to receive the driver parameters, and *stat4* is the length of the buffer.

Driver Parameter Area

The driver parameters of ID*27 are listed in Table 2-12.

Table 2-12. ID*27 Driver Parameters

Parameter	Parameter Description
+1	Drive Address
+2	Unit Number
+3	Starting Head Number
+4	Starting Cylinder
+5	Reserved
+6	Number of Tracks
+7	Physical Sectors/Track
+8	Number of Surfaces

ID*27 Extended Status

The extended status is read by calling RMPAR after an EXEC read/write request. The returned status words are shown in Figure 2-51.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	X		0						Error Code							
DVT17	Transmission Log = 0															
DVT18	Extended Status #1															
DVT19	Extended Status #2															

Figure 2-51. ID*27 Status Parameters

Bits 14 and 15 of DVT16 are used by the operating system. Bits 0 through 5 of DVT16 contain error codes, interpreted as explained in Table 2-13.

Table 2-13. Error Codes from DVT16, Bits 0 to 5

Error Code	Explanation
00	Normal request completion
01	Illegal request, such as a track number error
*02	Device not ready
*03	Device has timed out
*05	Transmission error
*06	Device write-protected
*12B	Device fault (hardware error)
*77B	Media failure. The disk controller status is stored in DVT18. (See Table 3-6.)
*Codes not reported unless the UE bit of <i>cntwd</i> is set.	

DVT17 (*xstat*(2)) is the transmission log, always zero.

DVT18 (*xstat*(3)) describes the error in DVT16. When there is no error, DVT16 is set to zero. If an error occurs, DVT is set to the error number, and, if the device can report its status, the status is stored in DVT18. Figure 2-52 shows the format of DVT18.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Status				Unit # of Disk							

Figure 2-52. DVT18 Format for ID*27

The reserved bits are always zero.

The status information in bits 8 through 12 is described in Table 2-14.

Table 2-14. Status Codes from DVT18 (Bits 8 through 12)

Status Code	Explanation
0	Normal completion of request. No error occurred in processing this request.
3	Illegal drive type. An unknown drive type has been set in the drive select register on the controller.
10B	Unrecognizable Data error. A disk read operation was terminated because of a data error, such as a bad block or a CRC/ECC error.
11B	Head/Sector compare error. The controller could not find an ID field or data address mark on the requested track. This error could be a hardware or media failure. The driver will retry the request.
23B	Hardware error. The controller has detected an error within the disk drive. Read the extended status from DVT19 to find the cause.

DVT19 contains additional device status information. As with DVT16 and DVT18, DVT19 is normally set to zero. When an error occurs and the device can report its status, DVT18 and DVT19 contain the status. The format of DVT19 is shown in Figure 2-53.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I	Reserved						0	WP	0	FLT	0	SE	DNR	0	

Figure 2-53. DVT19 Format for ID*27

- I:** Indicates that one or more of bits (0, 1, 2, or 4) are set.
- WP:** Indicates that the floppy disk is write protected.
- FLT:** Indicates a hardware fault. The controller was busy or hung, a write fault occurred, track zero was not found, or a handshake timeout occurred on a command transfer.
- SE:** Indicates that the cylinder, head, or sector value is out of bounds, or a non-existent track was requested. The driver will retry the request.
- DNR:** Indicates that the drive is not ready. For the microfloppy, the drive is not connected or no disk is inserted in the drive. For the hard disk, the drive is not up to speed.

ID*27 Error Information

For some errors, the driver makes three retry attempts if the first attempt fails. If all three retries are unsuccessful, the driver issues an error message, and the transmission log is set to zero. On all errors except a *TRK error, the disk LU is set down after the unsuccessful attempt or third unsuccessful retry. The error messages issued by the system for failed requests are described in this section.

I/O - RQ @ LUxx

The request for LUxx is illegal for one of the following reasons:

1. The *track* or *sector* value was negative or too large.
2. The *sector* value was odd.
3. A write request required more sectors than are available.

I/O - NR @ LUxx,D

LU xx is not ready for one of the following reasons:

1. There is no disk in the drive.
2. The disk drive is not connected.

I/O - TO @ LUxx

The driver has timed out on a request because:

1. The disk controller hung.
2. The disk malfunctioned or was not connected.

I/O - TE @ LUxx

Transmission error, caused by a DMA parity.

I/O - WP @ LUxx

The disk is write-protected.

I/O - FA @ LUxx

Indicates a controller of configuration problem, caused by one of the following conditions:

1. Configuration switches indicate that no drive is connected to this unit or address.
2. A timeout occurred on a controller handshake.
3. The controller was busy at the wrong time.
4. The drive reported a write fault or a track zero error.

***TRK xx LU xx STAT=zz**

The storage medium failed. The disk drive status is stored in DVT18, and is reported in this message as zz. The disk is not set down as a result of this error, but the request is flushed.

Interface Driver IDQ35

Users should not call the SCSI interface driver directly. By the time the device driver calls the interface driver, the device driver has transformed the DVT to the following format for normal read and write requests (Z-Buffer requests are passed through unchanged):

Device Driver DVT Transformation															
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0															
DVT 15	Type	UE	Z	Subfunction						x	x	L	UD	Req *	
DVT 16	0	Request buffer address													
DVT 17	Request length														
DVT 18	High-order word of block address for the transfer														
DVT 19	Low-order word of block address for the transfer														
DVT 20	I	d	S	d	d	1	d	d	d	d	d				

* Request Code

d = Driver-dependent bit. This bit depends on the device driver (see the descriptions of DVT word 20 for each of the device drivers in this chapter to obtain more information).

I = Initial Flag Set by generator, cleared on first entry to driver.

S = Serial Bit If set (= 1), this bit indicates that the device is a Serial device. If this bit is not set (= 0), the device is a block device.

Bit 10 of DVT 20 is set to 1 on all calls to the SCSI bus. This bit is 0 for calls from DDC00.

The interface driver then performs the requested I/O to the interface card. At the completion of the request, the status is stored in the DVT in the following format:

DVT Status:															
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0															
DVT 6	AV	Driver type = 30B						Driver status							
DVT 15	Type	UE	Z	Subfunction						x	x	L	UD	Req *	
DVT 16	Error code														
DVT 17	Transmission log														
DVT 18	Extended status #1 Transaction status/SCSI status														
DVT 19	Extended status #2 SCSI sense key/Additional sense code														
DVT 20	I	d	S	d	d	1	d	d	d	d	d				

* Request Code

d = Driver-dependent bit. This bit depends on the device driver (see the descriptions of DVT word 20 for each of the device drivers in this chapter to obtain more information).

I = Initial Flag Set by generator, cleared on first entry to driver.

S = Serial Bit If set (= 1), this bit indicates that the device is a Serial device. If this bit is not set (= 0), the device is a block device.

Driver Error (DVT Word 16)

DVT word 16, bits 5 through 0 (zero) contain error codes for the DDQ35 device driver. The error codes are as follows:

<u>Dec.</u>	<u>Octal</u>	<u>Error and Solution</u>
1	1B	= Driver rejected call. Illegal subfunction bits.
12	14B	= Generation error. The number of IFT extension words was not sufficient.
28	34B	= Driver defined error. Protocol failure; the SCSI card returned a response that the driver was not expecting.
40	50B	= Driver defined error. TermPwr switch is not enabled, the fuse on the card is blown, or the cable is not attached.
41	51B	= Driver defined error. A call to the interface driver bypassing the device driver was the first call seen; this is not allowed as the device driver setup for the LU has not been done.
42	52B	= Driver defined error. The card SCSI address is the same as the device SCSI address.
43	53B	= Driver defined error. Select error; incorrect SCSI address. No device on the bus responded to the SCSI address that we got from DVP 1.
44	54B	= Driver defined error. Incorrect length for CDB in Z-buffer call, or illegal CDB type (per SCSI-2 standard).
45	55B	= Driver defined error. Firmware revision code not as expected (driver or firmware is too old).
62	76B	= SCSI error. DVT18 contains the SCSI error code.

PROM Storage Module Driver ID*36

ID*36 drives the HP 12008A PROM storage module. The PROM storage module uses track and sector addressing for compatibility with the existing system, to permit program loads and data access. ID*36 is accessed via EXEC requests, and all I/O transfers are under card DMA control.

ID*36 cannot write to a PROM. Write requests are ignored, and no error messages are generated by write requests, to ensure PROM compatibility with FMP.

ID*36 Read/Write Request

The call sequences for the read and write requests are:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)
Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)

Control Word *cntwd*

The control word (*cntwd*) format is shown in Figure 2-54.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	UE	0	77B						LU					

Figure 2-54. Read/Write Control Word for ID*36

The UE bit is the user error bit which allows the user program to process device errors. The LU bits contain the LU of the PROM device.

Bits 6 through 11 must be set for any read or write request to a disk-type device or EXEC will reject the request and issue an IO01 error message.

bufr and *bufln*

The *bufr* parameter is the user buffer address for input or output data. The *bufln* parameter is the length of the data record to be transferred. A positive value indicates length in words; a negative value indicates length in bytes.

track and *sector*

The *track* and *sector* parameters specify the address in PROM memory from which the data is read. The driver uses these track and sector addresses, the number of blocks per track on the device, and the total number of tracks for error checking to calculate the corresponding PROM module address. Note that, just as for a disk, *sector* must be an even integer value for all read or write requests.

A- and B-Register Contents

After each EXEC read/write request, the A-Register will contain word 6 of the device table. The format of word 6 is the same format as *stat1* returns after an EXEC 13 extended status request (described later in this section).

After an EXEC read/write request, the B-Register contains either the positive number of words or characters transmitted during the last request, depending on the type of request. If the request specified positive words for the length, the B-Register contains the positive words read or written; if the request specified negative characters, the B-Register contains the positive characters read or written.

Example

The following FORTRAN program segment reads from track 20 sector 4 of a PROM module, LU 17. The data is read into the buffer called BUFR.

```

IMPLICIT INTEGER(A-Z)
DIM BUFR (128)
BUFLN = 128
TRACK = 20
SECTOR = 4
CNTWD = 17 + 7700B
CALL EXEC (1, CNTWD, BUFR, BUFLN, TRACK, SECTOR)

```

ID*36 Control Requests

Unlike many peripheral I/O devices that can be controlled through EXEC requests, there are no control requests available to the PROM modules via the ID*36 interface driver.

ID*36 Status Reporting

The call sequence for the ID*36 status request is:

```
CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])
```

Control Word *cntwd*

The format of the control word (*cntwd*) is shown in Figure 2-55.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-55. Status Request Control Word for ID*36

stat1 and *stat2* Status Parameters

Figure 2-56 shows the format of status parameters *stat1* and *stat2*.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Driver Type = 36B						0						E	
<i>stat2</i>	AV		Interface Type = 36B						0		I/O Select Code					

Figure 2-56. *stat1* and *stat2* Status Words Format for ID*36

In *stat1*, the AV bits (device availability) are defined as for all other drivers. The E-bit indicates an error; it is set by the system if the driver reports any error in DVT16.

stat3 and *stat4* Status Parameters

If the Z-bit in *cntwd* is 0, *stat3* returns the first word of the driver parameter area and *stat4* returns the second word.

If the Z-bit is 1, *stat3* names the buffer (or address) into which the driver parameter area is returned. *stat4* is the length of *stat3*.

ID*36 Extended Status

In addition to the status information returned by the EXEC 13 request, extended device status is also available. The extended status is read by a call to RMPAR immediately after a nonbuffered read or write request.

At the completion of a nonbuffered I/O request, DVT16 through DVT19 are stored in words 2 through 5 of the calling program ID segment. RMPAR retrieves those words and returns them to a user buffer.

Because words 2 through 5 of the ID segment are used as temporary storage, the information stored there is valid only as long as the calling program does not use the temporary storage area. Most EXEC calls use the temporary storage, so RMPAR must be called immediately after an I/O request to guarantee good status information.

The extended status information returned in the 5-word RMPAR array (called *xstat*) is shown in Figure 2-57.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>xstat</i> (1) / DVT16	x	x	0						Error Code							
<i>xstat</i> (2) / DVT17	Transmission Log															
<i>xstat</i> (3) / DVT18	Card Status															
<i>xstat</i> (4) / DVT19	Undefined															
<i>xstat</i> (5)	Undefined															

Figure 2-57. DVT16 to DVT19 Extended Status Words Format for ID*36

The 6-bit Error Code describes the device error, if any. If any of the Error Code bits are set, bit 0 of the device status word is also set. The following error codes can be returned:

- 0 No Error
- 1 Illegal Request
- 3 Timeout
- 77B Media Error

Transmission Log is a value returned by the driver indicating the positive number of words or characters transferred in a nonbuffered read, write, or control request. The value is either the positive number of words or the positive number of characters transferred, depending on the original EXEC request.

Card Status is a value returned by the driver, indicating the status of the card. It is picked up from register 31B on the PROM storage module (refer to the *HP 12008A PROM Storage Module Reference Manual*).

Driver Parameter Area

The below listed values are returned in the driver parameter area for ID*36. Where the parameter has no significance for a PROM module (for example, number of surfaces, number of spare tracks) the entry should be set to zero.

Table 2-15. ID*36 Driver Parameter Area

Parameter	Parameter Description
+1	0
+2	0
+3	0
+4	0
+5	0
+6	0
+7	0
+8	Number of Tracks (64)
	Number of Blocks/Track (4)
	Blocking Factor (1)

ID*36 Error Information

For unsuccessful data transmissions, the system returns the following error message:

I/O - RQ @ LUXx

This error can result if you specify a track or sector value as an odd number or a negative number, if you specify a sector or track number greater than the maximum allowed value, or if you attempt to write more sectors than are available on the PROM.

HP-IB Interface Card Driver ID*37

The HP-IB Interface Card Driver ID*37 controls the HP 12009A HP-IB Card. The HP-IB port is a primary I/O path between the CPU and I/O devices that range from simple switches to high-speed disks. Other than sharing the same bus wires and some programming conventions, bus devices vary in programming complexity, data rates, and interpretation of the HP-IB protocol.

Although ID*37 performs overall control of basic HP-IB functions, it does not attempt to resolve the contention problems that may arise due to the multiprogramming functions of the system. Continuity in such operations must be maintained within the application program.

The driver will resolve contention problems arising from the interleaving of data transfers through programs when more than one program is operating the HP-IB. If this type of bus activity is anticipated, all logical unit access should be controlled through the LU Lock feature, or bus activity should be synchronized with resource numbers. (Refer to the *RTE-A Programmer's Reference Manual* for more information on LU locking and resource numbers.) Refer also to the *HP 12009 HP-IB Interface Card Reference Manual* for details of the hardware functions.

The following mnemonics are used in this section to describe functions, commands, control line descriptions, and data line descriptions used by the HP-IB and ID*37. The first eight mnemonics define the eight data management lines of the bus; DIO defines the eight data lines; CR, LF, and EOR are special characters that are part of the data.

Table 2-16. ID*37 and HP-IB Mnemonics

Mnemonic	Meaning	Mnemonic	Meaning
DAV	Data Valid	REN	Remote Enable
NRFD	Not Ready For Data	EOI	End or Identify
NDAC	Not Data Accepted	DIO	Data Input/Output
IFC	Interface Clear	CR	Carriage Return
ATN	Attention	LF	Line Feed
SRQ	Service Request	EOR	End of Record

ID*37 Operating Modes

There are two modes of operation supported by ID*37: Auto-Addressing and Direct I/O. The desired mode is selected by specifying a particular logical unit number. For Auto-Addressing, the LU of the device should be supplied in an I/O request, and the driver constructs the proper device address. For Direct I/O, the LU of the bus is supplied and the user is responsible for addressing each required device.

Auto-Addressing

In auto-addressing mode, ID*37 assumes complete control of the HP-IB bus. Using the LU number associated with a particular device, you may read and write data as well as send device-control information. All addressing of the device LU is taken care of by ID*37. With auto-addressing, you are assured that the program will not issue unaddressed data or control information which could adversely effect other devices on the HP-IB bus.

In auto-addressing mode, ID*37 will first “unaddress” all non-participating devices by issuing an UNTALK (UNT) and then an UNLISTEN (UNL) to all devices on the bus. It then will address the device LU specified in the control word of the EXEC request with either a TALK (MTA) for a read request, or a LISTEN (MLA) for a write request. (A secondary address may be supplied in PRAM3.) The activity for auto-addressing is

Read: UNT , UNL , MLA (*controller*) , MTA , [*secondary address*] , *data* , *terminator*
Write: UNT , UNL , MTA (*controller*) , MLA , [*secondary address*] , *data* , *terminator*

The driver then initiates the data transfer according to the format specified in the subfunction field of the control word, *cntwd*.

Direct I/O

When using direct I/O to the bus, you are responsible for all command and addressing requirements. This mode is usually used to address multiple listeners or to provide universal or selected device control commands.

Note that when using the auto-addressing mode, the driver assumes complete control of the HP-IB, while in the direct I/O mode of operation, ID*37 provides none. As such, any mix of the two modes in a single application must be approached with caution. In this situation, the overall activity of bus control lines must be considered, with the responsibility for continuity of operation resting with the user.

Figure 2-58 presents a schematic comparison of the two operating modes.

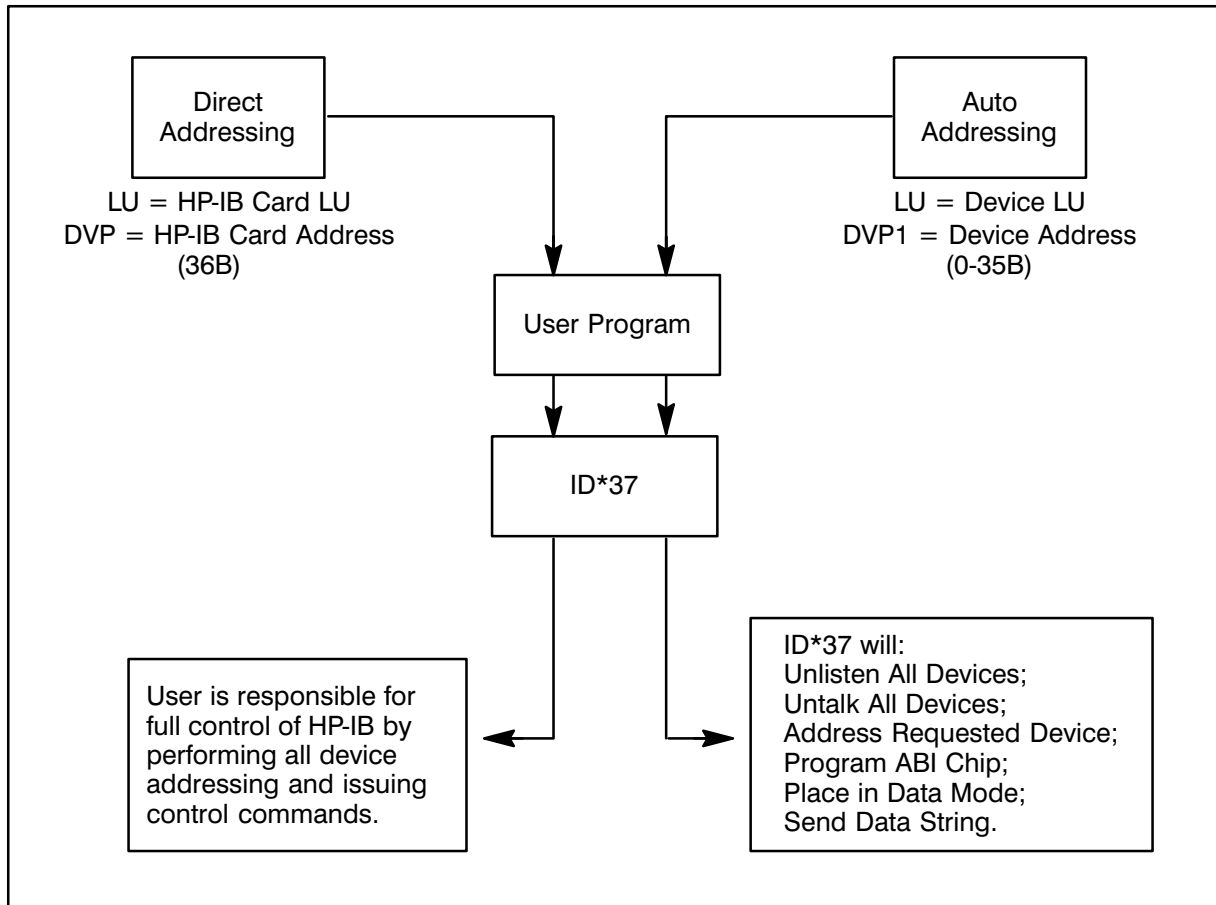


Figure 2-58. Direct-Addressing/Auto-Addressing Mode Comparison

ID*37 End-Of-Record Processing

For ASCII data, the EOR is signalled by a carriage return and line feed (CRLF). For binary data, the EOR is indicated by asserting the EOI bus control line. For data transfers of more than 256 bytes, LF is not a logical terminator; the PHI/ABI chip terminates transfers by word count or EOI only.

For write requests, the subfunctions have the following meaning on EOR:

- 00 ASCII Data Record Format (Normal ASCII). The data transfer is terminated by a CRLF supplied by the driver, unless the last character is an underscore (ASCII “_”, 137B), which suppresses the CRLF. The LF causes the EOI line to be set.
- 01 Fixed Length Binary Record Format (Normal Binary). EOR is supplied by the driver with the last byte of data.
- 20B Transparent ASCII Format. No EOR signal is supplied by the driver. The data transfer is terminated by an LF in the user data buffer.
- 21B Transparent Binary Format. EOR signals are not supplied by the driver, so the EOI line is never asserted. The transfer is terminated by an LF in the user data buffer.

For read requests, the subfunctions have the following meaning on EOR:

Note If the word count in *bufln* is reached, or if the device sets EOI, the driver terminates the transfer for every read request format except transparent binary.

- 00 ASCII Data Record Format (Normal ASCII). Data transfers are terminated by an LF in the data or by the device setting the EOI line. LF does not terminate large buffers (>256 bytes); EOI must be used to terminate the transfer. If a CRLF or LF is used to terminate the data, they are not included in the transmission log count or in the returned data buffer. This is the mode used by most HP-IB devices for data termination.
- 01 Fixed Binary Record Format (Normal Binary). The received data is terminated by an EOI signal. As stated above, LF does not terminate large buffers (>256 bytes). Such buffers are terminated by word count or EOI only.
- 20B Transparent ASCII Format. For this format, the user will usually receive data as needed, and supply an LF when the read request is to be terminated.
- 21B Transparent Binary Format. No EOR signalling from the device will be responded to by the driver.

ID*37 Auto-Addressed Read/Write Requests

The call sequences for the read and write requests are:

READ Request: CALL EXEC(1 , *cntwd* , *bufr* , *bufln* [, *pram3*])
 WRITE Request: CALL EXEC(2 , *cntwd* , *bufr* , *bufln* [, *pram3*])

Control Word *cntwd*

Figure 2-59 shows the format of the read/write request control word in auto-addressed mode.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	Subfunction						Device LU					

Figure 2-59. *cntwd* Format for an Auto-Addressed Read/Write

The BB (driver bypass), NB (Nonbuffered mode), and UE (User Error) bits are as defined in Chapter 1, Read/Write Parameter Definitions. Bit 12 (Z-bit) must be set to zero because a control buffer is not allowed in the auto-addressing mode. If the Z-bit is set to 1, the request is terminated.

The Subfunction field defines the type of data to be transferred and specifies end-of-record processing. Refer to the End-of-Record Processing section in this chapter for a description of this field.

bufr and *bufln*

The I/O buffer used in the EXEC request is described by parameters *bufr* and *bufln*. *bufr* is the address of the I/O buffer, and *bufln* is the length of the buffer defined as the positive number of words or the negative number of characters in the buffer.

Note The buffer returned by ID*37 is valid only to the number of words or characters specified in the transmission log (returned in the B-Register).

pram3 (Secondary Address)

Some devices have subchannels within them that are accessed with a secondary address. Secondary addresses specify the subchannel in the device to be accessed. The secondary address is a decimal value between 0 and 31. If *pram3* is zero, however, the driver assumes that there is no secondary address. An actual secondary address of 0 must be specified as 400B, using bit 8 to indicate that this parameter has a value.

ID*37 Auto-Addressed Control Requests

The call sequence for the control request is:

```
CALL EXEC( 3 ,cntwd[ ,pram1[ ,pram2[ ,pram3[ ,pram4 ] ] ] ] )
```

Note For high-speed devices, special cable-length restrictions apply. Refer to the *HP 12009A HP-IB Interface Reference Manual* for a description of the high-speed data cable restrictions.

Control requests control devices or device drivers. Auto-addressed control requests program devices directly, so the programmer must make sure that a device can respond to control requests directed to it. Figure 2-60 shows the control word (*cntwd*) format used by ID*37 for an auto-addressed control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	Subfunction						Device LU					

Figure 2-60. *cntwd* Format for an Auto-Addressed Control Request

The BB (driver bypass), NB (Nonbuffered mode), and UE (user error) bits are as defined in Chapter 1. Bit 12, the Z-bit, must be set to zero since no control buffer can be sent in auto-addressed mode. The function codes are:

- 00 Selected Device Clear (SDC).
- 06 Device Dynamic Status (Serial Poll).
- 16B Set REN True.
- 17B Go To Local (GTL) .
- 20B Enable SRQ Program Scheduling.
- 21B Disable SRQ Program Scheduling.
- 22B Set Interface Driver Timeout.
- 23B Enable Parallel Poll Interrupts.
- 24B Set Device Address.
- 27B Group Execute Trigger (GET).
- 30B Disable code 20B from allowing SRQ interrupts.
- 31B Restore ability of code 20B to allow SRQ interrupts.
- 40B Enable Parallel Poll Program Scheduling (PPE).
- 41B Disable Parallel Poll Program Scheduling (PPD).

Function Code 00: Selected Device Clear (SDC)

The Selected Device Clear causes the addressed device to reset to a predetermined state. The device is addressed, an SDC command is issued, and the device resets itself. Refer to the device reference manual for a description of its clear state.

Command Sequence: UNT, UNL, TLK, LSN, SDC

Function Code 06B: Device Dynamic Status

The dynamic status request reads the device status into words 16 through 18 of the device table (DVT16 - DVT18). The program must make a call to RMPAR to retrieve the status information. The format of the returned status information is shown in Figure 2-61.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16												Error Codes				
DVT17	Transmission Log=0															
DVT18	HP-IB Device Status															

Figure 2-61. DVT16 to DVT18 Extended Status Words Format

DVT16: Bits 14 and 15 are used by the system. The decimal error codes returned to bits 0 through 5 are:

- 00 - No error.
- 01 - Illegal request.
- 03 - Device timeout.
- 20 - Insufficient space in IFT extension for poll table entry. Request is flushed.

DVT17 returns the transmission log (always 0).

DVT18 is the device status byte.

Most HP-IB devices return eight bits of status information in response to a serial poll by ID*37. An SPE command sequence is issued and the addressed device returns its status byte into DVT18; status can be read by a call to RMPAR. If the device is in the SRQ table (see function code 20B), and has performed an SRQ, then the status at the time of the SRQ is returned. Bit 6 of the status byte (DVT18) is set to 1 to indicate that the device has performed an SRQ. Bit 6 is normally reset to 0 after each I/O request. The driver terminates this request by issuing a serial poll disable.

Command Sequence: UNT, UNL, SPE, TLK, LSN, STATUS, SPD

Function Code 16B: Set REN True

This function sets the REN bus line true, causing addressed devices to enter the remote mode. Some devices require REN to be true before they can be programmed over the HP-IB. Refer to the device user's manuals for further information.

Command Sequence: REN, UNT, UNL, TLK, LSN

Function Code 17B: Go to Local (GTL)

The GTL command is issued, allowing the addressed device to go to local control mode.

Command Sequence: UNT, UNL, TLK, LSN, GTL

Function Code 20B: SRQ Program Scheduling

This request causes a user alarm program, defined in *pram1* through *pram3*, to be scheduled in response to a device service request. For example; if the program ALARM is to be scheduled, *pram1* will contain ASCII AL, *pram2* will contain ASCII AR, and *pram3* will contain ASCII M. Note that the program name should be 5 characters or less; trailing blanks will fill to the required five-character length.

pram4 may contain a single valued integer to be passed to the alarm program, which can be retrieved in the alarm program via a call to RMPAR. The first three values picked up by a call to RMPAR are:

1. Interrupting LU
2. Passed parameter (*pram4*)
3. Status

SRQ interrupts will be disabled on the bus unless at least one device has issued this control request. It is the user's responsibility to ensure that if any device is able to generate an SRQ interrupt, either synchronously or asynchronously, it must have an alarm program defined by this control request. Failure to do so will result in the system error message "ILLEGAL INTERRUPT" if the device asserts the SRQ line. All SRQ interrupts also will be disabled, since there will be no way for the driver to clear the SRQ without knowing which device to poll.

Each device that uses SRQ interrupts needs four words in the interface extension area. The default extension size of 134 words includes space for one device; therefore, if more than one HP-IB device will use SRQ interrupts, the default extension size must be increased by four words for each additional device. For example, the following statement allows two devices to use serial polling:

```
IFT,%ID*37::rte_a,SC:25B,TX:138
```

Note In an RTE-A session environment the program to be scheduled on interrupt must be RP'd in the system session.

The alarm program must terminate saving resources. This is accomplished by a call to EXEC 6, as described in the *RTE-A Programmer's Reference Manual*. If the program does not terminate saving resources, it will only be able to service one SRQ. The second SRQ will fail to schedule the alarm program, and the following error will be reported:

```
POLL ERR: NO PROG LUxx
```

where *xx* is the LU number of the device that asserted the SRQ line.

Function Code 21B: Disable SRQ Program Scheduling

The addressed device will be removed from the driver's list of devices to poll upon an SRQ interrupt. If the list is left empty by this request, all SRQ interrupts will be disabled.

Function Code 22B: Set Interface Driver Timeout

This request will set the current timeout value for ID*37 to a new value contained in *pram1* (specified in tens of milliseconds). For example, to set the timeout to 100 seconds, *pram1* should contain 10000.

Function Code 23B: Parallel Poll Assignment

When *pram1* = 0, *pram2* identifies a DIO line to be assigned to the addressed device for parallel polling response. When *pram1* = 1, parallel polling is disabled.

If *pram2* is negative, the device requests service by setting the line to 0. If *pram2* is positive, it requests service by setting the line to 1.

If a parallel poll program scheduling control request has been made (function code 40B) for the addressed device, the defined program will be scheduled when the DIO line is asserted in response to a parallel poll. If the parallel poll is busy, one scheduling retry will be attempted after a delay of 500 milliseconds.

Command Sequence: UNT, UNL, TLK, LSN, PPC, PPE

Function Code 24B: Set Device Address

This request allows you to programmatically set the address for a device. The device address (0 through 35B) is defined by *pram1*, and will be placed in the first word of the driver parameter area.

Function Code 27B: Group Execute Trigger (GET)

This request causes the addressed device to initiate a pre-programmed action (for example, triggering a DVM to make a voltage reading as set up by a previous request).

Command Sequence: UNT, UNL, TLK, LSN, GET

Function Code 30B: Disable SRQ Interrupts

Function code 30B disables the ability of code 20B requests to turn on SRQ processing. A function code 31B request restores the ability of code 20B requests to turn on SRQ processing. If a program executes a number of code 20B control requests to establish SRQ processing for a number of bus devices, a code 30B request should be executed first to prevent an SRQ interrupt from occurring while the program completes all of the code 20B requests.

The HP-IB disk device driver does not permit control requests. Code 30B and 31B requests should be sent to non-disk HP-IB LUs. If there are no non-disk HP-IB LUs, set the BB (driver bypass) bit in *cntwd* of a function code 30/31B request so the device driver does not see the control request.

In a program with five function code 20B requests, the sequence of code 30B, code 20B, and code 31B requests looks like this:

Code	Result
30B	Code 20 requests cannot enable SRQ interrupts
20B	SRQ processing established for first device
20B	SRQ processing established for second device
20B	SRQ processing established for third device
20B	SRQ processing established for fourth device
31B	Code 20B requests can enable SRQ interrupts
20B	SRQ processing established for first device and SRQ interrupts enabled.

It is important that the last function code 20B request follow the function code 31B request, so that SRQ interrupts can be enabled. The function code 31B request does not enable SRQ interrupts, it only restores the ability of code 20B requests to enable them.

Function Code 31B: Restore SRQ Interrupts

After a function code 30B request has been executed, function code 20B requests do not enable SRQ interrupts. A function code 31B request restores the ability of function code 20B requests to enable them. Function codes 30B and 31B are used as a pair around a group of function code 20B requests, to prevent SRQ interrupts from occurring until a program finishes executing the group of function code 20B requests, as in the sequence above. Again, it is important that the last function code 20B request follow the function code 31B request, to enable SRQ interrupts. The function code 31B request does not enable them.

Function Code 40B: Enable Parallel Poll Program Scheduling

Any time a parallel poll line is assigned to the addressed device interrupts, a program described by *pram1* through *pram3* will be scheduled (see function code 20B for an example of passing the program name in *pram1* through *pram3*). *pram4* is an optional single valued integer to be passed to the scheduled program upon interrupt. When picked up by the scheduled program with a call to RMPAR, *pram4* will be the second RMPAR parameter, while the LU of the device causing the interrupt will be the first parameter. The third RMPAR parameter is the status.

Note In an RTE-A session environment the program to be scheduled on interrupt must be RP'd in the system session.

Function Code 41B: Disable Parallel Poll Program Scheduling

This request will clear the ID*37 internal poll table, disabling all interrupts from any assigned poll line for the device.

ID*37 Direct I/O Read/Write Requests

Note To make best use of the direct I/O calls described below, the control buffer must contain ABI programming information. ABI programming is explained in the *HP 12009 HP-IB Interface Card Reference Manual*.

The call sequences for the direct I/O read/write requests are:

Read Request: CALL EXEC(1, cntwd, bufr, bufln[,pram3[,pram4]])
Write Request: CALL EXEC(2, cntwd, bufr, bufln[,pram3[,pram4]])

Direct I/O gives direct access to the bus, permitting special bus control and device access. All addresses and commands must be generated by the controlling program. *pram3* and *pram4* are used to pass commands and addresses, as described in the following paragraphs.

In direct I/O, a control buffer is established to pass device addresses and commands. If there are commands in the command buffer (that is, $Z = 1$), they are placed on the bus lines in the Command Mode (Attention Line Enabled). If *pram4* is positive, one command is transferred per word. If *pram4* is negative, each byte contains a command. If there is data in the data buffer, it is placed on the bus lines in the data mode (Attention Line Disabled). The driver passes the commands and data exactly as they appear in the command and data buffers. Note that when *pram3* and *pram4* are used to pass commands, *bufln* must be set to 0.

For a double-buffered request (that is, $Z = 1$) the driver processes all command buffer data placed on the data lines (DIO lines) as either device addresses or as universal commands. If *pram4* is positive, data is sent in the word mode. If *pram4* is negative, it is sent in byte mode.

On direct transfer between two or more devices, always make the host computer a listener (read request) to ensure that data transfers occur.

Device Addresses

There are two ASCII address characters for each bus device, one for the talk address, and one for the listen address. When a device transmits data, it becomes a talker. Talk addresses range from 100B (@) to 136B (^). When a device receives data, it becomes a listener. Listen addresses range from 40B (space) to 76B (>). Before information is sent over the bus, one talker and at least one listener must be addressed.

The talker and listeners are addressed by sending ASCII address characters over the bus via the control buffer. Before addressing new listeners and transmitting data, it is good practice to send an UNLISTEN command, 77B (?), to make sure that only the necessary devices are listening. It is also a good practice to issue an UNTALK command to all devices by transmitting an ASCII _ (137B) before establishing the talker.

As an example, to “unlisten” and “untalk” all devices on the bus, set the device with an address of 5 as a talker, and set the devices with addresses 6 and 7 to be listeners:

Command Buffer	Octal	ASCII	Description
Byte 1	77	?	(Unlisten all Devices)
Byte 2	137	_	(Untalk all Devices)
Byte 3	105	Ē	(Talk Address)
Byte 4	46	&	(Listen Address)
Byte 5	47	'	(Listen Address)

Control Word *cntwd*

Figure 2-62 shows the format of the control word (*cntwd*) used by ID*37 for a direct I/O read or write request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	Subfunction						Bus LU					

Figure 2-62. *cntwd* Format for a Direct I/O Read/Write Request

The NB (Nonbuffered mode) and UE (User Error) bits are as defined in Chapter 1. The Bus LU bits describe the LU of the bus to which all data and commands apply. Therefore, to address specific devices on the bus, the control buffer should be used as previously described. Bit 15 (BB-bit) should be set to 1 if a device driver exists for the bus LU and is to be bypassed.

The Subfunction field is used for defining the type of data to be transmitted, and for end-of-record processing. Refer to the End-of-Record processing section for a description of this field.

The Z-bit (bit 12) should be set whenever a control buffer is to be sent to the driver via *pram3* and *pram4*. See the definition of *pram3* and *pram4* for more information on how to set up these parameters for the EXEC request.

*buf*r and *buf*ln

The I/O buffer used in the EXEC request is described by parameters *buf*r and *buf*ln. *buf*r is the address of the I/O data buffer, and *buf*ln is the length of the buffer, defined as either the positive number of words or the negative number of characters in the buffer.

***pram3* and *pram4* Control Buffer Descriptors**

As previously described, a control buffer may be passed to the driver to provide device addressing, multiple device response, and universal device commands. The control buffer containing these commands is described by optional parameters *pram3* and *pram4*. (The Z-bit must be set to indicate to the driver that the control buffer should be used by the driver.) *pram3* will contain the address of the buffer, while *pram4* will contain the length of the buffer in the positive number of words or the negative number of characters. If positive, the control buffer will be sent in word mode; if negative, it will be sent in byte mode with an ATN test.

ID*37 Direct I/O Control Request

The call sequence for the direct I/O control request is:

```
CALL EXEC( 3 ,cntwd[ ,pram1[ ,pram2[ ,pram3[ ,pram4]]]] )
```

Control requests in the direct I/O mode relate to bus activity instead of device activity. That is, a direct I/O control request will affect the entire bus and all addressed devices connected to the bus. It is your responsibility to ensure that the programming requirements are within the capabilities of the addressed devices.

Control Word *cntwd*

Figure 2-63 shows the format of the control word (*cntwd*) used by ID*37 for a direct I/O control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	Subfunction						Bus LU					

Figure 2-63. *cntwd* Format for a Direct I/O Control Request

The NB (Nonbuffered mode) and UE (User Error) bits are defined in Chapter 1 under the Read/Write Parameter Descriptions. The Bus LU bits describe the LU of the bus to which the control request is to be made. Bit 15 (BB-bit) should be set to one if a device driver exists for the device.

The Z-bit (bit 12) should be set when a control buffer is to be sent to the driver via *pram3* and *pram4*. The Z-bit may be set for function codes of 00 (if *pram1* = 0), 16, 17, 23, and 27. The following function codes are available for ID*37:

- 00 Clear and Reset Device
- 06 Dynamic Bus Status
- 16B Set REN True (Remote Enable)
- 17B Go To Local (GTL)
- 23B Parallel Poll Interrupt
- 25B Local Lockout (LLO)
- 27B Group Execute Trigger (GET)
- 40B Enable Parallel Poll Program Scheduling
- 41B Disable Parallel Poll Program Scheduling
- 51B Bail Out (Abort)

Function Code 00: Clear and Reset Device

This request causes either selected devices or the entire bus to be cleared, as determined by *pram1*.

If *pram1* is 0, a selected device clear (SDC) command is sent onto the bus, causing all listening devices to be reset to a predetermined state. Refer to the associated device user's manual for a description of the device reset state. (The Z-buffer may be used.)

If memory size is limited, issue the following two calls:

```
CALL EXEC(3,5100B+IBLU)
CALL EXEC(2,10000B+IBLU,0,0,24B*400B,-1)
```

The EXEC 3 call is an interface clear (IFC) and the EXEC 2 call sends a universal device clear (DVC).

Function Code 06B: Dynamic Bus Status

This request causes the status of the HP-IB interface card to be returned in words 16 through 19 of the device table (DVT16-DVT19). Therefore, after completion of the request, a call to RMPAR will retrieve the status information. The format of the returned status information is shown in Figure 2-64.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	x	x	0						Error Code							
DVT17	Transmission Log = 0															
DVT18	HP-IB Interface Card Status															
DVT19	ABI Interrupting Conditions								ABI Status							

Figure 2-64. DVT16 to DVT19 Extended Status Information

In DVT16, bits 14 and 15 are used by the system. The decimal error codes returned to bits 0 through 5 are:

- 1 Illegal request.
- 3 Device timeout.
- 20 Insufficient space in IFT extension for poll table entry. Request is flushed.

The timeout error results in the driver downing the device and not flushing the request. The interface card is cleared of the request to allow processing of other device requests.

DVT17 is the transmission log (always 0).

DVT18 is a status word. The format of the HP-IB interface card status (upper byte) is defined below. The parallel poll response returned in the lower byte (bits 0-7) reflects the status of the parallel poll DIO lines.

Bit	Meaning if Set
15	Parallel Poll in Effect
14	Group Execute Trigger (GET) Received
13	End-Of-Record (EOI or Line Feed) Received
12	Secondary Address Received
11	PHI Interrupt Request
10	DMA Operation Frozen
9	HP-IB Service Request Pending (SRQ)
8	ABI Requesting Transfer (DMARQ)
7-0	Parallel Poll Status (DIO 7-0)

DVT19 returns the status of the ABI registers. Refer to the *HP-IB User's Reference Manual* for the format of this word.

Function Code 16B: Set REN True

This request asserts the REN bus line, causing any addressed devices to go to remote mode. The control buffer may be used to address the devices and, if not used, any previously addressed devices will be set to remote. Some HP-IB devices require the REN line to be true before they will send or receive information under program control. Refer to the associated device user's manual for further information.

Function Code 17B: Go To Local and Clear Local Lockout (GTL)

When *pram1* is set to 1, the GTL command is sent to allow any addressed devices to go to local control mode. The control buffer may be used to address the devices and, if not used, any previously addressed devices will be set to local mode. If *pram1* is set to 0, the REN line will be set to false to allow any responding device to go to local mode.

Function Code 23B: Parallel Poll Configure

If *pram1* is 0, *pram2* specifies a DIO line to be assigned to the device for parallel polling response, and the control buffer must contain the address of the device to be configured. If *pram2* is negative, the device responds to a parallel poll by setting the DIO line to 0. If *pram2* is positive, it responds by setting the line to 1. The control buffer must contain the address of the device when *pram1* = 0. Once the driver addresses these devices, the parallel poll configure (PPC) and parallel poll enable (PPE) commands will be sent on the bus. If a parallel poll program-scheduling control request has been made (function code 40B) for the bus LU, that program will be scheduled when the assigned DIO line interrupts.

If *pram1* is 1, the DIO line previously assigned to the bus LU is disabled and the parallel poll configure (PPC) and parallel poll disable (PPD) commands are sent on the bus to all the devices addressed by the control buffer (control buffer is required).

If *pram1* is 2, a parallel poll unconfigure (PPU) command is sent on the bus to reset all parallel poll devices.

Function Code 25B: Local Lockout (LLO)

This request causes all recognizing devices on the bus to lock out local operation. As long as the local lockout message is in effect, no device can be returned to front-panel control except by sending the clear lockout message (refer to function code 17B). All devices that recognize the LLO bus command are locked out, whether they have been addressed or not.

Function Code 27B: Group Execute Trigger (GET)

This request causes all addressed devices to initiate a pre-programmed action (for example, triggering a DVM to make a voltage reading as set up by a previous request). The control buffer may be used to address the devices and, if not used, any previously addressed devices will be triggered.

Function Code 40B: Enable Parallel Poll Program Scheduling

Any time a parallel poll line is assigned to an addressed device, an alarm program described by *pram1* through *pram3* is scheduled. For example, if the program ALARM is to be scheduled, *pram1* will contain an ASCII AL, *pram2* will contain an ASCII AR, and *pram3* will contain an ASCII M. The program name should be five characters or less; if less, trailing blanks are used to fill to the five-character length.

pram4 is an optional single valued integer to be passed to the scheduled program and can be picked up by a call to RMPAR. *pram4* will be the second RMPAR parameter, while the LU of the bus causing the interrupt will be the first parameter.

Note

In an RTE-A session environment the program to be scheduled on interrupt must be RP'd in the system session.

Function Code 41B: Disable Parallel Poll Program Scheduling

This request will clear the ID*37 internal poll table, causing interrupts from any assigned poll line to be disabled.

Function Code 51B: Bailout (ABORT)

The interface clear (IFC) line will be set true for 10 milliseconds and then set false.

***pram3* and *pram4* Control Buffer Descriptors**

As previously described, a control buffer may be passed to the driver to provide device addressing, multiple device response, and universal device commands. The control buffer containing these commands is described by optional parameters *pram3* and *pram4*. (The Z-bit must be set to indicate to the driver that the control buffer should be used.) *pram1* will contain the address of the buffer; *pram2* will contain the length of the buffer in the positive number of words or the negative number of characters. If positive, the control buffer will be sent in word mode; if negative it will be sent in byte mode. The attention (ATN) line will always be set.

ID*37 Status Request

The call sequence for the ID*37 status request is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4] ] ] )
```

Control Word *cntwd*

Figure 2-65 shows the control word (*cntwd*) format for an EXEC status request for ID*37.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-65. ID*37 Status Request *cntwd* Format

stat1 and *stat2* Status Parameters

The format of the returned status parameters *stat1* and *stat2* is shown in Figure 2-66.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type=xx				Status					E				
<i>stat2</i>	AV		Interface Type=37				0	0	I/O Select Code							

Figure 2-66. *stat1* and *stat2* Status Words Format

The AV (device availability) and Device Type bits in *stat1* are defined in the Status Request Parameter section in Chapter 1. The status information contained in bits 1 through 7 is defined on a device level; refer to the associated device driver documentation for the error codes returned. The E-bit (bit 0) is set if any error is returned in DVT18.

In *stat2*, the interface type is 37B, corresponding to the HP-IB interface card. All other bits are as defined in Chapter 1.

stat3 and *stat4* Status Parameters

If the Z-bit in *cntwd* is 0, *stat3* returns the first word of the driver parameter area and *stat4* returns the second word.

If the Z-bit is 1, *stat3* names the buffer (or address) to return the driver parameter area into, and *stat4* is the length of *stat3*.

Driver Parameter Area

Any words returned from the driver parameter area to *stat3/stat4*, or to the status buffer described by *stat3/stat4*, are device-dependent. Refer to the associated device driver documentation for the definition of the returned information.

ID*37 Extended Status

As discussed in Chapter 1, extended status can be obtained by making a call to RMPAR after an EXEC request to determine the status of the request. The format of the returned status is shown in Figure 2-67.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	x		0						Error Code							
DVT17	Transmission Log = 0															
DVT18	HP-IB Interface Card Status															
DVT19	ABI Interrupting Conditions								ABI Status							

Figure 2-67. DVT16 to DVT19 Extended Status Words

The content of the words is identical to that defined in the Direct I/O Dynamic Status information for ID*37. The only difference between the extended and the dynamic status is that the dynamic status obtains the immediate status of the device, while the extended status obtains the status of the card after the last I/O request completed.

HP-IB Universal Commands

Any of the HP-IB defined commands, except Pass Control, may be included in the command buffer.

Table 2-17 lists the universal commands with their octal values and a short description of each.

Table 2-17. HP-IB Universal Commands

Command	Octal Code	Description
TLK	100-136	Talk Address. Establish addressed device as a talker on the bus. Any previous talker is automatically unaddressed.
LSN	40-76	Listen Address. Add addressed device to current list of listeners on the bus. If no listeners exist, the addressed device becomes the only one active.
UNT	137	Untalk. Current talker becomes unaddressed.
UNL	77	Unlisten. Causes all current listeners on the bus to be unaddressed.
GET	10	Group Execute Trigger. Causes all currently addressed devices to initiate pre-programmed action.
SDC	04	Selected Device Clear. Causes all currently addressed devices to reset to a predetermined state.
DCL	24	Universal Device Clear. Causes all responding devices to return to a predetermined state (devices need not be addressed).
GTL	01	Go To Local. Causes the currently addressed devices to return to local device control.
LLO	21	Local Lockout. Causes all responding devices to disable front panel local-reset buttons. (Devices need not be addressed.)
SPE	30	Serial Poll Enable. Establishes serial poll mode, such that all cooperating devices, when addressed, provide status information. An ensuing read takes an 8-bit byte of status.
SPD	31	Serial Poll Disable. Terminate serial poll, returning cooperating devices to their normal data transmission state.
PPC	05	Parallel Poll Configure. Assigns an HP-IB DIO line to cooperating device or group of devices for responding to a parallel poll.
PPU	25	Parallel Poll Unconfigure. Resets all parallel poll devices to a predetermined state.
PPE	140-57	Parallel Poll Enable. Enable device to have parallel poll response configured.
PPD	160	Parallel Poll Disable. Disable device from responding to a parallel poll.

RAM Disk Interface Driver IDR37

IDR37 is the RAM disk interface driver. It is the software interface between the operating system and the RAM disk in memory.

IDR37 Read/Write Request

The call sequences for the IDR37 read and write requests are:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)
 Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln* , *track* , *sector*)

Control Word *cntwd*

Figure 2-68 shows the format of the control word (*cntwd*) for a read or write request to IDR37.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		UE	Z	77B						LU					

Figure 2-68. IDR37 Read/Write Request *cntwd* Format

The UE (User Error) and LU (Logical Unit identifier) bits are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

Bit 12, the Z-bit, must be set to zero because *track* and *sector* are integer variables, not control buffer descriptions.

Bits 6 through 11 must be set to 77B for a read or write request to a disk. If bits 6 through 11 are set to 77B for a read or write request to any other type of device, EXEC will reject the request and issue an IO01 error message (not enough parameters in the I/O call).

bufr and *bufln*

The EXEC request I/O buffer is described by the parameters *bufr* and *bufln*, which are defined in the Read/Write Request Parameter Definitions section of Chapter 1.

track and *sector* Parameters

The *track* and *sector* parameters identify the track and logical sector positions to which data is written, or from which it is read. *sector* must be an even integer value for all read or write requests. The legal values of *sector* are shown in Table 2-18.

Table 2-18. Legal Track and Sector Values

Pages/Track	1	2	3	4	5	6	7	8
Sector	0-15	0-31	0-47	0-63	0-79	0-95	0-111	0-127
Number of Directory Entries per Track	30	62	94	126	158	190	222	254

The physical sector value for a given configuration is a fixed number that describes how many 256-byte physical sectors, or blocks, can be stored on each track. Each physical sector is twice as long as the logical sector (128 bytes) used in program calls.

A- and B-Register Contents

The A-Register contains word 6 of the device table after completion of each EXEC read or write request to a disk. The format of word 6 is the same as the format of *stat1*, described in the IDR37 Status Request section of this chapter.

The B-Register contains the positive number of words or negative number of characters that were transmitted during the last read/write request.

Control Word Read Request Example

The following FORTRAN program reads data from track 100, sector 32 of a RAM disk at LU 12, into a buffer variable called BUFR.

```
FTN7X Program Read
      IMPLICIT INTEGER (A-Z)
      DIMENSION BUFR (128)
      :
      BUFLN = 128
      TRACK = 100
      SECTOR = 32
      CNTWD = 12 + 7700B
      CALL EXEC(1, CNTWD, BUFR, BUFLN, TRACK, SECTOR)
      end
```

IDR37 Control Request

IDR37 supports two control requests for configuration purposes. The driver reports an IORQ error and attempts to abort the offending program if:

- A configuration request specifies N pages of memory and N does not contain an integer number of tracks of the specified size.
- A configuration request is made for more memory than the largest free block of dynamic memory.
- A configuration request is made and the disk is already configured.
- A deallocation request is made and the disk is not configured.
- Any other control request is received.

IDR37 Configuration Request

Normally this request is made from CI, therefore, the configuration request is shown as a CI CN command. The CN command can be entered from the CI prompt or in the Welcome file.

The IDR37 configuration command is:

```
CI> CN lu 25B pages [pgs/trk]
```

where:

lu = the LU number of the RAM disk.
pages = the number of pages of RAM disk memory.
pgs/trk = the number of pages per track (1 through 8). The default is 4.

Note that the number of pages of RAM disk memory (*pages*) must be evenly divisible by the number of pages per track (*pgs/trk*).

For example, the following command:

```
CI> CN 2 25B 2000
```

configures the RAM disk at LU 2 to use 2000 pages organized into 4 pages/track.

You would want to use different size tracks because D.RTR defaults directory sizes and their extents to one track. If you were using only a few entries in a directory, you might use a smaller track size. Table 3-1 shows the number of directory entries/track for each configuration.

Once the RAM disk is configured, you must initialize it with the appropriate CI or FMGR commands as if it were a true disk with nothing on it.

IDR37 Deallocate Request

To return RAM disk memory to the system's dynamic memory pool, dismount the disk from the file system and then issue the following command:

```
CI> CN lu 26B
```

Once deallocated, the RAM disk may be reconfigured, if desired.

IDR37 Status Request

The status request call sequence is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4] ] ] )
```

Control Word *cntwd*

Figure 2-69 shows the control word (*cntwd*) format for an EXEC status request to IDR37. The system returns the status of the last disk accessed.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure 2-69. IDR37 Status Request *cntwd* Format

The Z-bit is described below in the *stat3* and *stat4* Status Parameters section.

stat1 and *stat2* Status Parameters

The returned status parameters *stat1* and *stat2* are shown in Figure 2-70.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Driver Type=33B				0		0	0	0	0	0			
<i>stat2</i>	AV		Interface Type=30B				0	I/O Select Code= 0								

Figure 2-70. IDR37 Status Request Parameters

AV (device availability) and Driver Type of *stat1* are defined in the Status Request Parameter Definitions section of Chapter 1. The status code returned to *stat1* bits 0 through 7 is always 0.

stat2 is the interface type (30B) of the disk interface card. AV and Select Code are defined in the Status Parameter Definitions section of Chapter 1. Because there is no real device, the select code is always 0.

stat3 and *stat4* Status Parameters

If the Z-bit is 0, *stat3* and *stat4* return the first two words of the driver parameter area.

If the Z-bit is 1, *stat3* is the name of a buffer to receive the driver parameters, and *stat4* is the length of the buffer.

Driver Parameter Area

The driver parameters of IDR37 are listed in Table 2-19.

Table 2-19. IDR37 Driver Parameters

Parameter	Parameter Description
+1	First page used for this RAM disk
+2	Last page +1 for this RAM disk
+3	Pointer to memory descriptor for the RAM disk memory
+4	Not used
+5	Not used
+6	Number of Tracks
+7	Physical Sectors/Track
+8	Not used

IDR37 Extended Status

IDR37 does not return meaningful extended status.

GPIO/Parallel Interface Card Driver ID*50

ID*50 drives the HP 12006A Parallel Interface Card (PIC), and is a general purpose interface driver. The PIC performs 8- or 16-bit parallel data transfers between the system and parallel I/O devices. The driver is accessed via EXEC calls, and all I/O transfers except transparent latched reads are under card DMA control. Refer to the *HP 12006A Interface Kit Reference Manual* for hardware information.

ID*50 Read/Write Requests

The call sequences for the ID*50 read and write requests are:

Read Request: CALL EXEC(1 , *cntwd* , *bufr* , *bufln* [, *pram1*])
Write Request: CALL EXEC(2 , *cntwd* , *bufr* , *bufln* [, *pram1*])

Control Word *cntwd*

The format of the read/write request control word (*cntwd*) is shown in Figure 2-71.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	x	TL	x=0	LU								

Figure 2-71. ID*50 Read/Write Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are defined in the Read/Write Parameter section, Chapter 1. All x bits (6-8, 10 and 11) are not used for write requests and should be set to zero for write requests.

The TL (Transparent Latched mode) bit configures the PIC so data is dynamically available. Exactly one reading is returned in *bufr* with a transmission log of 1. Transparent latched reads will work with the PIC in either 8-bit or 16-bit mode. In the case of 8-bit, bits 15 through 8 of word 1 of *bufr* contain the reading. One reading will be taken even if the user buffer length is greater than 1 (or less than -1 for bytes). TL is ignored for Control and Write requests, and should not be set for non-PIC cards using ID*50.

The BB (Bypass) bit must be set to 1 if a driver is defined for the device and is to be bypassed.

bufr and *bufln*

The I/O buffer used in the EXEC request is described by parameters *bufr* and *bufln*, defined in the Read/Write Parameter section of Chapter 1.

pram1 Optional Parameter

All bits set in *pram1* will be set in the card control register 31. The value in *pram1* will be OR'd with the first driver parameter word to determine the actual card control register setting before the read or write request is executed.

If *pram1* is not supplied in the EXEC request, the first driver parameter word is used to set the card control register.

The first driver parameter word is set either at generation or with a configure card control request, function code 40B. The card control register for the parallel interface card is described later in this section.

ID*50 Control Request

The call sequence for the control request is:

```
CALL EXEC( 3 , cntwd , pram1[ , pram2[ , pram3[ , pram4 ] ] ] )
```

Control Word *cntwd*

Figure 2-72 illustrates the format of the *cntwd* used by ID*50 when programming a control request to the ID*50.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	Function Code						LU					

Figure 2-72. ID*50 Control Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are defined in the Control Request Parameter section of Chapter 1. The BB-bit (Bypass device driver) must be set to one if a driver has been defined for the device.

The Function Code bits are summarized below.

- 00 Clear and Reset Card.
- 06B Return Dynamic Status of Card.
- 20B Enable Program Scheduling.
- 21B Disable Program Scheduling.
- 23B Enable/Disable Asynchronous Interrupts.
- 24B Set PIC Control Lines.
- 40B Configure Card Control Word.

Function Code 00: Clear and Reset Card

This request sets the interface card to a known state by halting all processing.

Function Code 06B: Return Dynamic Status of Card

A dynamic status request will return information on the immediate status of the device to DVT16 through DVT18. The format of the status information is shown in Figure 2-73. After completion of the request, subsequent calls to RMPAR will retrieve the status information from the DVT.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	x	x	0						Error Code							
DVT17	Transmission Log = 0															
DVT18	HP-IB Interface Card Status															

Figure 2-73. DVT16 to DVT18 Extended Status Words

DVT16 reflects error information for the device that is connected to the interface card. The error codes are device-dependent; a definition of the bits should be obtained from the associated device documentation.

DVT17 is the transmission log (0 on an error, else the original will be posted).

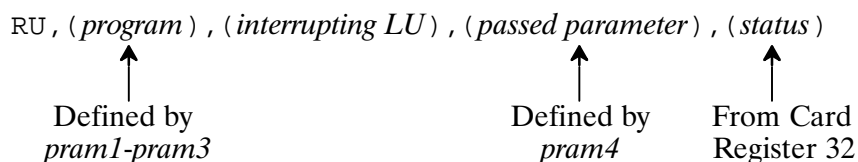
DVT18 reflects the status as read from control register 32.

Function Code 20B: Enable Program Scheduling

This request enables a user program, defined by parameters *pram1* through *pram3*, to be scheduled in response to any subsequent card asynchronous interrupt. The program name must be five characters or; if the name is less than five characters, trailing blanks are used to fill to the required five-character length. A function code 23B request is not necessary if a function code 20B has been executed.

pram4 may contain a single valued integer to be passed to the program, which may be retrieved by a call to RMPAR from the scheduled program. *pram4* will be the second value returned by RMPAR. When the program is scheduled, it will also be passed the card status as read from card register 32. It too may be retrieved in RMPAR parameter 3.

The effective program schedule request is:



In VC+ systems, the program must be attached to the system session.

Function Code 21B: Disable Program Scheduling

The program that was enabled by the function code 20B request is disabled by this request. All card interrupts will be disabled when the card is not busy.

Function Code 23B: Enable/Disable Asynchronous Interrupts

This function enables or disables driver processing of asynchronous interrupts according to the value *pram1*:

pram1 = 0 disable interrupts
 = 1 enable interrupts

The driver will schedule the interrupt program only if the following conditions are met:

1. Program scheduling has been enabled via a function code 20B request.
2. Asynchronous interrupts are enabled.
3. The program is available to be scheduled.

If DVP3 bit 2 is set, asynchronous interrupts are automatically disabled by the driver upon scheduling the interrupt program. Asynchronous interrupts must be enabled again following completion of the program.

Function Code 24B: Set PIC Control Lines

This request changes the PIC Control Line states in the card control register 31. The HP 12006A Control lines are constantly driven out to the device. This request allows you to change the state of these lines without executing a read/write request.

Bits 0-3 of *pram1* are OR'd with DVP1 and the result is output to the card control register 31.

If program scheduling is enabled, *pram1* in function code 24B has no effect. In this instance, you should reconfigure DVP1 via a function code 40B request, and then execute a function code 24B request to load the new DVP1 value into card control register 31.

This request should only be used by HP 12006A applications.

Function Code 40B: Configure Card Control Word

For function code 40B, *pram1* contains a new card control word to be stored in DVP1. Each time an I/O request is received, the value of DVP1 is OR'd with either the value of *pram1* in the request or the value of the Request Code (DVT15), depending on the current state of bit 3 of DVP2 (see below). The OR'd result is then sent to the card.

pram2 is stored in DVP2. DVP2 is defined as follows:

Bit 0: Set to 1 for 8-bit transfers, set to 0 for 16-bit transfers.

Bits 1 and 2 determine DMA AUTO bit for I/O request:

bit 2	bit 1	Input Request	Output Request
0	0	clear AUTO	set AUTO
0	1	set AUTO	set AUTO
1	0	set AUTO	clear AUTO
1	1	clear AUTO	clear AUTO

Bit 3: When set to 1, ignore optional parameter on read/write requests and place the EXEC code (1 or 2) into the card control register, OR'd with Driver Parameter 1.

Bit 4: When set to 1, on driver completion exit, do not CLC 30b,C and do not change the card control word if arming for an asynch interrupt.

Bit 5: When set to 1, DMA Completion interrupts are inhibited on write requests.

Bit 6: When set to 1, an Illegal Interrupt message will be generated if an interrupt occurs and the program to be scheduled is busy. When set to 0, the interrupt will be ignored.

Bit 7: If your device wants to use the PIC for read/writes and asynchronous interrupt, and

1. Your device has device command and device flag tied together, and
2. Your device requires a device command signal when asynchronous interrupts are enabled,

then in certain configurations, the PIC can generate constant interrupts. This is because your device can treat device command assertion for interrupt the same as for read/write, that is, immediate device flag, which generates an interrupt. To allow devices of this type to generate asynchronous interrupts, bit 7 of DVT Parameter Word 2 should be used. When this bit is set, a CLF 30B is issued before the interrupt system is re-enabled. This has the effect of cancelling the device flag (generated by device command tied to device flag) before the system sees the flag. The device can then assert device flag (or STO if IRQEN in Register 31 is set) to signal an asynchronous interrupt. When bit 7 is clear, no CLF 30B is issued. Bit 7 should only be set for PIC devices having device command tied to device flag. If bit 7 is set in other instances, the PIC may lose an interrupt from the device. User devices should use ST0-ST4 to signal the device whether a read, write, or arm for interrupt is the reason for device command being asserted.

Also see DVP3 bit 0 if the device does not require a device command signal when armed for asynchronous interrupt.

Bit 8: When set to 1, CNTL0 is asserted when the PIC is set up for a read request.
When set to 0, bit 0 of DVP1 is used for CNTL0 state.

Bit 9: When set to 1, CNTL1 is asserted when the PIC is set up for a read request.
When set to 0, bit 1 of DVP1 is used for CNTL1 state.

Bit 10: When set to 1, CNTL2 is asserted when the PIC is set up for a read request.
When set to 0, bit 2 of DVP1 is used for CNTL2 state.

Bit 11: When set to 1, CNTL3 is asserted when the PIC is set up for a read request.
When set to 0, bit 3 of DVP1 is used for CNTL3 state.

Bit 12: When set to 1, CNTL0 is asserted when the PIC is armed for program scheduling.
When set to 0, bit 0 of DVP1 is used for CNTL0 state.

Bit 13: When set to 1, CNTL1 is asserted when the PIC is armed for program scheduling.
When set to 0, bit 1 of DVP1 is used for CNTL1 state.

Bit 14: When set to 1, CNTL2 is asserted when the PIC is armed for program scheduling.
When set to 0, bit 2 of DVP1 is used for CNTL2 state.

Bit 15: When set to 1, CNTL3 is asserted when the PIC is armed for program scheduling.
When set to 0, bit 3 of DVP1 is used for CNTL3 state.

All bits not listed are ignored.

By setting bits 12-15, the device can examine the CNTL lines along with Device Command and determine if the PIC is armed for program schedule or waiting for a read or write.

By setting bits 8-11, the device can examine the CNTL lines along with Device Command and determine if the PIC is set up for a read or write request.

Bits 12-15 and 8-11 are always IOR'd with the default card control word in DVP1 and the result is sent to the card. These bits should not be set for non-PIC cards using ID*50.

Bits 1 and 2 work together to determine the AUTO bit. Bits 4 and 5 may be useful to other users with highly buffered devices, where completion of DMA does not always indicate completion of the transfer.

pram3 is stored in DVP3. DVP3 is defined as follows:

Bit 0: When set, do not assert DVCMD to the device when arming the PIC for an asynchronous interrupt. Also see DVP2 bit 7.

Bit 1: When set, enable IRQEN in the card control register 31 before arming the PIC for asynchronous interrupts, if enabled. This allows the device to assert ST0 or device flag for interrupt service. If DVP2 bit 4 is set, this bit will have no effect.

Bit 2: When set, the driver will disable asynchronous interrupts upon scheduling the interrupt program. You must reenale interrupts after the interrupt program has completed.

Bit 3: When set, the driver will hold off pending interrupts if the interrupt program is not dormant. When the program becomes dormant, it is scheduled and the card is armed for interrupt again. The interrupt program is checked for dormancy every 20 milliseconds.

Bit 4-15: Reserved.

If both bits 2 and 3 are set, bit 3 will be ignored.

ID*50 Status Reporting

The call sequence for the ID*50 status-reporting request is:

```
CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]])
```

Control Word *cntwd*

Figure 2-74 shows the control word (*cntwd*) format for an EXEC status request for ID*50.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0						LU					

Figure 2-74. ID*50 Status Request *cntwd* Format

stat1 and stat2 Status Parameters

The format of the returned status parameters (*stat1/stat2*) is shown in Figure 2-75.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type						Status							
<i>stat2</i>	AV		Interface Type=50						0	0	I/O Select Code					

Figure 2-75. *stat1* and *stat2* Status Words Format

All bits in *stat1* and *stat2* are defined in the Status Request Parameter section of Chapter 1.

stat3 and stat4 Status Parameters

If the Z-bit of *cntwd* is zero, *stat3* returns the first word of the driver parameter area, and *stat4* returns the second word of the driver parameter area.

If Z is set to 1, *stat3* is the buffer to return the driver parameter area, and *stat4* is the length of the *stat3* buffer.

Driver Parameter Area

ID*50 requires three words in the driver parameter area and 2 in the DVT extension, all of which are set by the configure card control request (function code 40B). Words 4 and above of the driver parameter area may be defined as required by device drivers using ID*50.

An IO07 error will occur if an incorrect number of DVT parameters and/or extension words was specified in the generation.

ID*50 Parallel Interface Card Control Register

The driver automatically includes the output control word for each I/O operation with the parallel interface card (PIC). Bits 4 through 15 of the output control word are, in general, supplied from the first word of the driver parameter area established at system generation. These bits can be redefined by a configure card control request (function code 40B). Refer to the *HP 12006 Parallel Interface Reference Manual* for details of the functional operation of the card. The PIC control register format is shown in Figure 2-76.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSTU	x	DFS	DCL	PLV	LBYEN	IRQEN	TSTL	SRM	DRM	SNS	GP Line Bits				

Figure 2-76. PIC Control Register Format

TSTU: Test Upper byte. Used for diagnostic checkout of the upper byte of the control and status registers.

- x:** Not used.
- DFS:** Device Flag Select. Selects the edge of the device flag (DEVFLG) pulse that will cause the data service request (SRQ) to the I/O master. The transition signals ‘data ready’ on the input and ‘ready for more data’ on the output:
- 1 positive-going edge
 - 0 negative-going edge
- DCL:** Device Command Clear. Selects the edge of DEVFLG that will clear Device Command (DC). This bit only has an effect if PLV (bit 10) is clear.
- 1 clear DC on the opposite edge as that selected by DFS (that is, DFS 1 = positive-going edge; DCL 1 = negative-going edge).
 - 0 clear DC on the same edge as that selected by DFS.
- PLV:** Pulse/Level Select. Selects between a level-mode or a pulsed command signal. The 200-nanosecond pulsed signal is one period of the system clock (SCLK).
- 1 pulsed command signal
 - 0 level-mode device command
- The Device Command signal is asserted during execution of an STC command, or produced automatically during each DMA transfer.
- The level-mode Device Command is cleared on a selected edge of DEVFLG. If the control flip-flop is clear, the level-mode Device Command is held in the cleared state while DMA is running.
- LBYEN:** This bit set to 1 enables early termination of a DMA input transfer. Causes the assertion of ST1 for shutdown of a DMA input before word-count rollover occurs. When this feature is used, ST1 should be asserted prior to the DEVFLG accompanying the last word or byte.
- IRQEN:** Interrupt Request Enable. Enables the interrupt request signal to the I/O master. ST0 is asserted to set the flag, thereby causing an interrupt.
- 1 Enable interrupt request.
 - 0 Disable interrupt request from being asserted.
- TSTL:** Test Lower Byte. Is connected to bit 7 of the status word. When set, enables diagnostic testing of the operation of the lower byte of the control and status registers.
- SRM:** Select Status Register Mode.
- 1 Load status register (register 32) on each DEVFLG assertion.
 - 0 Cause status register to act like a transparent latch so that status is dynamically available.
- DRM:** Select Data Register Mode.
- 1 Load input data register on each DEVFLG assertion.
 - 0 Cause data register to act like a transparent latch so that 16 data bits are dynamically available.

SNS: Interface Sense Select. Select either ground-true or high-true for transfer of all 40 status, data, and control signals.

0 Invert all signals for ground-true device interface.

1 Do not invert signals; a high-true positive device interface results.

GPL: General Purpose Lines. The four GPL bits may be used to address, control, or handshake with various types of peripheral devices. They are latched on the HP 12006A during DMA self-configuration, and are constantly driven out to the device.

ID*50 Parallel Interface Card Status

DVT18 reflects the parallel interface card status from register 32 after each EXEC request to the driver. This value can be obtained by making a dynamic status control request to obtain the immediate status of the device, or by making a call to RMPAR after an EXEC read, write or control request to obtain the status of the last request. DVT18 is shown in Figure 2-77.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSTU	SLAVE	DMAON	PACK	GPL			TSTL	DFF	FLAG	DCSS	Status				

Figure 2-77. DVT18 Word Format for ID*50 Dynamic Status Request

TSTU: Used for diagnostic purposes to check out the operation of the upper byte of the control and status registers.

SLAVE: Used to request slave mode (access to the Virtual Control Panel code). If this bit is set to 0, PIC is in normal mode of operation. If set to 1, PIC is requesting slave mode.

DMAON: Q output of DMAON flip-flop. This bit is set to 0 when DMA is running, and set to 1 at all other times.

PACK: Q output of PACK flip-flop. This bit is set to 1 whenever byte mode DMA is running.

GPL: General Purpose Lines. These are identical to the GPL bits of the control register and are used for addressing, control, or handshaking with various types of peripheral devices.

TSTL: Used for diagnostic purposes to check out the operation of the lower byte of the control and status registers. This bit is connected to the TSTL-bit of the control register.

DFF: Used for diagnostic purposes to check out the operation of the device command flip-flop.

FLAG: Contains the inverted sense of the device flag line. Initially, FLAG should be in its nonactive state. The information is chiefly used for diagnostic purposes.

DCSS: Contains the state selected for the device command sense switch.

- 1 Active-high device command selected.
- 0 Active-low device command selected.

The four STATUS lines are loaded on each DEVFLG assertion if bit 6 (SRM) of the control register is set to 1. If SRM is set to 0, the four lines will reflect dynamic status information of the device connected to the PIC.

ID*50 Program Scheduling

Program scheduling on interrupt is set up by executing a control 20B request. The program name must always be specified in this request. The program must be RP'd in the system session (on VC+) and should terminate saving resources. Program scheduling is disabled via a control 21B request.

Asynchronous interrupt handling can be enabled/disabled via a control 23B request. A control 20B request automatically enables asynchronous interrupt handling. Using this request, interrupts can be enabled/disabled with affecting the program name setup with the 20B request.

When program scheduling has been enabled, the device can interrupt by asserting device flag. If DVP3 bit 1 has been set, the device can also interrupt by asserting ST0.

If the device expects no DVCMD assertion when the PIC has been armed for interrupt, DVP3 bit 0 can be set. If the device expects DVCMD to be asserted when armed, and DVCMD and device flag are tied together, DVP2 bit 7 should be set. Should this bit not be set, continuous device interrupts, and a subsequent system hang might occur.

ID*50 can be configured to automatically signal the device when the PIC is armed for interrupt by setting the appropriate bits 12-15 in DVP2. When these bits are set, the associated control lines, CTL0-3 will be asserted to the device.

Should the interrupt program be non-dormant when the driver receives an asynchronous interrupt, the driver can be configured for one of four actions:

1. DVP2 bit 6, DVP3 bits 2, 3 all not set. The interrupt is ignored.
2. DVP2 bit 6 set, DVP3 bits 2, 3 not set. The interrupt is ignored. An illegal interrupt message is printed on the system console.
3. DVP2 bit 6 set, DVP3 bit 2 set, DVP3 bit 3 not set. ID*50 disables asynchronous interrupts upon scheduling the interrupt program. When the program is ready to terminate, it should execute a control 23B request to reenables interrupts. Alternately, the 23B request could be issued later. This mechanism gives you full control over when asynchronous interrupts are enabled.
4. DVP2 bit 6 set, DVP3 bit 2 not set, DVP3 bit 3 set. ID*50 holds off exactly one pending interrupt and then disables interrupts internally. The program is checked for availability every 20 milliseconds. When it becomes available, the program is scheduled to process the pending interrupt and interrupts are enabled again automatically. This is the preferred method of driver configuration.

PIC Intercomputer Communications Driver ID*52

The Parallel Interface Intercomputer Communications Driver (ID*52) provides a means of interconnecting two systems and passing data between programs at each system, and allows interactive programs to be controlled remotely across the communications link. This driver requires two HP 12006A PIC cards connected to the 48-pin, HP-supplied connector kit. For bi-directional communication, the functionality of ID*52 may not be satisfactory for use with only one pair of PIC cards. It is recommended that two pairs of PIC cards be used, one pair for each direction of communication.

For additional information, refer to the *HP 12006A Parallel Interface Card Reference Manual*, part number 12006-90001.

In using the information in this section, refer also to the preceding discussion of the GPIO/Parallel Interface Card Driver ID*50.

ID*52 Read/Write Requests

The call sequences for the read, write, and write/read requests are:

Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln*)
Write Request: CALL EXEC (2 , *cntwd* , *bufr* , *bufln* [, *pram1*])
Write/Read Request: CALL EXEC (1 , *cntwd* , *bufr* , *bufln* , *wbuf* , *wbufl*)

Control Word *cntwd*

The format of the control word (*cntwd*) for performing an EXEC read or write request to driver ID*52 is shown in Figure 2-78.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	x=0						LU					

Figure 2-78. ID*52 Read/Write Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are defined in Chapter 1 under the Read/Write Parameter description. The x-bits are not used and should be set to 0. The BB-bit (Bypass device driver) must be set to 1 if a driver is defined for the device. The Z-bit should always be set for a Write/Read request.

bufr and *bufln*, *wbuf* and *wbufl*

The I/O buffers for write and read used in the EXEC request are described by parameters *bufr* and *bufln*, defined in Chapter 1. The *wbuf* parameter in the write/read request should contain the buffer to be written prior to the read operation.

pram1 Optional Parameter

On write only, the first parameter (or zero) is sent to the receiver to be returned in the fourth status parameter via RMPAR, as defined in the Extended Device Status section of Chapter 1.

ID*52 Control Requests

The call sequence for the control request is:

```
CALL EXEC(3 ,cntwd ,pram1[ ,pram2[ ,pram3[ ,pram4] ] ] )
```

Control Word *cntwd*

Figure 2-79 shows the format of the control word (*cntwd*) used by ID*52 for programming a control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	Function Code						LU					

Figure 2-79. ID*52 Control Request *cntwd* Format

The NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are defined in Chapter 1. The BB-bit (Bypass device driver) must be set to one if a driver is defined for the device. The Function Code bits are:

- 00 Clear and Reset Card.
- 01B Send EOF.
- 06B Return Dynamic Status of Card .
- 11B Send Top-of-Form (same as Send EOF).
- 20B Enable Program Scheduling.
- 21B Disable Program Scheduling.
- 22B Set Timeout.
- 23B Disable Program Scheduling (same as 21).
- 45B Use level mode for Device Command signal.
- 46B Use pulse mode for Device Command signal.

Function Code 00: Clear and Reset Card

This request sets the interface card to a known state by halting all processing.

Function Code 01B: Send EOF

This request will send an End-of-File (EOF) indicator to the receiver. The status returned from the request will have the EOF bit set, and the request transmission log will be zero.

Function Code 06B: Return Dynamic Status of Card

A dynamic status request will return status information into DVT16 through DVT18 reflecting the immediate status of the device. After completion of the request, a call to RMPAR will retrieve the status information from the DVT. The format of the status information is shown in Figure 2-80.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16	x	x	0						Error Code							
DVT17	Transmission Log = 0															
DVT18	Card Status															

Figure 2-80. DVT16 to DVT18 Extended Status Words

In DVT16, the octal and decimal error codes returned in bits 0 through 5 are:

Octal	Decimal	
1	1	Illegal request.
3	3	Line timeout. Request is flushed and remote computer is interrupted if it has an active request. The device is not downed.
24	20	Busy error. Write or Send EOF was attempted when remote computer was trying to send, or Read was attempted when remote computer was attempting to receive. Request is flushed; the device is not downed.
77	63	Request retry. This error will not be seen by the program unless it sets the UE bit in the control word of the original I/O request. Normally, request is retried and the device is not downed. Retries can occur if: <ul style="list-style-type: none"> – The local or remote computer's power failed. – The remote computer's request was aborted or timed out.

DVT17 contains the transmission log (number of words received on a read, or original number of words specified on a write will be posted).

DVT18 contains the status as read from control register 32.

Function Code 11B: Top-of-Form EOF

This request performs an EOF function identical to control request 01.

Function Code 20B: Enable Program Scheduling

This request enables a user program, defined by parameters *pram1* through *pram3*, to be scheduled in response to any subsequent asynchronous interrupt. Note that the program name must be five characters; names of less than five characters will have trailing blanks added to reach the five-character length. The asynchronous interrupt will then automatically occur whenever the remote driver initiates an active request and the local driver is not busy.

pram4 may contain a single valued integer to be passed to the program, which may be retrieved by a call to RMPAR in the scheduled program. *pram4* will be the second value returned by RMPAR. When the program is scheduled, it will also be passed the card status as read from card register 32. It may be retrieved in RMPAR parameter five.

The effective program schedule request is:

```
RU, (program) , (interrupting_LU) , (passed_parameter) , , , (status)
```

↑
↑
↑
 Defined by
 Defined by
 From Card
 pram1-pram3 *pram4* Register 32

If the program is busy, the driver will attempt to schedule the program at 100-millisecond intervals until either the schedule succeeds, the remote request is removed, or the driver receives a new request.

If the first two characters of the program name to be scheduled are “FM” (that is FMGR), and the program is busy, the system attention flag will be set instead of retrying the schedule at 100-msec intervals. This will result in the system prompt being received by the remote requestor and thus facilitate remote interactive control of a system.

Function Codes 21B/23B: Disable Program Scheduling

The program that was enabled by the enable program scheduling control request above will be disabled upon this request. The card interrupts will be disabled when the card is not busy.

Function Code 22B: Set Timeout

pram1 will contain the new communications line timeout, specified in tens of milliseconds. If zero is specified, no timeout will be enabled for the device, and a request can remain active on the line indefinitely.

Function Codes 45B/46B: Use Level/Pulse Mode DVCMD

These function codes specify how to set the PLV Bit (bit 10) of the output control word (described in the preceding section “ID*50 Parallel Interface Card Control Register”). Function code 45B specifies level mode Device Command (DVCMD) signal. Function code 46B specifies pulse mode DVCMD signal. They are mutually exclusive.

ID*52 defaults to pulse mode, however, note that the pulse is approximately 227 nanoseconds and the capacitor at C1 on the PIC card filters out signals of less than 525 nanoseconds duration.

ID*52 Status Reporting

The call sequence for the status request is:

```
CALL EXEC(13 ,cntwd ,stat1[ ,stat2[ ,stat3[ ,stat4]]])
```

Control Word *cntwd*

Figure 2-81 shows the control word (*cntwd*) format for an EXEC status request for ID*52.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		Z	0						LU						

Figure 2-81. ID*52 Status Request *cntwd* Format

The LU (Logical Unit assignment) and Z-bits are as defined in the Status Parameters section of Chapter 1.

stat1 and *stat2* Status Parameters

The format of the returned status parameters (*stat1/stat2*) is shown in Figure 2-82.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i>	AV		Device Type						Status							
<i>stat2</i>	AV		Interface Type=52						0	0	I/O Select Code					

Figure 2-82. *stat1* and *stat2* Status Words Format

All bits in *stat1* and *stat2* are as defined in Chapter 1.

stat3 and *stat4* Status Parameters

If the control word Z-bit is 0, *stat3* returns the first word of the driver parameter area, and *stat4* returns the second word of the driver parameter area.

If Z is set to 1, *stat3* is the buffer to return the driver parameter area, and *stat4* is the length of the *stat3* buffer.

Driver Parameter Area

ID*52 does not use the driver parameter area.

ID*52 Parallel Interface Card Control Register

For each I/O operation with the parallel interface card (PIC), the driver automatically includes the output control word as shown in Figure 2-83. Bits 4 through 15 of the output control word are described in the previous ID*50 Parallel Interface Card Control Register section.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSTU	x	DFS	DCL	PLV	LBYEN	IRQEN	TSTL	SRM	DRM	SNS	RTS	RTR	EDT	INT	

Figure 2-83. PIC Control Register Format

Bits 0 through 3 are used to manage the intercomputer communications line protocol:

INT: Interrupt remote. Setting this bit to 1 causes the computer to be interrupted, providing IRQEN is enabled in the PIC card control word. ID*52 is enabled to receive this interrupt whenever a DMA request is active, or when a DMA request is not active but asynchronous program scheduling is enabled.

EDT: Early DMA terminator. The sending computer asserts this line just prior to sending its last word of data. This causes the current read quadruplet to complete and post any residue. EDT is also set while an active DMA read is in progress so that the sending driver can synchronize in determining whether or not it must set the AUTO bit in the DMA control word of the first quadruplet. The AUTO bit also is set in the last quadruplet of the data transfer if an odd number of bytes was specified in the request.

RTR: Ready to Receive. The remote computer currently has an active read request.

RTS: Ready to Send. The remote computer currently has an active write request.

Bits 4 through 15 of the output control word are described in the preceding section under the ID*50 Parallel Interface Card Control Register discussion.

ID*52 Parallel Interface Card Status

Figure 2-84 shows the parallel interface card status stored from register 32 into DVT18 after each EXEC request to the driver. This value can be obtained by making a dynamic status control request to obtain the immediate status of the device, or by making a call to RMPAR after an EXEC read, write, or control request to obtain the status of the last request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSTU	SLAVE	DMAON	x	RTS	RTR	EDT	INT	TSTL	DFF	FLAG	DCSS	RTS	RTR	EDT	INT

Figure 2-84. DVT18 Format for ID*52 Dynamic Status Request

Status bits 15-12 and 7-4 are described in the ID*50 Parallel Interface Card Status section. The usage of RTS/RTR/EDT/INT was described in the immediately preceding discussion of Control Register contents. Bits 11-8 are the current state of the local computer's control lines and bits 3-0 are the current state of the remote computer's control lines.

If status bit 3 is set to 1, the remote computer is actively attempting to send (write).

If status bit 2 is set to 1, the remote computer is actively attempting to receive (read).

Serial I/O Drivers

This section describes the serial I/O drivers supported on an RTE-A Operating System that were first introduced at Revision 4010. Both serial I/O interface and serial I/O device drivers have been unified to gain the advantages of universal device driver compatibility, ease of use, speed, efficiency, increased applicability, and reduced table size. The transition of drivers from the original drivers is summarized by the table below.

Table 2-20. Comparison of Original and Current Drivers

Driver			Device
Original	Current	First Released:	
DD*00	DDC00	4010	Terminal
DD*20	DDC01	4010	Terminal with CTU device
none	DDC01	4010	Terminal with printer
Driver			Interface Card
Original	Current	First Released:	
IDM00	none	4010	HP 12040A-C 8-Channel MUX
none	ID800	4010	HP 12040D 8-Channel MUX
none	ID801	4010	HP 12040D 8-Channel MUX with modems
none	ID400	4010	HP 12100A 4-Channel OBIO
ID*00	ID100	5000	HP 12005A/B ASIC Card
ID*01	ID101	5000	HP 12005A/B ASIC Card with modems
none	IDZ00	5000	HP TELNET Pseudo Terminal LU

Drivers IDM00, ID*00, ID*01, DD*00, and DD*20 relocatables continue to exist as part of the RTE-A Operating System and are documented in this manual in Appendix J, “Serial I/O Drivers (prior to Rev. 4010)”.

Special Characters

The special characters recognized in normal ASCII reads are:

Carriage Return	(octal 015)	(Ctrl-M)
Line Feed	(octal 012)	(Ctrl-J)
Backspace	(octal 010)	(Ctrl-H)
Delete	(octal 177)	
EOT	(octal 004)	(Ctrl-D)

The special characters are described in the following subsections.

Carriage Return (octal 015) (Ctrl-M)

This character is the read terminator for normal ASCII reads, and the default terminator for transparent ASCII reads. The read request is complete when the carriage return is received and a CRLF sequence is output by the driver to move the cursor to the next line on the CRT.

Line Feed (octal 012) (Ctrl-J)

This character is stripped from the incoming data by some existing drivers. ID400, ID800, ID801, and ID100 do not strip line feeds. For applications where those drivers may be used, and line feeds are generated by the device connected to the serial interface, the utility subroutine HpCrtStripChar should be called to ensure proper operation.

Backspace (octal 010) (Ctrl-H)

This character causes the preceding character in the buffer to be erased. Successive backspace characters remove a character at a time from the buffer until the buffer is empty. Note that if echo is enabled and the read is not transparent, the cursor moves to the left on the screen. The drivers then issue a blank, causing the character to disappear from the screen, followed by another backspace to reposition the cursor. Thus as each backspace is entered, a character is erased from the screen.

Delete (octal 177)

This character, which is labeled DEL or RUBOUT on some terminals, causes the driver to reset the buffer pointer. This causes the driver to forget all the characters that have been entered into the buffer, so that you can “start over”. If echo is enabled and the read is not transparent, the driver acknowledges the delete key by sending a backslash (\) CRLF sequence to the terminal.

EOT (octal 004) (Ctrl-D)

When this character is received and the read is a normal ASCII read, it causes the read to be terminated, the transmission log to be set to zero, and bit 5 to be set in the status bits. The cursor position is *not* altered (no CRLF is issued). Any data that the user entered prior to the Ctrl-D is lost.

Break

The break condition terminates a read, write, or control request. For more information, see the Program Scheduling Conditions section in this chapter.

ASCII vs. Binary Read Modes

One advantage of a normal ASCII read is that no action is taken until a carriage return is entered. You can change the input buffer to correct errors or decide that what was entered is not desired. This is essential in creating a “friendly” program. When single-character reads are used, each keystroke is evaluated as it is entered, thereby denying the user an opportunity to think about the input.

Additionally, there are hardware advantages to posting a read that terminates on a character instead of a count. For some I/O cards, the CPU overhead is considerably less for an ASCII read than a binary read. Input via a series of single-character binary reads entails interaction up to the program level for each character. The program may not be the highest priority program and may have even been swapped out to disk, thereby causing dropped characters if the next read is not posted within one character time. Unless FIFO buffer mode is in effect, the system will lose characters when even moderate baud rates are used with single-character input.

For these reasons, you should avoid single-character reads in all but the most trivial cases (such as the permission prompts used by LI).

Serial I/O Interface Drivers

The following interface drivers are currently available.

ID100	for HP 12005A/B ASIC
ID101	for HP 12005A/B ASIC with modems
IDZ00	for TELNET Pseudo Terminal LUs
ID400	for HP A400 On-Board I/O
ID800	for HP 12040D MUX
ID801	for HP 12040D MUX with modem

For a brief review of the features of the new drivers and how they differ from earlier versions, see the Comparison of Drivers appendix.

Serial I/O Device Drivers

Driver DDC00 has replaced driver DD*00 for terminals, and DDC01 has replaced DD*20 for terminals with CTUs. For additional information on device and interface drivers, refer to Appendix E of the *RTE-A System Generation and Installation Manual*, part number 92077-90034.

DD*00 and DD*20 are documented in the Serial I/O Drivers (Rev. 4010) appendix of this manual.

Slave device support using DDC01

DDC00 and DDC01 are the device (logical level) drivers for the serial drivers. DDC00 provides support for terminals (both HP and non-HP), directly connected printers, and 'black boxes'. DDC01 adds functionality to DDC00 to support 'slave devices' on the HP terminals. DDC01 is a superset of DDC00, so only one of them needs to be generated into the system.

Supported slave devices are the Cartridge Tape Units which were built into the HP 264x terminals, or printers (either external or internal) which are used by many of the HP terminals. In addition, several of the terminal emulator packages for PCs and workstations also emulate slave devices.

When a slave device is present on a terminal, you might use it in one of two ways. One way is to write programs to perform specific functions and embed the knowledge of the terminal protocols in the program. The other way to use the slave devices is to make the knowledge of the terminal protocols a part of the standard operating system, thus making the devices accessible to all programs without special coding. This is the approach that is taken in RTE-A on the HP 1000 systems when DDC01 is used.

On an RTE-A system, the slave devices are visible as separate LUs. Because RTE-A treats them as LUs, any program that allows you to direct its I/O to an LU can be used with slave printers or CTUs.

Capability

CTUs are a good emulation of the 9-track magnetic tape drives. A program that works with DD*23 or DD*24 should also work with DDC01.

Slaved printers are not quite as good at emulating the system lineprinters, because the terminal firmware appends a carriage return or line feed to each record. Overprinting is impossible, either from the FORTRAN carriage control characters in column 1, or from an underscore at the end of the line as used with terminals.

The driver also does not try to eject half-page, quarter-page, and so on, because it has no information about the printer's characteristics. Therefore, the driver performs only one of the following: (1) page eject, or (2) ignore carriage control.

Generation

To support slave devices, minor changes are required in your system generation answer file.

In the relocation phase, replace DDC00.REL with DDC01.REL. The penalty for using DDC01 instead of DDC00 is approximately 800 words.

In the table generation phase, you must include a DVT for each device, in addition to the DVT for the terminal. For example, for an HP 264x with dual CTUs and a slaved printer, 4 DVTs are required, one for the terminal and three for the slaved devices. When assigning the LUs for the DVTs, keep in mind that the terminal LU must be the LU with the lowest number. It is a good idea to use LU numbers that are related in some way to help the user remember the LU numbers. As an example, you might assign the left CTU drive on all terminals to an LU 10 higher than the LU of the terminal itself, the right CTU 20 higher, and the slaved printer 30 higher.

The generation records that are available for slaved device support are:

HP_Ctu:L	left cartridge drive on an HP 264x
HP_Ctu:R	right cartridge drive on an HP 264x
HP_Slaved_Serial	printers connected via a serial port to the terminal
HP_Slaved_HPIB	printers connected via an HP-IB port to the terminal
HP_Internal_Prtr	printers built into the HP 262x terminals

Selecting from the printer generation records can be confusing. That is because the terminal firmware can address the printer as either device 4 (usual for serial) or device 5 (usual for HP-IB) without regard to which interface method is used for the physical connection. First, try HP_Slaved_Serial and then try HP_Slaved_HPIB. HP_Internal_Prtr is simply an alias to HP_Slaved_Serial.

In the node list section, the slave devices and the terminal LUs must be in a single node list.

Initialization

Only the host device, not the slave device, must be initialized in the system Welcome file. If you want to initialize a slave device in your Welcome file, you must initialize it after you have initialized the host device.

TELNET Pseudo Terminal LU

TELNET is an NS-ARPA/1000 and ARPA/1000 service used to communicate as a virtual terminal. TELNET is run from the user node that requests connection to a remote node. Driver IDZ00 redirects application program requests transparently from a Remote Server by way of EXEC calls.

For more information, refer to the *NS-ARPA/1000 User/Programmer Reference Manual*, part number 91790-90020, and the *NS-ARPA/1000 Generation and Initialization Manual*, part number 91790-90030; or the *ARPA/1000 User's Manual*, part number 98170-90002, and the *ARPA/1000 Node Manager's Manual*, part number 98170-90001.

Using the Bypass Bit (Bit 15)

Use of the bypass bit is not recommended and is not supported. Setting this bit to 1 causes the device driver not to be entered, and there will be no timeout in effect. Although this will slightly increase throughput, there should be no need to use it. Supervisory functions such as program scheduling and multibuffering are performed by the device driver. Normal reads and writes are performed by the interface driver.

Read Request

The calling sequence for a read request is

```
CALL EXEC(1, cntwd, bufr, bufln[, pram3, pram4])
```

bufr and *bufln*

The *bufr* parameter is the user buffer that receives data from the read request. *bufr* cannot be a FORTRAN character data type. Refer to the HpCrtReadChar subroutine in the *RTE-A • RTE-6/VM Relocatable Libraries Reference Manual*, part number 92077-90037, for information on FORTRAN character variables.

The *bufln* parameter is the request length. It defines the maximum transfer size. Fewer characters may be transferred in an ASCII mode read. After the read, the B-Register contains the transmission log, a positive value in the same units as the original request length indicating the number of valid characters stored in the user buffer.

If *bufln* is positive, it indicates the maximum number of words to be transferred (up to 32767 words for ID400, up to 16384 for the other serial drivers). The transmission log is expressed in a positive number of words, the rounded-up character count divided by 2. If *bufln* is negative, it indicates the number of bytes to be transferred. The returned transmission log will be the positive number of characters received. In binary mode, a zero-length read terminates immediately with a transmission log of zero. In ASCII mode, a zero-length request terminates when the terminator or timeout is detected with a transmission log of zero.

If the input data ends on an odd (left) byte, the last word is padded with a blank (octal 40) in ASCII mode or a null (octal 0) in binary mode. This byte is *not* included in the transmission log computation. In all cases, the contents of the user buffer beyond the last valid word as indicated by the transmission log is undefined. There is no guarantee that the buffer has not been altered by editing that may have occurred, nor should it be assumed that the terminator byte, if any, is present in the buffer.

A read clears the DVT6 error bits. A null binary read may be used to clear the bits, without any effect on the card.

Caution When using the ASIC card and reading from devices capable of delivering characters in a burst, the read buffer must be declared longer by one or two characters than the maximum data length, including the terminator. Thus if you are reading the HP terminal primary status, which is nine bytes long including the carriage return, the read length should be either ten characters or five words. If this is not done, the ASIC card may be unable to accept the data properly.

***cntwd* (Read Request Control Word)**

The read control word (*cntwd*) is shown in Figure 2-85.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OS			Z	R	TR	R	EC	AH	BI	Device LU					

Figure 2-85. Serial Driver Read Request Control Word (*cntwd*)

OS: RTE option bits; refer to the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

Z: Write/Read bit. If the Z bit is clear, *pram3* and *pram4* in the call are ignored. If the Z bit is set, *pram3* and *pram4* in the call describe a buffer (referred to as the Z-buffer). The Z-buffer is written to the external device before performing the read or write described by *bufr* and *bufln*. *pram3* is the buffer address and *pram4* is a character count, if negative, or word count, if positive. The Z-buffer is sent in normal ASCII mode, with the data bytes followed by a driver generated Carriage Return Line Feed (CRLF), unless disabled by a trailing underscore.

When HP protocol is enabled, the control buffer is sent with ENQ/ACK handshaking enabled. The normal data buffer, described by *bufr* and *bufln*, is transferred according to the rules established by the remainder of the bits in the control word.

Z-buffer calls enable you to execute a command and a data transfer in a single call. A write call followed by a read call cannot always respond quickly enough to prevent loss of data, especially in the case where the external device has a quick "turnaround" time and the user program is pre-empted by other higher priority programs. The write/read call should be able to receive the data under almost all circumstances, because the I/O card is switched from the transmit mode to the receive mode in less than one character's time following transmission of the Z-buffer.

For a special use of the Z-buffer on the A400 4-channel MUX, refer to the section "Pseudo Full Duplex (A400 Only)".

TR: Transparency bit. During an ASCII read, this bit determines whether special character processing is done. When the bit is clear, the incoming data stream is examined for special characters, which cause the contents of the user buffer to be changed before the read is completed. This is sometimes referred to as input editing. If the transparency bit is set, the termination character is the only special character. The terminator character can be altered. Refer to the section "Function Code 17B Definable Terminator" later in this chapter.

This bit and the binary bit (bit 6) indicate read request type as described in Table 2-21.

R: Reserved; should be set to 0.

EC: Echo bit. Echo is enabled when this bit is set and disabled when it is clear.

AH: Auto-Home bit. Auto-Home has no effect in character mode transfers. Refer to the “Block Mode Read” section later in this chapter.

BI: Binary bit. Binary reads are performed when the binary bit is set. A binary read ends when the user’s request length has been satisfied, that is, the buffer has been filled or timed out. All characters received are saved in the buffer.

ASCII reads are performed when the binary bit is clear. An ASCII read ends only when a terminator character is detected in the incoming data stream or timeout. The terminator character is a delimiter only, and as such might not be saved in the user’s buffer. If more characters than will fit in the user’s buffer are received before the terminator character is received, the excess characters are lost without notification.

This bit and the transparency bit (bit 10) indicate read request type as described in Table 2-21.

Table 2-21. Character Mode Transfers

Read Type	TR BI	Terminating Condition	Description of Read
NORMAL ASCII	0 0	Detection of CR or EOT Does not terminate on character count	Allow special character recognition. If echo enabled, echo each incoming character as received. Send LF for terminating CR if echo is enabled, send CRLF if disabled. Send “\CRLF” for DEL if echo enabled, send nothing if disabled.
TRANSPARENT ASCII	1 0	Detection of CR (or other if CN17 has changed it). Does not terminate on character count	No special character recognition other than the terminator. If echo enabled, echo each incoming character as received. Send LF for terminating CR if echo is enabled, send CRLF if disabled. Note that the terminating condition may be changed with a CN17 call.
BINARY	X 1	Satisfy character count	If echo enabled, echo each incoming character as received.
<p>Note: This table is for character mode transfers. For information concerning block mode transfers, see the Block Mode Read section later in this chapter.</p>			

Pseudo Full Duplex (A400 Only)

On the A400 multiplexer (driver ID400), a special use of the Z-buffer bit in conjunction with the Binary and Transparent bits set to 1 in FIFO mode allows a simultaneous read and write. The limited size of the A400's FIFO buffer (95 characters) makes this useful. Using this pseudo full duplex mode, large writes may be performed without concern of loss of incoming data. (If less than 95 characters will come in during this time period, or XON/XOFF protocol is used, normal FIFO buffering may be used.) This is not needed on the 8-channel MUX during a write.

The data is written from the Z-buffer in transparent mode (rather than the usual ASCII mode), while any received data is simultaneously posted to the data buffer. The transmission log of the read will be in bytes in DVT 17. The read data is processed only on word boundaries, so if an odd number of bytes is received, one byte will be left on the card in the FIFO buffer. The transmission log of the write will be in DVT 19. The call terminates when the write data is transmitted; it does not wait for the read to complete. This request is rejected under any of the following conditions:

- FIFO buffer mode is not enabled.
- HP protocol mode is enabled.
- the interface driver is not ID400.

This call is very specialized and should be used only when the normal techniques of FIFO buffering are not adequate.

Block Mode Read

The Auto-Home bit is used by block mode reads. The Echo bit is ignored in block mode reads. For additional information on the PG (Page Mode) bit, refer to “Device/Interface Driver Communication” in this chapter.

Table 2-22. HP Block Mode

Read Type	PG BIT	Terminating Condition	Description of Read
BLOCK LINE	0	Detection of CR. No termination on character count.	Inhibit echo. If first character received is a DC2, send DC1. Set the TR bit to inhibit the CRLF normally sent at termination.
BLOCK PAGE	1	Detection of RS. No termination on character count.	Inhibit echo. If first character received is a DC2, send a DC1. Set the TR bit to inhibit the CRLF normally sent at termination.

Table 2-22 is for terminals physically configured for HP block mode which can be set using the terminal's block mode key, or the appropriate escape sequence. To ensure that the driver and the terminal are in the same mode of operation, a CN 25B call must be issued after each terminal mode change. HP block mode transfers are only available if HP protocol has been selected and the terminal has been configured with its G and H straps enabled. Refer to the CN 34 and CN 25 sections of this manual.

If there is a driver mismatch of the terminal mode and the driver mode, the terminal will appear to lock up or hang. If the terminal is in block mode and the driver is in character mode, the

terminal will hang, waiting for a DC1/DC2/DC1 handshake sequence that is not issued by the driver in character mode. You can escape from this condition by doing a soft reset on the terminal. If the terminal is in character mode and the driver is in block page mode, entering an RS character (cntl ^) from the keyboard will terminate the read. In both cases, the read will eventually terminate if the LU has a non-zero timeout set. For this reason, it is strongly recommended that all terminals have timeouts assigned.

For your convenience, the HpCrt libraries HpCrtPageMode, HpCrtLineMode, and HpCrtCharMode are available and should be used to set the terminal and initiate the CN 25 call. These libraries are described in the *RTE-A • RTE-6 Relocatable Libraries Reference Manual*, part number 92077-90037. These subroutines do the required CN 25 call for the user.

AH: Auto-Home Bit

The auto-home bit is set to inform the driver that it should output a control sequence during the long handshake used by the terminal in *block page mode only*. The control sequence emitted is “Esc c Esc H”, which is the command to lock the keyboard and home the cursor (including transmit only fields).

The purpose of the home cursor is to remove the restriction that you must manually position the cursor to home before pressing the ENTER key. The purpose of the locked keyboard is to prevent data changing on the screen until the program permits it. The keyboard lock condition will persist until specifically cleared by the application program with an explicit write of an “Esc b” sequence. In addition, you can re-enable the keyboard by performing a soft terminal reset, if necessary.

The auto-home bit is effective only on user initiated transfers (when you press the ENTER key). It has no effect on program initiated transfers (the program sends “Esc d” to simulate the ENTER key), because such transfers do not use the terminal’s long handshake mode. Refer to the Sample Page Mode Application appendix and the Example Protocol Charts appendix of this manual for details on record blocking and escape sequences.

Special Status Read

```
CALL EXEC(1,ior(lu,3700b),status_buffer,length)
```

This read request reads the driver configuration and status words. It can be used to find out which driver is in use. The call is identified as special because the function bits are 37B and the length is 32 words (or 64 bytes). This call causes the RTE-A device drivers to request card status from the interface drivers. Table 2-23 contains a summary of the contents of each word.

Note that this call is I/O suspended if a read is pending or if the LU has been downed. The write-through-pending-read bit in an XLUEX call allows the dynamic request to complete. The read will then be reposted.

The 32 words of *status_buffer* are defined as follows:

- Words 1-3 contain the device driver name in ASCII. For example, DDC00.
- Word 4 contains the revision code of the device driver.
- Word 5 is the current driver type from the DVT tables and may be different than the device driver type because of the effects of a CN 30 call.

- Word 6 contains the address of the DVT. The formats of the I/O Tables are documented in the *RTE-A System Design Manual*, part number 92077-90013.
- Words 7-10 describe the interface driver in the same way as words 1-4 describe the device driver.
- Word 11 is the firmware revision code in the major/minor format. This is non-zero only for the intelligent I/O cards (8-channel/4-channel MUX), or a TELNET Pseudo Terminal LU. For TELNET, word 11 is zero unless the port is connected, then it will be the revision code of the TELNET monitor program.
- Words 12-14 contain the name of the primary interrupt program. If primary scheduling is currently enabled, the sixth byte is “E”, otherwise it is “D”.
- Words 15-17 contain the secondary interrupt program in the same format as words 12-14.
- Words 18-23 are the optional parameters required in the respective control calls to set the driver to the current conditions.
- Word 24 contains the address of the IFT.
- Words 25-28 are the optional parameters required in the respective control calls to set the driver to the current conditions.
- Words 29-31 are reserved.
- Word 32 returns the contents of DVT 20.

Table 2-23. Special Status Read Words

1	Device driver name
2	”
3	”
4	Device driver revision code
5	DVT word 6
6	DVT address
7	Interface driver name
8	”
9	”
10	Interface driver revision code
11	Firmware revision code
12	Primary interrupt program name
13	”
14	”
15	Secondary interrupt program name
16	”
17	”
18	Echo back of CN17 parameters
19	Echo back of CN22 parameters
20	Echo back of CN30 parameters
21	Echo back of CN31 parameters
22	Echo back of CN33 parameters
23	Echo back of CN34 parameters
24	IFT address
25..27	Echo back of CN42 parameters
28	Echo back of CN16 parameters
29..31	Reserved
32	Echo back of DVT20

Write Request

The calling sequence for a write request is:

```
CALL EXEC( 2 , cntwd , bufr , bufln [ , pram3 , pram4 ] )
```

bufr and *bufln*

The *bufr* parameter is the user's buffer containing the data to be written to the selected LU. *bufr* cannot be a FORTRAN character data type. Refer to the HpCrtSendChar subroutine section in the *Relocatable Libraries Reference Manual*, part number 92077-90037.

The *bufln* parameter is the length of the data in the user buffer. If the length parameter is positive, it indicates the number of words to be transferred (0 to 32767 words, 0..077777B). If the length parameter is negative, it indicates the number of characters to be transferred (1 to 32768 bytes, 177777B.. 100000B). *bufln* set to zero produces a CRLF sequence in ASCII mode; in binary mode, the write is suppressed. The trailing carriage return/line feed combination that is expected by most terminals is supplied by the driver, so the user buffer need contain only the data characters. If this action is not desired, it can be suppressed by setting the TR bit in the *cntwd*.

cntwd (Write Request Control Word)

The write request control word (*cntwd*) has the following form:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OS			Z	R	TR	FH	EC	V	BI	Device LU					

Figure 2-86. Write Request Control Word *cntwd*

OS: Operating system specific bits; see the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

Z: Write/write bit. If clear, *pram3* and *pram4* of the call are ignored.

If set, the control buffer indicated by *pram3* and *pram4* are written to the external device in ASCII mode before the user buffer is written, as described in the Read Request section. The normal data buffer, described by *bufr* and *bufln*, is transferred according to the rules established by the remainder of the bits in the control word.

R: Reserved; set to 0.

TR: Transparency and Binary bits. These bits alter the normal addition of a trailing Carriage Return/Line Feed (CRLF) and determine whether HP protocol is used if selected by a CN 34.

TR BI

0 0 The characters in the user buffer are sent to the given LU without special processing, except for the last character. If it is an underscore (octal 137), the underscore itself and the trailing CRLF are suppressed. In other words, for a request length of N characters where the Nth character is an underscore, N minus 1 characters are transferred. If the Nth character is not an underscore, N plus 2 characters are transmitted, the extra 2 characters being a trailing CRLF. HP protocol is used.

TR BI

0 1 Special processing described above does not occur. Thus, exactly N characters
1 0 are transferred with no additions or deletions. Underscore processing and transparent write processing as described above are not available on slaved printers (printers that are part of or connected to the terminal). HP protocol is used.

TR BI

1 1 Inhibits HP protocol for this write. The buffer described by *bufr* and *bufln* are transferred without the ENQ/ACK handshake that would normally occur after each 80 bytes of output. If over 80 bytes are transmitted, an ENQ/ACK handshake will be pending at the beginning of the next write. ENQ/ACK handshakes may be removed permanently with the appropriate change of protocol using a CN 34 call.
Special processing described above does not occur.

In the case of TELNET LUs, the binary bit is ignored. ENQ/ACK handshaking, if appropriate, is handled by the remote terminal/interface card combination.

Refer to the Example Protocol Charts appendix for an example of the use of the FH and BI bits in a write/write call to write to the graphics memory of an HP terminal.

FH: Force handshake bit. When in HP protocol mode, this bit causes an ENQ/ACK handshake to be executed before the user's data buffer transfer begins.

For TELNET LUs, the force handshake bit is ignored. Handshaking is handled by the local terminal/interface card combination.

Refer to the Example Protocol Charts appendix for an example of the use of the FH and BI bits in a write/write call to write to the graphics memory of an HP terminal.

EC: Echo bit. The echo bit is ignored on writes.

V: Printer Honesty mode bit. This bit is ignored for ports that are not configured as printers. If set, column 1 has no special meaning. If clear, column 1 is used for FORTRAN style carriage control.

Control Requests

The calling sequence for a control request is

```
CALL EXEC(3, cntwd[, pram1[, pram2, pram3]])
```

cntwd (Control Request Control Word)

The control request control word (*cntwd*) is shown in Figure 2-87.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OS				Function						Device LU					

Figure 2-87. Control Request Control Word *cntwd*

OS: these bits are defined in the *RTE-A Programmer's Reference Manual*, part number 92077-90007.

Function: contains the function code that defines the control action to be performed. Function control codes are referred to by the octal value, so the six bits allow for control codes 00 to 77 (octal).

If function codes other than the ones listed below are issued, the driver rejects the call as an illegal request, or accepts it with no action, depending on the state of the “nice” bit set in the CN 34 call. This also applies to illegal bits being set, in which case 15B is returned. Note that not specifying a function may cause a default CN command to be executed, depending on the driver.

The following is a list of the control functions:

Code	Function
06B *	Obtain dynamic status
11B	Line spacing/page eject
16B *	Configure baud rate generator (ID800/01 only)
17B *	Define terminator
20B	Define/enable primary program scheduling
21B	Disable program scheduling
22B	Set device timeout
25B	Read HP terminal straps into driver
26B *	Flush input buffers
30B *	Set port ID
31B *	Modem environment
32B *	Generate break
33B *	Configure driver responses
34B *	Set port protocol
35B *	Reset BRG (ID800/01 only)
40B	Define/enable secondary program scheduling
41B	Disable secondary program scheduling

* indicates the control functions that are performed by the interface driver. Some control functions are implemented in the device driver and others in the

interface driver. If a function is not implemented in either, the device driver rejects it or processes it by accepting it with no action. This is done at the device level to hold down system overhead.

Device LU: contains the LU number to control.

Function Code 6B: Dynamic Status

```
CALL EXEC(3,ior(lu,600b)[,pram1])
CALL ABREG(istatus,itlog)
CALL RMPAR(istat_array)
```

Function code 6B returns the requested port status in DVT6 and DVT16-19. The standard form of the EXEC 3 call is to set *pram1* equal to zero or omit it. *pram1* can be set to a non-zero value to return information from the interface driver to the device driver. Refer to Dynamic Status Special Forms, later in this section.

The ABREG routine, documented in the *Relocatable Libraries Reference Manual*, can be used to retrieve the contents of the A-Register (DVT 6) after the dynamic status call.

The RMPAR routine, documented in the *RTE-A Programmer's Reference Manual*, can be used to retrieve extended status. Five words of status are returned in the five-word array *istat_array*. The format of DVT6 and *istat_array* (DVT16-19) is as follows:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DVT6	AV		Device Type					EF	BR	EM	LD	OF	PF	TO	E		
<i>istat_array</i> {	DVT16	0								Error Code							
	DVT17	Length of Type Ahead Data Available															
	DVT18	Interface Card Status (according to <i>pram1</i>)															
	DVT19	IF Driver ID				Interface Driver Revision Code											
	DVT20	Undefined															

Note

The dynamic status request is treated as a normal I/O request (it is not buffered). The request is handled by the driver. If the device is already down, the request will not be completed until the device is UP'ed. For status information without the possibility of being suspended waiting on a down device, see the EXEC 13 request.

DVT6: Device Status

This status word is always returned in the A-Register after unbuffered requests. The A-Register can be accessed from high level languages via an ABREG call. The status field in DVT6 is updated by each call to the drivers.

A read clears the DVT6 error bits. A null binary read may be used to clear the bits, without any effect on the card.

The format of the status word in DVT6 is as follows:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT6	AV		Device Type						EF	BR	EM	LD	OF	PF	TO	E

- AV:** Device availability. These bits are used by the system for I/O control. The DS operator command can also be used to examine the availability.
- If AV is 00 the DVT is available for a new request to be initiated (the device is free to process a new request).
- If AV is 01 the associated device has been set down by the driver or the operator. New requests will be suspended on the downed device.
- If AV is 10 the device is busy processing an I/O request. New requests may be pending (that is, linked through word 1 of the DVT).
- If AV is 11 the device is down, but busy with a request (such as an abort request).
- Device Type:** Driver device type. This 6-bit value describes the type of device associated with the current DVT. Refer to Table 2-29 for device type values.
- EF:** End of file (set by CTU drivers only).
- BR:** Break character detected on received data line.
- EM:** End of medium (set by detection of EOT in Normal ASCII read).
- LD:** Line down; valid for modem lines after first connect. Also indicates a speed sense failure for ID400 and ID800/801, or that the LU is currently disconnected for Telnet LUs.
- OF:** Overflow error; the application is losing data.
- PF:** Parity or framing error; there was a bit error in the data.
- TO:** Timeout by the device driver.
- E:** Error. Set to 1 if any DVT16 error bits are set. Note that bits 6, 5, 3, 2, and 1 are cleared on each read request.

DVT16: Error Code (System and Driver Defined Errors)

DVT16 bits 0 through 5 return the error code if one occurs. Error code descriptions are listed in Table 2-24 and Table 2-25.

Note that driver defined errors are reported in decimal. The octal values are provided as an aid to those going into driver source; some drivers have the driver defined error codes coded in octal.

Table 2-24. DVT16 System Defined Errors

Decimal	Octal	Driver	Description
1	1	All	Illegal Request
2	2	All	Device Not Ready. Attempt to read or write with Line Down (LD) bit set.
3	3	All	Timeout
4	4	All	End of Tape (EOT)
5	5	All	Transmission Error
6	6	All	Write Protected
7	7	All	Address Error
8	10	All	Serial Poll Failure (HPIB)
9	11	All	Group Poll Failure (HPIB)
10	12	All	Fault (?)
11	13	All	Data Communication Error
12	14	All	Generation Error, table extension size insufficient.
13	15	All	Invalid Request (Bad Parameters). Attempt to perform illegal configuration in a control request, or to set invalid bits.

Table 2-25. DVT16 Driver Defined Errors

Decimal	Octal	Driver	Description
21	25	DDC00/1	No room in program schedule table.
22	26	DDC00/1	'internal problem'.
25	31	ID800	Control and status word mismatch, that is, control word was a command but a write complete status was sent back.
27	33	ID400	Initialization error. During initialization of card on powerup or powerfail recovery, the SFTFAIL bit never cleared indicating that the firmware failed it's internal self-test (romsum, ramcheck, or port error).
28	34	ID400	DMA timeout. Before a DMA transfer, the driver verifies that DMA has completed. If not yet completed, it will wait - this wait loop is what timed out. Possible hardware problem or the MCU is in self-test loop.
28	34	ID800	DMA timeout - DMA never completed.
29	35	ID400	SRQ timeout. During initialization of card on power up or power fail recovery, the initial data transfer is performed using programmatic I/O. The flag 30B was not set in the given time frame. Possible hardware problems or the MCU is in self-test loop.
29	35	ID800	Timeout while waiting for Flag 30B to become set.
30	36	ID400	VCP switch error. The user changed the value of the VCP break enable switch after initialization on powerup. The driver received a VCP entry when not expected.
31	37	ID800	Firmware and driver are incompatible - probably an older revision of the mux.
41	51	ID400	MCU's SCI detected an overrun. It either indicates a firmware bug (interrupts turned off too long) or a low-level hardware problem.
42	52	ID10X/ID80X	Read request length is greater than 16384 words.
43	53	ID400	No status word available. This indicates a mismatch in state between the driver/firmware/PINT bit. It indicates that the PINT bit was set on the MCU, but when requested, no status was available.
***	Note: All 50 series errors from ID400 are serious and cause a state reset of the firmware, aborting whatever was occurring at the time that the error was encountered.		
50	62	ID400	Undefined by firmware.
51	63	ID400	IS3- timeout in control word loop. Serious firmware error, control word aborted.
52	64	ID400	IS3- timeout in firmware's status word loop. Serious firmware error, status word aborted.

Table 2-25. DVT16 Driver Defined Errors (continued)

Decimal	Octal	Driver	Description
53	65	ID400	No read configuration during process character firmware routine.
54	66	ID400	Timeout during status word transfer of immediate transfer routine, used in CN 6 requests.
55	67	ID400	No read configuration available after MCU->CPU transfer.
56	70	ID400	No write configuration available after CPU->MCU transfer.
59	73	ID400	These errors are caused by status word collisions.

DVT17: Transmission Log

This status word is returned in the B-Register after unbuffered requests. See the read call description. On a dynamic status request, this status word will reflect 0 data words transferred by the request, or if FIFO buffer mode is enabled, the number of data bytes available.

DVT18: Interface Card Status

This word varies depending upon the type of interface card. For ID400 (the 4-channel MUX), the word contains the contents of the microprocessor's memory location (indicated by the lower byte of *pram1*) returned in the lower byte of DVT18. For ID800 (the 8-channel MUX), the word contains the contents of the microprocessor's memory location (indicated by *pram1*) returned in the lower byte of DVT18. For example:

```
CALL EXEC(3,iOr(LU,600b),1234)
```

will return the contents of memory location 1234 from the MUX firmware memory (with only the lower byte being significant).

DVT19: Interface Driver Information

This word contains the interface driver revision code in bits 12 through 0 and the interface driver ID in bits 15 through 13.

Table 2-26. DVT19 Interface Driver ID

Driver ID Code	Interface	Driver
000	12005 ASIC card	ID100
001	12005 ASIC card w/modems	ID101
010	A400 4-channel MUX	ID400
011	Telnet Pseudo Terminal Interface	IDZ00
100	12040D 8-channel MUX	ID800
101	12040D 8-channel MUX w/modems	ID801

DVT20: Driver Communication Area

DVT word 20 is described in the “Interface-Device Communication” section in this chapter.

Dynamic Status Special Forms

To facilitate communications between the device driver and the interface drivers in RTE-A, the dynamic status call optional parameter can be set to -1 . The format of DVT 18 is then changed to:

**Firmware revision code for ID400 or ID800;
TELNET revision code for IDZ00 if LU is connected.**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MS Byte of Rev. Code								LS Byte of Rev. Code							

Figure 2-88. ID400 or ID800 Revision Codes

When the optional parameter is set to -2 , the driver returns the constant 123456B in DVT words 17 and 18 for use in distinguishing Revision 5000 drivers from the previous drivers. Refer to the HpCrtlib section of the *Relocatable Libraries Reference Manual*.

When the optional parameter is positive, the corresponding MUX memory location is read.

Function Code 11B: Line Spacing/Page Eject

For hardcopy devices (as defined by the H bit in the CN 34 call), this control call emits a specified number of line feeds or performs a page eject, depending on the *NumLines* parameter. For CRT devices (the H bit clear), the *NumLines* parameter is ignored and a single line feed is always emitted.

```
CALL EXEC(3,ior(lu,1100b)[,NumLines])
```

emits the number of line feeds specified by a positive parameter. The default is one line. Some drivers limit *NumLines* to 128 line feeds.

```
CALL EXEC(3,ior(lu,1100b),-1)
CALL EXEC(3,ior(lu,1100b),-2)
```

Either of the above calls performs a page eject only on hardcopy devices. When the parameter is -1 , the form feed is conditional. A parameter of -2 performs an unconditional form feed. If backward compatibility is needed, all form feeds can be made unconditional by setting bit 10 in the CN 34 call.

Function Code 16B: Configure Baud Rate Generator

Function code 16B applies only to ID800. This control call reconfigures the baud rate generator to a range other than the pre-defined BRG setting (see CN 30 for default settings). This control request should be used only if the desired BRG range cannot be obtained from the CN 30 control call, and it must be used before any ports on a given BRG are initialized. If ports are already initialized on the same BRG, then the BRG must be reset before the CN 16 can be executed. See the CN 35 command to reset a BRG. Note that issuing a CN 16 affects all ports on a given BRG.

```
CALL EXEC(3,ior(lu,1600b),Brgrange)
```

will set a baud rate generator to a pre-configured setting for the CN 30 control code.

The *lu* parameter specifies the LU on a particular baud rate group that you want to reconfigure. Note that this effects the other LUs on that baud rate group.

The format of the *Brgrange* parameter is as follows:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRG	Reserved							BRG Range							

BRG:

- 0 Set Baud Rate Group 0
- 1 Set Baud Rate Group 1

The assignment of ports to a Baud Rate Group (BRG) is determined by jumpers in the cable connector hood.

BRG Range: These bits set a given baud rate group to one of the following ranges:

BRG Range	Possible Baud Rates
0B	Reserved
1B	1800 *
2B	134 *
3B	14,400
4B	9600 19,200 38,400 *
5B	4800 * 9600 * 19,200 *
6B	2400 4800 9600
7B	1200 2400 * 4800
10B	300 * 600 * 1200 *
11B	75 * 150 * 300
12B	-- -- 110 *
13B	600 * -- --

* If you specify a given baud rate using the CN 30B command, and the BRG is not assigned, the starred range is selected. For example, if you specify 9600 baud, range 5B is selected.

Function Code 17B: Define Termination Character

This control call allows you to redefine the terminator used in transparent ASCII reads.

```
CALL EXEC(3,ior(lu,1700b),Terminator)
```

where the *Terminator* parameter is defined as:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						NE	FI	E	Terminator						

Figure 2-89. Word Format for Code 17B Definable Terminator

- Reserved:** Reserved; set to 0.
- NE:** No-echo bit. The no-echo bit in EXEC read requests is used to determine if characters should be echoed as they are read.
- 0 Read requests are echoed.
 - 1 Read requests are not echoed.
- FI:** FIFO Terminator bit. If this bit is set, the termination character is enabled as a FIFO terminator, that is, when the termination character is received in the FIFO buffer, program scheduling occurs as described in the section on Function Code 20B. This is supported by drivers ID800/801 only.
- E:** Enable bit.
- 0 Transparent ASCII reads are terminated by a CR, as described in the “Read Request” section.
 - 1 All subsequent transparent ASCII reads terminate upon receipt of the character specified in bits 7 through 0. This also disables the CRLF that is normally output upon receipt of the terminator of an ASCII read.
- Terminator:** Termination character. These bits define the new transparent ASCII read termination character.
- For the ASIC card (driver ID100), the user-definable termination characters available are CR (15b) and RS (36b). To define other termination characters, a non-standard special character PROM is required. Refer to the appendix “Creating a Special Character PROM”.

You can use the CN 17 call to perform transparent ASCII reads that do not output a CRLF when the terminating CR is detected. If CRLF suppression is needed, issue the call with a parameter of 415B to set the termination character to 15B (Carriage Return).

Function Code 20B: Enable Program Scheduling

This control call enables program scheduling upon recognition of certain interrupting conditions. Those conditions vary depending on which driver and interface card are in use as explained below.

Program Scheduling Conditions

DDC00 performs program scheduling upon receipt of an unsolicited character or BREAK key.

An unsolicited character is any character that arrives when the driver is not executing a read. Some driver/interface combinations have FIFO buffer modes that effectively leave a read pending at all times. When this mode is invoked, the BREAK key is a more reliable way to schedule the prompt program. The BREAK key is a key present on most keyboards that causes a long space condition to be sent on the communications line, approximately 250 milliseconds. BREAK is a condition on the line, not a character. BREAK is always recognized, even if a read is pending.

A break condition can cause entry into VCP (Virtual Control Panel) for the system console unless the appropriate switch is set on the CPU card to disable it. It is suggested that the switch be in the *disable VCP* position except when the system is being used in a development mode.

When the program is scheduled, five parameters are passed:

- Parameter 1 = interrupting LU
- Parameter 2 = 0 if primary program, -1 if secondary
- Parameter 3 = DVT address
- Parameter 4 = 0
- Parameter 5 = 121212B

The program can retrieve the parameters with a RMPAR call.

Pass Program Name

To alter which program is scheduled upon interrupt, the following call can be used:

```
CALL EXEC(3,ior(LU,2000b),2hAB,2hCD,2hE )
```

This call passes the name of a new program in the three optional parameters. The name must be passed as three parameters as shown above, with blank padding if the name is less than five characters long. Note the use of only capital letters, since case folding is not performed. For example, to pass the name CX the following call would be used (020040B is the octal representation of two blanks):

```
CALL EXEC(3,ior(LU,2000B),2hCX,20040B,20040B)
```

The following calls are NOT valid:

```
CALL EXEC(3,ior(LU,2000b),2hCX)
CALL EXEC(3,ior(LU,2000b),'ABCDE')
CALL EXEC(3,ior(LU,2000b),5hABCDE)
CALL EXEC(3,ior(LU,2000b),5habcde)
```

The program must have been linked as a system utility and RP'd in the system session, either in the BOOTEX process, the link file, or online by the user.

RTE Compatibility: Pass Program Name

To facilitate portability between operating systems, it is suggested that the utility subroutine HpCrtSchedProg in HpCrt.lib be used in place of the CN 20 calls.

```
CALL HpCrtSchedProg(LU, 5hPRMR[ , 2hPR ] )
CALL HpCrtSchedProg(LU, 5hSECND, 2hSE )
```

To enable a program that is already known, either from the system generation or from a call to HpCrtSchedProg, the following form of the EXEC call can be used:

```
CALL EXEC(3,ior(LU,2000B))
```

Refer to the System Generation Considerations appendix for details about how the programs can be specified at system generation.

Function Code 21B: Disable Program Scheduling

```
CALL EXEC(3,ior(LU,2100B))
```

This call disables program scheduling enabled by function codes 20 and 40, as well as the system prompt. It does not change the name of the interrupt schedule program, so a subsequent function 20 or 40 call with no parameters will re-enable the same programs. Refer to the Program Scheduling appendix for a state diagram of the CN 20, 21, 40, and 41 calls. This call does not completely disable program scheduling if the LU is 1.

Disabling program scheduling is done in the device driver. The interface drivers still interrupt the system on program scheduling conditions. In particular, the system designer should be aware that the MUX interrupts on each character once the FIFO has filled. If it is desired that the MUX be used with a device that continuously sends data, it may be necessary to provide a program that is interrupt scheduled on FIFO full to issue a buffer flush.

Function Code 22B: Set Device Timeout

```
CALL EXEC(3,ior(LU,2200B),NewTimeout)
```

This call changes the value of the device timeout. The *NewTimeout* parameter is the number of centiseconds (100ths of a second) allowed for a read or a write to complete. If a nonzero value is passed, it is complemented and then used as a count-up timer. If *NewTimeout* is set to 1, the device timeout equals .01 seconds. Likewise, setting *NewTimeout* to 32767 sets the device timeout to 327.67 seconds. Even longer timeouts are possible, but they appear odd when expressed in decimal because of the lack of an unsigned integer data type. Numbers from -32768 through -1 produce timeouts in the range 327.68 seconds to 655.35 seconds.

A timeout value of 0 disables timeout processing, giving an infinite timeout. *This is not recommended.* Because read calls are blocking I/O, messages cannot be written to a terminal that has a read outstanding. In addition, a timeout ensures that the driver is allowed to detect and correct certain error conditions. When a read times out, the driver returns the status word, with bit 1 set, to the program. The status word is available in the A-Register, which can be accessed from high level languages via the ABREG call. All interactive programs should check this bit and take appropriate action upon timeout (usually they should loop back to re-prompt the user).

This control call has a similar effect as the CI TO command:

```
CI> TO,lu,NewTimeout
```

The old timeout value is stored in DVT17 and the B-Register by the CN 22 call. Although a program could use this to retrieve the old value for later restoration, the recommended procedure is to use the special status read instead.

End of operation status for a nonbuffered request is returned in the registers as follows:

A-Register = Device status found in DVT word 6 (see STAT1 section).

B-Register = Device status found in DVT word 17.

Function Code 25B: Read HP Terminal Straps

```
CALL EXEC(3,ior(lu,2500B))
```

This call must be issued each time the operating mode of an HP terminal is changed, to allow the driver to update the internal flags that indicate which protocol to use in a read call. To make this easier, the utility subroutines HpCrtPageMode, HpCrtLineMode, and HpCrtCharMode are available. Note that this EXEC call clears FIFO buffers. The contents of DVT20 are returned in the B-Register when this call is issued to an unbuffered LU, so the current terminal state can be determined.

Function Code 26B: Flush Input Buffers

```
CALL EXEC(3,ior(lu,2600B))
```

This function clears the input buffers on the interface card. If FIFO buffer mode is enabled, it clears any data that is being held on the MUX card. This has the same effect as re-issuing the “enter FIFO buffer mode” command. For all drivers that support FIFO mode, this call clears the hardware and software error bits.

Function Code 30B: Set Port Configuration

```
CALL EXEC(3,ior(lu,3000B),Portword)
```

This function establishes the logical connection between the LU and the physical device connected to the card. It is used by the MUX drivers to map LUs to the ports and to set the baud rate on the card when possible.

All configuration parameters can be specified at generation. The values of driver parameters 1 through 4 initialize the CN 17, 20, 30, 33, and 40 parameters. These values may be changed in the welcome file or at any time.

The specified configuration occurs on the first request to the interface driver with control requests (such as a read or write request) on a per LU basis. Note that this configuration does not occur when it is initiated from the remote device (such as a carriage return before initializing the port). In the welcome file, ports may be initialized by a “CN *lu* 6B”, “CN *lu* 11B”, or a write request to the particular port.

Note Function code 30B must be issued before any other request is sent to a specified port. Any requests sent to an LU prior to function code 30B are ignored by the driver, with the exception that CN 6 requests are processed, to aid in identifying the drivers (see CN 6, -2 option). Also, the LU number given should specify a unique device on the MUX. If the request is invalid, or if a conflict exists, an error is issued and the request rejected. The driver parameters can be set so that the port is pre-configured (refer to the System Generation Considerations appendix).

The *Portword* parameter has the following format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF		M	x	SB	PA	SS	S			PO					

where:

- CF: Number of data bits
- M: Modem control
- x: not set by the user, returned by the driver
- SB: Number of stop bits
- PA: Parity checking
- SS: Speed sense at logon
- S: Speed (baud rate)
- PO: Port number

The fields of the *Portword* parameter are defined in the following subsections. Note that if a call is made that specifies a field designated as not available (N/A) in the tables that follow, the driver rejects the control call with an error 15B return. When making a call to a driver for which the field is not valid, set it to zero.

CF: Character Framing

These bits indicate the number of data bits per character as follows:

Bits 15,14 – CF Character Framing data bits/character	ID800/01	ID400	ID100/01	IDZ00
00: 8 bits	ok	ok	ok*	ok
01: 7 bits	ok	N/A	ok*	N/A
10: 6 bits	ok	N/A	N/A	N/A
11: 5 bits	ok	N/A	N/A	N/A
<p>* Character framing can be specified on the ASIC card only if jumper W3 is in position B.</p> <p>If jumper W3 is in position A, character framing depends on bits 8 and 9 (parity sense) as follows:</p> <p>parity enabled → 7-bit character framing parity disabled → 8-bit character framing</p> <p>Refer to the <i>HP 12005B ASIC Installation and Reference Manual</i>, part number 12005-90002.</p>				

When using 5-bit data, the upper three bits must be set to zero. Otherwise, undesirable results may occur due to limitations in the 8-channel MUX hardware.

M: Modem Control

This bit indicates modem control as follows:

Bit 13 – M Modem Control	ID800	ID801	ID400	ID100	ID101	IDZ00
0: no modem control	ok	ok	ok	ok	ok	N/A
1: modem control	N/A	ok	ok	N/A	ok	ok

Note

Modem control is available on ports B and C only on the 4-channel MUX. Only modems that require control by the interface and are connected to these ports should be used with these calls. Do not attempt to use modem control with the 8-channel multiplexers unless the HP 37214 Modem Card Cage is installed.

Modem control must be set for TELNET LUs because TELNET uses HPMDM to control session connection/disconnection.

This call is used for modem control. The modem is controlled by the driver, using the RS-232 control leads. Short-haul modems are a form of line driver, and do not require modem control.

Once a port is declared to be a modem port, reads and writes are rejected unless the proper connection sequence occurs. A modem port responds to modem control line state changes, such as when the telephone rings or when the line is cut off, by scheduling the program HPMDM.

HPMDM schedules the logon or logoff programs as needed to maintain the port in an operational state. The name HPMDM is hardcoded into the device driver, thus, if you wish to use your own modem line supervisory program, it must be named HPMDM and be mutually exclusive of the HP program.

Because the drivers normally reject writes when the line is in the down state, sending commands to a modem that is capable of auto-dial must be done by setting the special write code of 37B in the function bits. This code causes the driver to assert DTR and RTS before writing, and ignores the states of CTS, DSR, and CD. It also has the effect of suppressing CRLF, so if they are required by the modem, the program must supply them.

x: (Don't Care Bit)

This bit is not set by the user; it is returned by the MUX driver. This field was used by previous drivers to inform the driver and the firmware of the distribution of the two baud rate clocks to the UARTs on the 8-channel MUX cards. The wiring in the connector hood determines this bit. There are several different cables for the MUX cards, with different wiring arrangements. It is also possible that the wiring may have been changed in the field. To alleviate this problem, ID800/801 determine the BRG wiring during initialization.

The sensed value is returned in this bit as follows (any user-supplied value is ignored):

Bit 12 – x Baud Rate Generator	ID800/01	ID400	ID100/01
0: generator 0 1: generator 1	ok ok	ok N/A	ok N/A

SB: Stop Bit Selection

These bits indicate the number of stop bits as follows:

Bits 11,10 – SB Stop Bit Selection	ID800/01	ID400	ID100/01	IDZ00
00: 1 stop bit	ok	ok	ok*	ok
01: 2 stop bits	ok	N/A	ok*	N/A
10: 1.5 stop bits	ok	N/A	N/A*	N/A
11: reserved				
* The ASIC card has a hardware jumper to select the number of stop bits; therefore, the driver ignores these bits.				

PA: Parity Check

These bits indicate parity checking as follows:

Bits 9,8 – PA Parity Checking	ID800/01	ID400	ID100/01	IDZ00
0x: no parity	ok	ok	ok	ok
10: odd parity	ok	N/A	ok*	N/A
11: even parity	ok	N/A	ok*	N/A
<p>* The ASIC card has a hardware jumper to select even/odd parity; therefore, the driver cannot change the parity sense when using the ASIC card. The driver does, however, read the setting to confirm that the parity requested by the user is the same as the hardware setting.</p> <p>If the user request matches the hardware setting, the request is accepted with no error, otherwise, the request is rejected.</p>				

If parity checking hardware is not available on a card, it can be generated or checked in the user buffer by using the subroutines HpCrtParityGen and HpCrtParityChk. Parity checking is *not* disabled by binary reads. If parity checking is enabled, it is the user programmer's responsibility to check the parity bit in the status. Standard HP utility programs, such as CI or FMGR, do not check for parity error.

SS: Speed Sense at Logon

If this bit is set, speed sense is performed at logon.

S: Speed (Baud Rate Selection)

These bits select the baud rate as follows:

Bits 6 thru 3 – S Baud Rate Selection	ID800/01	ID400	ID100/01	IDZ00
00b (0000)	N/A	N/A	ok	ok
01b (0001) 600 baud ¹	ok	N/A	} 4	N/A
02b (0010) 75 baud	ok	N/A		N/A
03b (0011) 110 baud	ok	N/A		N/A
04b (0100) 134.5 baud	ok	N/A		N/A
05b (0101) 150 baud	ok	N/A		N/A
06b (0110) 300 baud	ok	ok		N/A
07b (0111) 1200 baud	ok	ok		N/A
10b (1000) 1800 baud	ok	N/A		N/A
11b (1001) 2400 baud	ok	N/A		N/A
12b (1010) 4800 baud	ok	N/A		N/A
13b (1011) 9600 baud	ok	ok		N/A
14b (1100) 19.2K baud	ok	ok		N/A
15b (1101) 38.4K baud ²	ok	N/A		N/A
16b (1110) 14.4K baud ^{1 2 3}	ok	ok		N/A
17b (1111) speed sense	ok	ok	N/A	

1. Prior to Revision 5.19 of the MUX firmware, 50 baud rate and 115.4K baud were available as selections 1 and 16, respectively. At Revision 5.19, these were changed to produce 600 and 14.4K baud instead.
2. Because RS-232 connections have a recommended upper frequency limit of 19.2K baud, the RS-422 circuitry on the cards should be used for all higher settings.
3. For the 4-channel multiplexer, selection 16b yields 76.8K baud.
4. The ASIC card is capable of these baud rates. The rate is set by switches on the card rather than by the software, therefore, bits 6 through 3 should always be zero for ID100/01.

There are interactions between channels that share a given baud rate generator on the 8-channel MUX (ID800/01). The baud rates of all channels on a BRG must have baud rates in 1, 1/2, and 1/4 ratios. For example, if the highest baud rate on BRG 1 is 9600 baud, then 4800 and 2400 baud are also available for use by for other ports in the same group. If 19.2K is the highest, then 9600 and 4800 are available, and so on.

Default BRG Ranges (ID800/ID801 Only)

For ID800 and ID801, the BRG range is determined by the baud rate of the first port accessed on the BRG. If this automatic range selection is not desired, the CN 16 call may be used. Refer to the Function Code 16B section of this chapter.

For example, ports 4 and 5 are on the same BRG and need to be set to 4800 and 2400 baud, respectively. If port 5 is set to 1200 baud first, then port 4 cannot be set to 4800 baud, because the BRG range was automatically set to [300 600 1200]. This problem can be overcome by first setting the BRG range to [1200 2400 4800] using a CN 16. It will not necessarily be corrected by using speed sensing. This will cause the BRG range to be determined by the first port that was speed sensed.

A baud rate selection of 17B causes the port to speed sense. This allows the user's terminal to determine the appropriate baud rate. If the port is enabled for HP protocol, the driver sends an ENQ at each available baud rate, then waits half a second for an ACK to be returned.

When an ACK is received correctly, the baud rate is then known, and the driver sends a carriage return to the terminal. Other than random characters that may appear on the screen as a result of the terminal receiving the ENQ characters at the wrong baud rate, the process is automatic and the user is not aware that it is occurring.

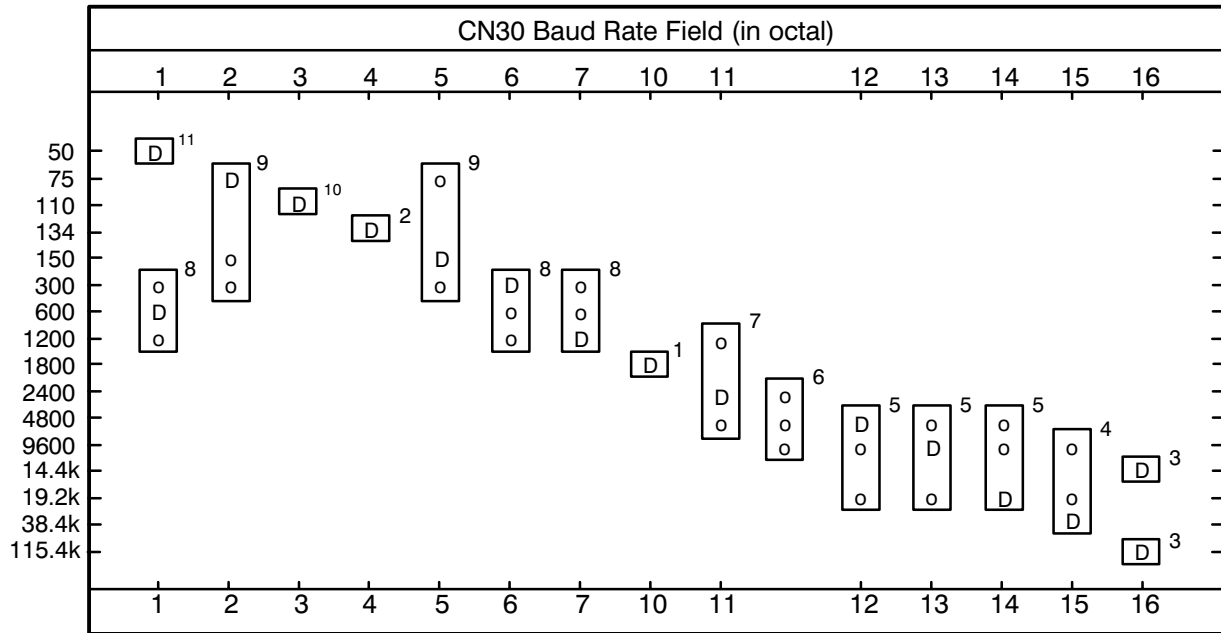
If the port is configured for non-HP protocol, the user must supply CR characters until the interface card can detect the appropriate baud rate. This may require several tries, as the card is sampling at one-second intervals. When the baud rate is detected, a CRLF is echoed to the terminal.

The new baud rate can be read by using the special status read. If the card is unable to detect or match the baud rate within the read timeout, the baud rate will be left at 17B and the line down bit is set in DVT6. (The programmer should ensure that the timeout is set to a reasonable value, such as two minutes, before issuing the speed sense command.) The programmer can detect speed sense failure from the CN 30 field of a special status read. Until the speed sense failure is corrected by another CN 30 call, all other requests are rejected. Asynchronous interrupts will cause speed sensing to occur again.

Note

The baud rates that are possible are limited by the hardware configuration of the cards (and in some cases the interface cable hoods) plus modems, line drivers, and so on, in the communications line, and the destination equipment. To operate at high baud rates the CRT and the driver must use a handshake protocol. Refer to the Function Code 34B section that follows.

Table 2-27. BRG Ranges



Note: The boxes enclose baud rates that are generated by the same BRG setting. The number to the upper right of each box is the BRG range selection (in decimal) for use in the CN 16 call. Within a given box, the default baud rate is shown by a 'D'. The other baud rates that are available with the BRG set to the given value are shown by 'o'.

For example, if the first port configured is set to 9600 baud by the CN 30 call (CN lu 30B 13), the default BRG selection will be 5. Once that port is configured, the other ports that share the BRG would be restricted to 4800, 9600, or 19.2K baud.

This is a reasonable default in most cases. However, assume you have 2400 baud terminals that must be connected to the same MUX. A CN 16 call could be included in the Welcome file (before the first port is configured) to force the BRG range to 6. This would allow the ports driven by that BRG to be configured to 2400, 4800, or 9600 baud instead.

PO: Port Number

The port number bits are defined as follows:

Bits 2 thru 0 – PO port number	ID800/01	ID400	ID100/01	IDZ00
000: port 0/A	ok	ok	ok	ok
001: port 1/B	ok	ok	N/A	N/A
010: port 2/C	ok	ok	N/A	N/A
011: port 3/D	ok	ok	N/A	N/A
100: port 4	ok	N/A	N/A	N/A
101: port 5	ok	N/A	N/A	N/A
110: port 6	ok	N/A	N/A	N/A
111: port 7	ok	N/A	N/A	N/A

Function Code 31B: Modem Environment

Function code 31 specifies how line open/close is to be done.

```
CALL EXEC(3,ior(lu,3100b),ModemControl)
```

where the *ModemControl* parameter is defined as follows:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A	C	Reserved													

A: Answer bit.

- 1 The interface driver schedules program HPMDM when ring indicator is detected to allow auto-answer.
- 0 The driver ignores the ring indicator signal.

C: Connect bit.

- 1 Connect modem, assert DTR and RTS lines, wait for DSR, CTS, and CD lines to be true.
- 0 Disconnect modem (hang up the phone).

Reserved: Reserved; set to 0.

A port is declared to be a modem port by setting bit 13 in the CN 30 call. Line state changes, such as when the telephone rings or when the line is cut off, will then schedule HPMDM. HPMDM schedules the logon or logoff programs as needed to maintain the port in an operational state. For a description of HPMDM and the modem control line protocols, refer to the Source Code Programs appendix of this manual.

Function Code 32B: Generate Break

Function code 32B allows for programmatic breaks. This function applies only to ID800/801 and ID400.

```
CALL EXEC(3,ior(lu,3200b)[,time])
```

where:

time is valid only for ID800/801 and specifies the break duration. For ID400, *time* is ignored and the transmit data line is held low for approximately 250 milliseconds.

For ID800/801, if *time* is equal to 0 or omitted, the transmit data line is held low for approximately 250 milliseconds. If *time* is not equal to zero, it specifies the number of “character times” for the break duration (“character times” is the time required to transmit one character at the current baud rate and framing). Values from 1 to 254 are legal.

Function Code 33B: FIFO Buffer Mode Control

FIFO buffer mode is available only on ID800/01 and ID400. The maximum FIFO is 1024 characters per port on the 8-channel MUX and 95 characters per port on the 4-channel MUX.

```
CALL EXEC(3,ior(lu,3300b),FIFOword)
```

where the *FIFOword* parameter is defined as follows:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	CH	SD	Reserved												

EN: FIFO mode enable.

- 0 No FIFO buffering. System attention is gained by pressing any key when a read is not pending. The remaining bits are ignored if bit 15 is not set.
- 1 Enable FIFO buffering. If data is received without a pending read, it is saved until the next read is posted, unless more data is received than can be held on the card. System attention is gained by pressing the **BREAK** key or when the termination character is detected with FIFO terminator enabled (bit 9 set in the CN 17 call).

CH: Character-by-character scheduling.

- 0 Accumulate characters, but take no action until a read is posted or a break is detected. (Must be zero for TELNET.)
- 1 Enable program scheduling for each character that is received.

SD: Save data on break.

- 0 Clear the buffer before scheduling the interrupt program when a break is detected.
- 1 Keep the FIFO buffer data, even when a break is detected.

Reserved: Reserved; set to 0.

A buffered read mode can be implemented for “single-character” style editors by the algorithm illustrated in Figure 2-90. The following is a step-by-step representation of the algorithm. It is a useful technique when you need full duplex I/O.

- 1 Enable FIFO mode.
- 2 Set timeout to a value short enough to satisfy the response time constraints.
- 3 Perform a dynamic status call (CN 6) to see if any characters are in the FIFO.
- 4a If characters are available, post a read for that many characters; process the data.
- 4b If no characters are available, post a read for one character.
- 5 When the read completes, check for a timeout.
- 6a If timeout, go to step 7.
- 6b If no timeout, 1 or more characters have been received; process the data.
- 7 See if outbound data is available.
- 8 If data is available, change timeout to a large enough value to allow the write to complete; write the data.
- 9 Perform any other processing necessary.
- 10 Go to step 2.

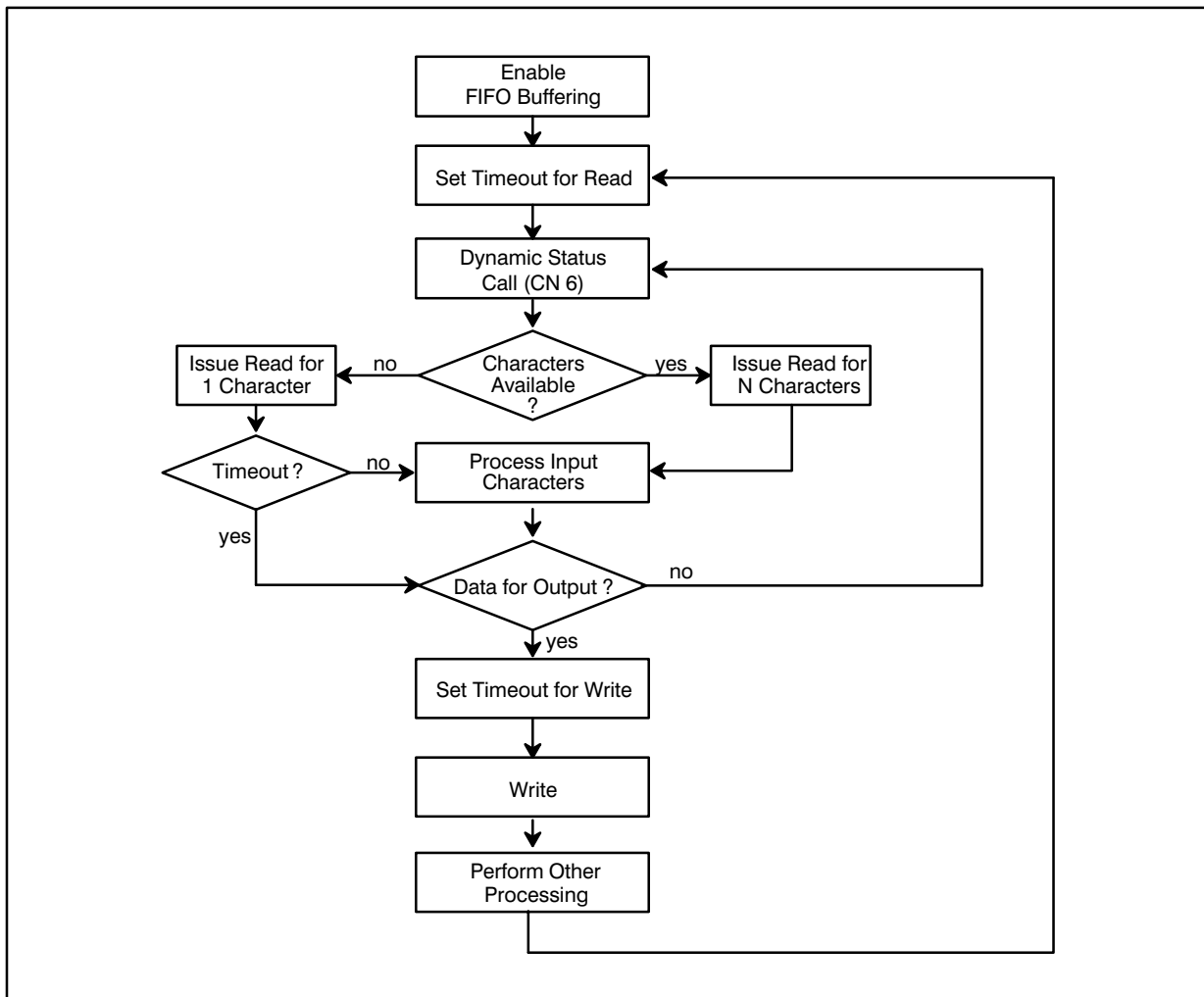


Figure 2-90. Buffered Read Mode Algorithm

Function Code 34B: Set Port Protocol

```
CALL EXEC(3,ior(lu,3400b),Protocol_Word)
```

where the *Protocol_Word* parameter is defined as:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		N	H	U	RD	P	Protocol								

R: Reserved; set to zero.

N: Nice bit.

0 Unimplemented control requests are rejected.

1 Unimplemented control requests are accepted but ignored.

Also, this bit can be set for DDC12 backward compatibility, that is, it allows the underscore (_) to suppress CRLF and still maintain carriage control in column 1. Bit 11 should be 0 in this case to select post-spacing. Refer to Table 2-28.

H: Hardcopy/spacing mode.

0 The port is connected to a CRT.

1 The port is connected to a hardcopy device. This enables page eject processing.

U: Unconditional page ejects.

0 CN 11,-1 page ejects are conditional.

1 CN 11,-1 page ejects are unconditional.

Setting bit 10 is required to ensure that form feeds are recognized. This causes the driver to treat CN,11B,-1 as an unconditional page eject.

RD: Return data on timeout.

Caution

The RD bit should not be set for interactive ports, as it leaves the possibility that destructive commands could be given to the system if the user inadvertently allows the terminal to timeout in the middle of a command. For example, if the user intended to enter the command "PU,@.lst", but was called away from the terminal, it could be seen as "PU,@".

0 No data is returned, transmission log is zero.

1 When a timeout is detected, the transmission log will indicate the number of data bytes in the buffer.

P: Printer/paper position.

0 Port is connected to an interactive terminal.

1 Port is connected to a printer and is, therefore, not interactive.

This bit changes the meaning of the H bit (bit 11) to be 0 = post spacing and 1 = pre-spacing. Carriage control is a matrix of capabilities defined by bits 8, 11, and 10 as described in Table 2-28.

Table 2-28. Carriage Control Capabilities

Bit								
8 (P-bit)	11 (H-bit)	10 (U-bit)	12 (N-bit)					
0	0	x	x	CRT	PRMPT	No CCTL	No CN 11	Valid underscore
0	1	0	x	TTY	PRMPT	No CCTL	Conditional CN 11	Valid underscore
0	1	1	x	TTY	PRMPT	No CCTL	Unconditional CN 11	Valid underscore
1	0	0	0	printer	no PRMPT	post-spacing CCTL	Conditional CN 11	Ignore underscore
1	0	0	1	printer	no PRMPT	post-spacing CCTL	Unconditional CN 11	Valid underscore
1	0	1	0	printer	no PRMPT	post-spacing CCTL	Conditional CN 11	Ignore underscore
1	0	1	1	printer	no PRMPT	post-spacing CCTL	Unconditional CN 11	Valid underscore
1	1	0	x	printer	no PRMPT	pre-spacing CCTL	Conditional CN 11	Ignore underscore
1	1	1	x	printer	no PRMPT	pre-spacing CCTL	Unconditional CN 11	Ignore underscore

CCTL means carriage control via column 1 as used by the FORTRAN formatter.
CCTL can be disabled by bit 7 of the control word in the write call.

In addition to the above, bits 8, 11, 6, and 1 determine the driver device type as described in Table 2-29.

Table 2-29. Device Type

8	11	6	1	Type	Description
0	0	0	0	0	non-HP CRT
0	0	x	1	5	HP CRT
0	0	1	x	5	HP CRT
0	1	1	0	6	non-HP hardcopy terminal (TTY)
0	1	x	1	6	HP hardcopy terminal (2635)
1	x	x	0	12	printer that does not use ENQ/ACK protocol
1	x	x	1	12	printer that uses ENQ/ACK protocol

Protocol: Bits 7 through 0 define the protocol as follows:

Bits 7 thru 0	ID800/01	ID400	ID100/01	Card Output Pacing	Card Input Pacing	Block Mode
000b: TTY	ok	ok	ok	None	None	no
001b: Xon/Xoff	ok	ok	N/A	Xon/Xoff	Xon/Xoff	no
002b: HP	ok	ok	ok	ENQ/ACK	DC1	yes
003b: HP Xon/Xoff	ok	ok	N/A	ENQ/ACK and Xon/Xoff	DC1 and Xon/Xoff	yes
004b: CPU-to-CPU	ok	ok	ok	None	None	no
010b: Hardware	N/A	ok	N/A	CTS	RTS	no
020b: Inverse HP	N/A	N/A	N/A	DC1	ENQ/ACK	no
040b: Half duplex	N/A	N/A	N/A	Special	Special	no
202b: Half HP	ok	ok	ok	None	DC1	yes
203b: Half HP Xon/Xoff	ok	ok	N/A	Xon/Xoff	DC1 and Xon/Xoff	yes

Note: Ports set to driver type 0 (non-HP CRT) can be forced to driver type 5 (permitting block mode optional) by setting bit 6. This can be done by ORing 100b with the desired protocol (bits 7-0). For example, the command: CN 1 34B 101B would configure a port to be driver type 5 with Xon/Xoff.

Caution The protocols listed in the previous table are those supported by Hewlett-Packard. They provide a solution for most configurations. Other bit combinations are possible, but may cause unpredictable and undesired results.

Reissuing the protocol command causes the port to enter the known state of transmit enable, for recovery from error situations where the CPU is waiting for an Xon from a device that will not send one.

The protocols available are:

TTY Protocol. No flow control handshaking, normal CRLF processing. This mode is primarily for interactive use with “dumb” terminals.

Xon/Xoff. This bi-directional protocol allows both the port and external devices to issue Xon and Xoff characters to pace the flow of incoming data. When the receiving end approaches a buffer full condition, the Xoff character is transmitted to suspend the flow of data. When buffer space is again available, the Xon character is transmitted to resume data flow. Excess Xon characters are discarded by the port in this mode, thus it is *not possible* to receive Xon characters as part of the data record.

It is possible to get into deadlock situations with this protocol. For example, assume that the port is connected to a line printer and the line printer runs out of paper. When the paper-out condition is detected, the printer transmits an Xoff to the CPU to stop the flow of data. If the user turns off the printer to reload paper, as is required by some printers, then turns it back on, expecting data to be printed, nothing will happen. The CPU is waiting for an Xon to give it permission to proceed.

Some printers have a front panel button that causes an Xon to be sent, while others send an Xon every time they are placed online. However, most do not have either feature. Because of this, if the port times out (beware of zero timeouts) during an Xon/Xoff write, the driver internally enables itself to the Xon state, and then goes down (but only if the hardcopy bit is set). If an Xon character is received, the driver calls \$UPIO to resume the transmission, or alternatively you can issue the UP command from another console.

In FIFO buffer mode, the interface card sends an Xoff when 15 bytes of buffer storage remain. A second Xoff is sent when only 10 bytes remain. An Xon is sent when the buffer level drops below 20 characters available.

The MUX card suspends output when it receives an Xoff. If it then receives a BREAK (which may indicate that the user wants to schedule the CM program), it reacts as if an Xon had been received. If the MUX did not do this, the prompt could not be displayed to the user.

HP PROTOCOL. Read pacing is by DC1 and DC1/DC2/DC1 software handshakes. Write pacing is by ENQ/ACK software handshakes. See the example protocol charts appendix for further information.

HP-Xon/Xoff. This type of protocol is a combination of HP and Xon/Xoff protocols. When this mode is enabled, the Xon/Xoff characters can cause interactions with EDIT/1000. Therefore, alternative characters must be used in place of control-S and control-Q. See the *EDIT/1000 User's Manual*, part number 92074-90001.

CPU-to-CPU. This protocol has no handshake; flow control must be done by higher level software. The advantage of this protocol over TTY is that it disables echoes at all times, overriding the echo bit in the read control word, and it does not append a line feed on write completion or generate a CRLF on read completion.

For normal ASCII reads, backspace and delete are processed as usual, except that no characters are echoed. Although this mode can be used to drive terminals that use local echo and auto-line feed (sometimes erroneously called "half-duplex"), it is primarily intended for non-interactive ports, as when driving "black boxes" or for terminal emulation.

Hardware Handshake (ID400 only). Logically similar to Xon/Xoff, but implemented with the RTS/CTS modem lines. This cannot be used on ports that have modem control enabled, as the definitions of the use of the RTS and CTS lines are in conflict. On the 4-channel MUX, only ports B and C have the modem control lines needed for this mode.

The external device can suspend output from the interface card by lowering the CTS line when it is about to fill its input buffers. It should be able to accept at least eight characters of additional data after setting CTS low. When CTS is asserted (taken high), the output will resume. If CTS is held low too long, such as when a printer runs out of paper, a timeout will occur that sets the LU down. When the CTS line is asserted in this case, the driver performs a call to \$UPIO to set the LU up again and restart the output. Note that it is possible for the same data to be transmitted twice in this case.

ID400 sets RTS low when 10 bytes of FIFO buffer remain, and sets it true when the buffer is emptied.

Half HP + Xon/Xoff. This protocol disables the ENQ/ACK portion of HP protocol and enables Xon/Xoff.

Half HP. This selection is used to disable the ENQ/ACK handshake of the HP protocol while preserving the D1 pacing on read requests. It is used with satellite links, statistical multiplexers, and some high speed modems.

The protocols that have been defined but not implemented are:

Inverse HP. This protocol is the logical inverse of the standard HP ENQ/ACK DC1/DC2 protocol. It is used to emulate an HP terminal to another HP system.

Half-Duplex. This protocol is used with half-duplex modems and terminals. It implements the line turnaround delays and specialized protocols that are used on high-speed half-duplex communication links.

ENQ/ACK Handshake Details

The following chart shows the algorithm used for the ENQ/ACK handshakes to pace writes in HP mode.

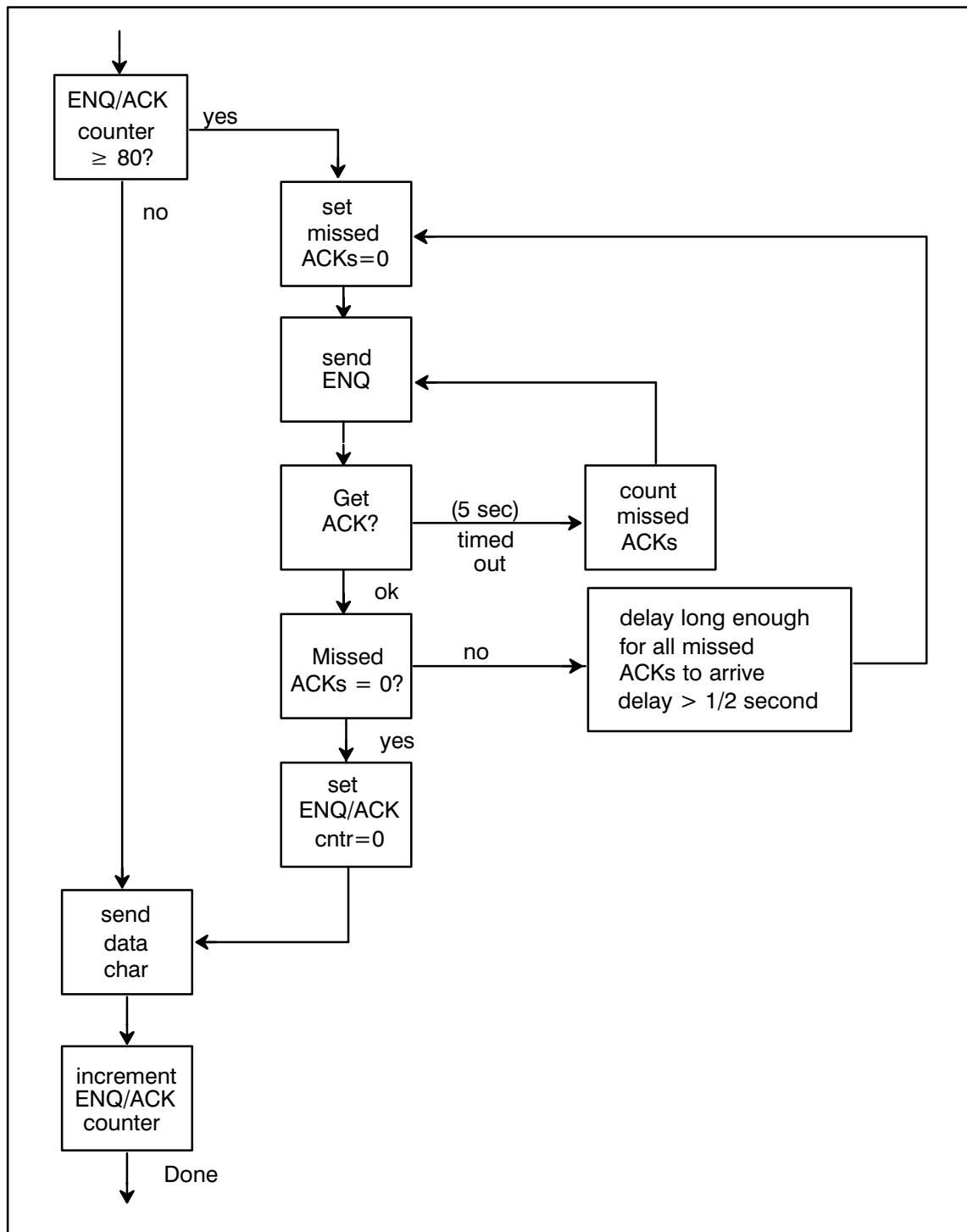


Figure 2-91. ENQ/ACK Handshaking

Function Code 35B: Reset a Baud Rate Generator

```
CALL EXEC(3,ior(lu,3500b),BRG_Control)
```

where the *BRG_Control* parameter is defined as:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														BRG	

BRG: 0 – Reset BRG #0
 1 – Reset BRG #1

This control call allows a programmatic reset of a baud rate generator (BRG). The current values assigned to the BRG are cleared, allowing the BRG to be set to a pre-configured range by a CN 16 call. The hardware is not physically reset until the BRG is assigned a new value.

Function Code 40B: Enable Program Scheduling

```
CALL EXEC(3,ior(lu,4000b)[,2hAB,2hCD,2hE ])
```

This function enables a program to be scheduled in the event that the primary program specified by function 20 is busy. Other than that, the description of Function Code 20 applies.

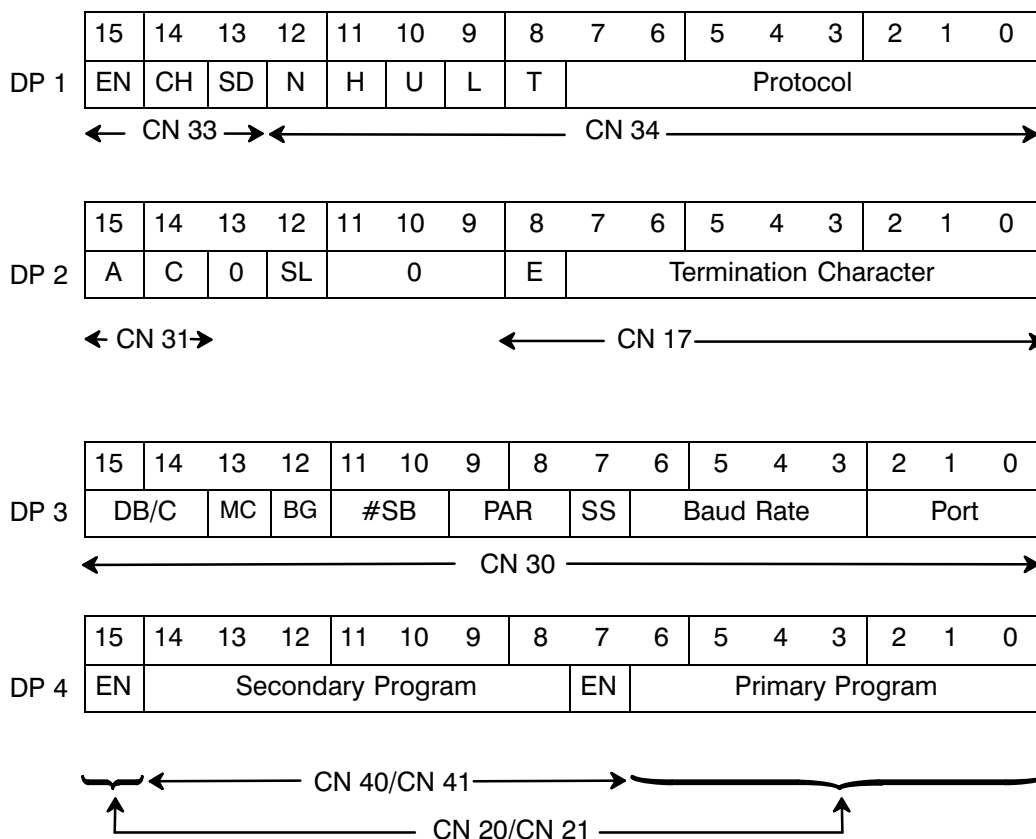
Function Code 41B: Disable Program Scheduling

```
CALL EXEC(3,ior(lu,4100b))
```

This function is used to disable the program specified by Function Code 40. Refer to the Program Scheduling appendix.

Device/Interface Driver Communication

This section describes the device driver to interface driver communication in RTE-A. The following four words are Driver Parameter words 1 through 4. For individual bit definitions, refer to the description of the appropriate CN function code in this chapter.



DVT 20 has 9 bits for communication between the device driver and the interface driver. The bit definitions of DVT 20 are as follows:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT20	I	BL	PG	SD	SP	0	0	MB	0	SH	Device Priority					

- 0:** Reserved; will be zero.
- I:** First entry bit.
This bit is set by the generator and cleared by the interface driver upon first entry when it has completed error checks.
- BL:** Block mode bit.
This bit is set if the driver senses the terminal to be in block mode during a CN 25 call (driver type 5 only).

- PG:** Page mode bit.
This bit is set if the driver senses the terminal to be in page mode during a CN 25 call (driver type 5 only).
- SD:** Slave device bit.
This bit is set for the DVTs that refer to a slave device connected to the terminal. DDC01 sets this bit (device type 20).
- SP:** Schedule program bit.
This bit is used by the interface driver to request program scheduling by the device driver because of a break or a modem line change. The reason for the schedule is passed in DVT 19 as follows:

DVT 19 = 1 for an incoming call
DVT 19 = 2 for an asynchronous disconnect

- MB:** Multibuffer bit.
This bit indicates a number of requests have been built in the DVT extension by the device driver.

- SH:** Schedule HPMDM bit.
The interface driver wants to schedule HPMDM because of a modem line change. DVT 19 contains one of the following:

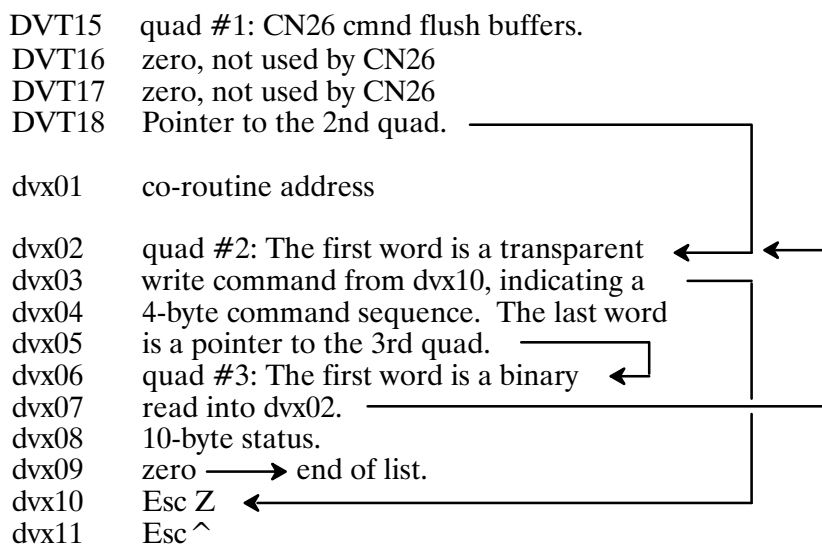
DVT 19 = -1 for AUTOR scheduling
DVT 19 = 1 for an incoming call
DVT 19 = 2 for an asynchronous disconnect
DVT 19 = 3 for connect request timeout
DVT 19 = 4 for connect request completion

Device Priority:

This bit indicates the priority assigned to the DVT for linking purposes on the IFT. Default linking is FIFO (priority ignored) unless the interface driver changes its Q bit (IFT 3) to specify priority linking.

Multibuffer Linked List Structure

The MB bit is used to inform the interface driver that the device driver has built a linked list of requests in the DVT and DVT extension. A multibuffer quad is a 4-word structure. For read and write commands, the first word is the command, the second word is the buffer address, and the third word is the request length. For control commands, words 2 and 3 are parameters, if needed by the command. The fourth word is a pointer to the next quad, or zero to terminate the list. An example usage from the CN25 call processing is as follows:



The third quad can read into the DVT extension at word 2 even though it is 5 words long and will overlay the list due to the quad being copied up to DVT words 15..18 by the time the read is executed.

HP Character Set

					←000-037B→		←040-077B→		←100-137B→		←140-177B→	
					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits				Col.	0	1	2	3	4	5	6	7
7	6	5	4	3	2	1	Row					
0	0	0	0	0	NUL	DLE	SP	0	@	P		p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(8	H	X	h	x
1	0	0	1	9	HT	EM)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	
1	1	0	1	13	CR	GS	-	=	M]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Example: The representation for the character "K" (column 4, row 11) is

Bit	7	6	5	4	3	2	1
Binary	1	0	0	1	0	1	1
Octal	1	1	3				

Note: *Depressing the Control Key while typing an upper case letter produces the corresponding control code on most terminals. For example, Control-H is a back space.

Table A-1. Hewlett-Packard Character Set for Computer Systems

This table shows Hewlett-Packard's implementation of ANS X3.4-1968 (USASCII) and ANS X3.32-1973. Some devices may substitute alternate characters from those shown in this chart (for example, Line Drawing Set or Scandinavian font). Consult the manual for your device.

The left and right byte columns show the octal patterns in a 16 bit word when the character occupies bits 8 to 14 (left byte) or 0 to 6 (right byte) and the rest of the bits are zero. To find the pattern of two characters in the same word, add the two values. For example, "AB" produces the octal pattern 040502. (The parity bits are zero in this chart.)

The octal values 0 through 37 and 177 are control codes. The octal values 40 through 176 are character codes.

Decimal Value	Octal Values		Mnemonic	Graphic ¹	Meaning
	Left Byte	Right Byte			
0	000000	000000	NUL	N _U	Null
1	000400	000001	SOH	S _H	Start of Heading
2	001000	000002	STX	S _X	Start of Text
3	001400	000003	EXT	E _X	End of Text
4	002000	000004	EOT	E _T	End of Transmission
5	002400	000005	ENQ	E _Q	Enquiry
6	003000	000006	ACK	A _K	Acknowledge
7	003400	000007	BEL		Bell, Attention Signal
8	004000	000010	BS	B _S	Backspace
9	004400	000011	HT	H _T	Horizontal Tabulation
10	005000	000012	LF	L _F	Line Feed
11	005400	000013	VT	V _T	Vertical Tabulation
12	006000	000014	FF	F _F	Form Feed
13	006400	000015	CR	C _R	Carriage Return
14	007000	000016	SO	S _O	Shift Out } Alternate
15	007400	000017	SI	S _I	Shift In } Character Set
16	010000	000020	DLE	D _L	Data Link Escape
17	010400	000021	DC1	D ₁	Device Control 1 (X-ON)
18	011000	000022	DC2	D ₂	Device Control 2 (TAPE)
19	011400	000023	DC3	D ₃	Device Control 3 (X-OFF)
20	012000	000024	DC4	D ₄	Device Control 4 (TAPE)
21	012400	000025	NAK	N _K	Negative Acknowledge
22	013000	000026	SYN	S _Y	Synchronous Idle
23	013400	000027	ETB	E _B	End of Transmission Block
24	014000	000030	CAN	C _N	Cancel
25	014400	000031	EM	E _M	End of Medium
26	015000	000032	SUB	S _B	Substitute
27	015400	000033	ESC	E _C	Escape ²
28	016000	000034	FS	F _S	File Separator
29	016400	000035	GS	G _S	Group Separator
30	017000	000036	RS	R _S	Record Separator
31	017400	000037	US	U _S	Unit Separator
127	077400	000177	DEL	■	Delete. Rubout ³

Table A-1. Hewlett-Packard Character Set for Computer Systems (continued)

This table shows Hewlett-Packard's implementation of ANS X3.4-1968 (USASCII) and ANS X3.32-1973. Some devices may substitute alternate characters from those shown in this chart (for example, Line Drawing Set or Scandinavian font). Consult the manual for your device.

The left and right byte columns show the octal patterns in a 16 bit word when the character occupies bits 8 to 14 (left byte) or 0 to 6 (right byte) and the rest of the bits are zero. To find the pattern of two characters in the same word, add the two values. For example, "AB" produces the octal pattern 040502. (The parity bits are zero in this chart.)

The octal values 0 through 37 and 177 are control codes. The octal values 40 through 176 are character codes.

Decimal Value	Octal Values		Character	Meaning
	Left Byte	Right Byte		
32	020000	000040		Space, Blank
33	020400	000041	!	Exclamation Point
34	021000	000042	"	Quotation Mark
35	021400	000043	#	Number Sign, Pound Sign
36	022000	000044	\$	Dollar Sign
37	022400	000045	%	Percent
38	023000	000046	&	Ampersand, And Sign
39	023400	000047	'	Apostrophe, Acute Accent
40	024000	000050	(Left (opening) Parenthesis
41	024400	000051)	Right (closing) Parenthesis
42	025000	000052	*	Asterisk, Star
43	025400	000053	+	Plus
44	026000	000054	,	Comma, Cedilla
45	026400	000055	-	Hyphen, Minus, Dash
46	027000	000056	.	Period, Decimal Point
47	027400	000057	/	Slash, Slant
48	030000	000060	0	} Digits, Numbers
49	030400	000061	1	
50	031000	000062	2	
51	031400	000063	3	
52	032000	000064	4	
53	032400	000065	5	
54	033000	000066	6	
55	033400	000067	7	
56	034000	000070	8	} Digits, Numbers
57	034400	000071	9	
58	035000	000072	:	Colon
59	035400	000073	;	Semicolon
60	036000	000074	<	Less Than
61	036400	000075	=	Equals
62	037000	000076	>	Greater Than
63	037400	000077	?	Question Mark

Table A-1. Hewlett-Packard Character Set for Computer Systems (continued)

Decimal Value	Octal Values		Character	Meaning
	Left Byte	Right Byte		
64	040000	000100	@	Commercial At
65	040400	000101	A	} Uppercase Letters
66	041000	000102	B	
67	041400	000103	C	
68	042000	000104	D	
69	042400	000105	E	
70	043000	000106	F	
71	043400	000107	G	
72	044000	000110	H	
73	044400	000111	I	
74	045000	000112	J	
75	045400	000113	K	
76	046000	000114	L	
77	046400	000115	M	
78	047000	000116	N	
79	047400	000117	O	
80	050000	000120	P	
81	050400	000121	Q	
82	051000	000122	R	
83	051400	000123	S	
84	052000	000124	T	
85	052400	000125	U	
86	053000	000126	V	
87	053400	000127	W	
88	054000	000130	X	
89	054400	000131	Y	
90	055000	000132	Z	
91	055400	000133	[Left (opening) Bracket
92	056000	000134	\	Backslash. Reverse Slant
93	056400	000135]	Right (closing) Bracket
94	057000	000136	^ ↑	Caret. Circumflex: Up Arrow ⁴
95	057400	000137	_ ←	Underline: Back Arrow ⁴

Note 1: This is the standard display representation. The software and hardware in your system determine if the control code is displayed, executed, or ignored. Some devices display all control codes as “ ”, “@”, or space.


Note 2: Escape is the first character of a special control sequence. For example, ESC followed by “J” clears the display on a HP2640 terminal.

Note 3: Delete may be displayed as “_”, “@”, or space.

Note 4: Normally, the caret and underline are displayed. Some devices substitute the up arrow and the back arrow.

Note 5: Some devices upshift lower case letters and symbols (‘ through “Blank”) to the corresponding upper case character (@ through ^). For example, the left brace would be converted to a left bracket.

Table A-1. Hewlett-Packard Character Set for Computer Systems (continued)

Decimal Value	Octal Values		Character	Meaning
	Left Byte	Right Byte		
96	060000	000140	`	Grave Accent ⁵ 
97	060400	000141	a	
98	061000	000142	b	
99	061400	000143	c	
100	062000	000144	d	
101	062400	000145	e	
102	063000	000146	f	
103	063400	000147	g	
104	064000	000150	h	
105	064400	000151	i	
106	065000	000152	j	
107	065400	000153	k	
108	066000	000154	l	
109	066400	000155	m	
110	067000	000156	n	
111	067400	000157	o	
112	070000	000160	p	
113	070400	000161	q	
114	071000	000162	r	
115	071400	000163	s	
116	072000	000164	t	
117	072400	000165	u	
118	073000	000166	v	
119	073400	000167	w	
120	074000	000170	x	
121	074400	000171	y	
122	075000	000172	z	
123	075400	000173	{	Left (opening) Brace ⁵
124	076000	000174		Vertical Line ⁵
125	076400	000175	}	Right (closing) Brace ⁵
126	077000	000176	~	Tilde, Overline ⁵

- Note 1: This is the standard display representation. The software and hardware in your system determine if the control code is displayed, executed, or ignored. Some devices display all control codes as “ ”, “@”, or space.
- Note 2: Escape is the first character of a special control sequence. For example, ESC followed by “J” clears the display on a HP2640 terminal.
- Note 3: Delete may be displayed as “_”, “@”, or space.
- Note 4: Normally, the caret and underline are displayed. Some devices substitute the up arrow and the back arrow.
- Note 5: Some devices upshift lowercase letters and symbols (‘ through ~) to the corresponding uppercase character (@ through ^). For example, the left brace would be converted to a left bracket.

Table A-2. HP 7970B BCD-ASCII Conversion

Symbol	BCD (Octal Code)	ASCII Equivalent (Octal Code)	Symbol	BCD (Octal Code)	ASCII Equivalent (Octal Code)
(space)	20	040	@	14	100
!	52	041	A	61	101
"	37	042	B	62	102
#	13	043	C	63	103
\$	53	044	D	64	104
%	57	045	E	65	105
&	† ¹	046	F	66	106
'	35	047	G	67	107
(34	050	H	70	110
)	74	051	I	71	111
*	54	052	J	41	112
+	60	053	K	42	113
,	33	054	L	43	114
-	40	055	M	44	115
.	73	056	N	45	116
/	21	057	O	46	117
0	12	060	P	47	120
1	01	061	Q	50	121
2	02	062	R	51	122
3	03	063	S	22	123
4	04	064	T	23	124
5	05	065	U	24	125
6	06	066	V	25	126
7	07	067	W	26	127
8	10	070	X	27	130
9	11	071	Y	30	131
:	15	072	Z	31	132
;	56	073	[75	133
<	76	074	\	36	134
=	17	075]	55	135
>	16	076	↑	77	136
?	72	077	←	32	137

Note 1: †The ASCII code 046 is converted to the BCD code for a space (20) when writing data onto a 7-track tape.

Quick Reference Guide

The following is quick reference information that includes the EXEC I/O calling sequences for each driver in this manual. The formats of the parameters in the EXEC request are also included, but the bits are not defined explicitly; refer to the related section for the detailed bit definitions.

DD*00

DD*00 Terminal Device Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln[, pram3[, pram4]])
Write Request: CALL EXEC(2, cntwd, bufr, bufln[, pram3[, pram4]])

cntwd >>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	Z	0	TR	PR	EC	0	BI	Device LU					

>> DD*00

Control Request: CALL EXEC(3, cntwd[, pram1[, pram2[, pram3[, pram4]]]])

cntwd >>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	Z	Function Code						Device LU					

>> DD*00

Function Codes

- 00 Clear and Reset Device
- 06B Dynamic Status
- 11B Line Skipping/Formfeed
- 20B Enable Primary Program Scheduling
- 21B Disable Primary Program Scheduling
- 22B Set System Request Timeout
- 23B Disable Asynchronous Interrupts
- 25B Read Terminal Straps
- 27B Set User Request Timeout
- 40B Enable Secondary Program Scheduling
- 41B Disable Secondary Program Scheduling
- 42B Get Character
- 44B Modify Configuration Word
- 45B Modify Trigger Character

Status Request: CALL EXEC(13, cntwd, stat1, stat2, stat3, stat4)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	0			Z	0				LU							

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<	AV		Device Type = 00,05					ED	0	EM	0			E		
stat2 <<	AV		Interface Type = 00				0		I/O Select Code							

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0																
stat3 <<	First Word of the Driver Parameter Area															
stat4 <<	Second Word of the Driver Parameter															

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1																
stat3 >>	Buffer to Return the Driver Parameter Area															
stat4 >>	Length of Buffer Described by stat3															

DD*00 Driver Parameter Area

Parameter	Parameter Description
+1	Terminal Configuration
+2	Suppress Space Flag
+3	Trigger Character before Read
+4	Device Timeout Value
+5	Scheduled Primary Program Name (Characters 1 and 2)
+6	Scheduled Primary Program Name (Characters 3 and 4)
+7	Scheduled Primary Program Name (Characters 5 and 6)
+8	Optional Parameter (Primary Program)
+9	Scheduled Secondary Program Name (Characters 1 and 2)
+10	Scheduled Secondary Program Name (Characters 3 and 4)
+11	Scheduled Secondary Program Name (Characters 5 and 6)
+12	Optional Parameter (Secondary Program)

DD*12

DD*12 HP-IB Line Printer Driver

Write Request: CALL EXEC(2, cntwd, bufr, bufln)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	0	TR	0	CC	0	LU						

cntwd >> >>DD*12

Control Request: CALL EXEC(3, cntwd[, pram1])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	Function Code						LU					

cntwd >> >>DD*12

Function Code

- 00 Clear and Reset Device
- 11B Line Skipping/Formfeed

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0						LU					

cntwd >>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		Device Type = 12					0		EM		0			E	
AV		Interface Type = 37					0		0		I/O Select Code				

stat1 <<
stat2 <<

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0															
stat3 << First Word of the Driver Parameter Area															
stat4 << Second Word of the Driver Parameter															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1															
stat3 >> Buffer to Return the Driver Parameter Area															
stat4 >> Length of Buffer Described by stat3															

DD*12 Driver Parameter Area

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP1		HP-IB Device Address													
DVP2		0													

DDC12 2608S HP-IB Line Printer Driver

Write Request: CALL EXEC(2 ,cntwd ,bufnr ,bufln)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				HON	VFC	0	V	0	LU						

Control Request: CALL EXEC(3 ,cntwd[,pram1])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				Control Code						LU					

Control Codes

- 00 Clear Control
- 06 Dynamic Status Request
- 11B Paper Motion Request
- 15B Change Character Sets
- 16B VFC Reset
- 20B Self Test
- 21B Left Margin Selection
- 30B Select Print mode

Status Request: CALL EXEC(13 ,cntwd ,stat1[,stat2[,stat3[,stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0						LU					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		Device Type = 12					0		EOM		0			E	
AV		Interface Type = 37					0		I/O Select Code						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
First Word of the Driver Parameter Area = DVP1															
Second Word of the Driver Parameter Area = DVP2															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Buffer to Return the Driver Parameter Area															
Length of Buffer Described by stat3															

DDC12

DDC12 Driver Parameter Area

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP1<<	HP-IB Address															
DVP2<<	1															
DVP3<<	TRN	0										Print Mode				
DVP4<<	APE	0													SS	
DVP5<<	PRG							Character Set 2				Character Set 1				
DVP6<<	0										Line Density					
DVP7<<	0							Left Margin Definition								

DVP1: HP-IB Address of printer

DVP2: Always 1 for DDC12

DVP3: TRN = Transparency Mode: 1 = on; 0 = off
 Print Mode =
 00B Standard-size (default)
 01B Double-size
 02B Graphics
 03B-17B Reserved (defaults to 00B)

DVP4: APE = Auto page eject flag 1 = on; 0 = off
 SS = Suppress spacing 1 = on; 0 = off

DVP5: PRG = Character set languages programmed
 Character Set 1, Character Set 2 = Language codes

DVP6: Line Density; 6 = 6 LPI; 8 = 8 LPI

DVP7: Left Margin Definition

DDC12 Extended Status

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XSTAT(1)<<	x		0						Error Code							
XSTAT(2)<<	Transmission Log															
XSTAT(3)<<	PR	x						OL	RDY	x			STF	NP	PF	

XSTAT(1): x = undefined

Error Codes

01	IO-RQ	Illegal Request
02	IO-NR	Not Ready
03	IO-TO	Timeout
05	IO-TE	Transmission Error
12B	IO-GE	Generation Error (also reported as IO-12)
21B	IO-21	Printer Failure
63B	IO-63	Power Failure with Automatic Restart

XSTAT(3): PR =Protocol Error 1 = on; 0 = off
 x =Undefined
 OL =Printer is Online 1 = on; 0 = off
 RDY=Printer Ready 1 = on; 0 = off
 STF =Self-test Failure 1 = on; 0 = off
 NP =Not Printing 1 = on; 0 = off
 PF =Power Fail without Recovery 1 = on; 0 = off

DD*20

DD*20 CTU Device Driver

Read Request: CALL EXEC(1 ,cntwd ,bufr ,bufln)

Write Request: CALL EXEC(2 ,cntwd ,bufr ,bufln)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd >>	0	NB	UE	0	0	TR	0	0	0	BI	Device LU					

>> DD*20

Control Request: CALL EXEC(3 ,cntwd [,pram1])

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd >>	0	NB	UE	Z	Function Code						Device LU					

>> DD*20

Function Codes

- 00 Clear and Reset Device
- 01 Write End of File (EOF)
- 02 Backspace One Record
- 03 Forward Space One Record
- 04 Rewind Tape
- 05 Rewind Tape
- 06 Dynamic Status
- 10B Write EOF if not previously written or not at load point
- 13B Forward Space One File
- 14B Backspace One File
- 26B Write End of Data (EOD)
- 27B Locate Absolute File on Tape

Status Request: CALL EXEC(13 ,cntwd ,stat1 [,stat2 [,stat3 [,stat4]]])

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd >>	0			Z	0						Device LU					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<	AV		Device Type = 20						EF	DB	EM	BM	SE	WP	OF	E
stat2 <<	AV		Interface Type = 00						0		I/O Select Code					

For Z=0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat3 <<	First Word of the Driver Parameter Area															

For Z=1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat3 >>	Buffer to Return the Driver Parameter Area															
stat4 >>	Length of Buffer Described by stat3															

DVP Format for DD*20

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP1	1 = Left CTU Requested								2 = Right CTU Requested							

DD*23

DD*23 HP-IB Magnetic Tape Device Driver

Read Request: CALL EXEC(1 , cntwd , bufr , bufln)
 Write Request: CALL EXEC(2 , cntwd , bufr , bufln)

cntwd>>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	0	TR	0	0	0	BI	Device LU					

>> DD*23

Control Request: CALL EXEC(3 , cntwd)

cntwd>>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0		TR	0			Device LU						

>> DD*23

← Function Code →

Function Codes

Code	Code	
TR off	TR on	
00	20B	Rewind
01	21B	Write End of File (EOF)
02	22B	Backspace One Record
03	23B	Forward Space One Record
04	24B	Rewind Tape (same as 00)
05	25B	Rewind/Standby
06	26B	Dynamic Status
12B	32B	Write Gap
13B	33B	Forward Space One File
14B	34B	Backspace One File

Status Request: CALL EXEC(13 , cntwd , stat1 [, stat2 [, stat3 [, stat4]]])

cntwd>>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0			Device LU								

stat1 <<

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		Device Type = 23					EOF	DB	EOM	BOM	SE	WP	OF	E	

stat2 <<

AV		Interface Type = 37					I/O Select Code				
----	--	---------------------	--	--	--	--	-----------------	--	--	--	--

For Z=0

stat3 <<

First Word of the Driver Parameter Area															
---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

stat4 <<

Second Word of the Driver Parameter Area															
--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

For Z=1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat3</i> >>	Buffer to Return the Driver Parameter Area															
<i>stat4</i> >>	Length of Buffer Described by <i>stat3</i>															

DVP Format for DD*23

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP1	HP-IB Mag Tape Address															
DVP2	Unit Number + 1 in Upper Byte															
DVP3	Retry Counter															

DD*24

DD*24 HP-IB Magnetic Tape Device Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln)

Write Request: CALL EXEC(2, cntwd, bufr, bufln)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	NB	UE	x	SR	TR	x	x	x	BI	Device LU					

cntwd>> >>DD*24

Control Request: CALL EXEC(3, cntwd[, pram1[, pram2[, pram3[, pram4]]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0		TR					Device LU					

cntwd>> >>DD*24

Function Codes

- 00 Rewind
 - 01 Write File Mark
 - 02 Backspace One Record
 - 03 Forward Space One Record
 - 04 Rewind Tape (same as 00)
 - 05 Rewind and Go Offline
 - 06 Dynamic Status
 - 12B Write Gap
 - 13B Forward Space One File
 - 14B Backspace One File
 - 15B Select Density *pram1* = 800 or 1600
 - 17B Diagnostics Request (set Z-bit = 1)
- pram1, pram2, pram3, pram4* carry instructions and data

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0						Device LU					

cntwd>>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
First Word of the Driver Parameter Area															
Second Word of the Driver Parameter Area															

For Z=0
stat3<<
stat4<<

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Buffer to Return the Driver Parameter Area															
Length of Buffer Described by <i>stat3</i>															

For Z=1
stat3>>
stat4>>

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat1</i> <<	AV		Device Type = 24B					EOF	0	EOM	BOM	SE	WP	OF	E	
<i>stat2</i> <<	AV		Interface Type = 37B					0	I/O Select Code							

DVP Format for DD*24

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVP1 <<	HP-IB Mag Tape Address															
DVP2 <<	Default Density															
DVP3 <<	ID Bytes From Tape Drive															
DVP4 <<	DO	0	IR	CR	Class			Reserved (0)								
DVP5 <<	Controller Error Code								Queued Request Count							

DDQ24

DDQ24 SCSI Tape Device Driver

Read Request: CALL EXEC(1 , cntwd , bufr , bufln)

Write Request: CALL EXEC(2 , cntwd , bufr , bufln)

Z-Read Request: CALL EXEC(1 , cntwd , bufr , bufln , command , commandln)

Z-Write Request: CALL EXEC(2 , cntwd , bufr , bufln , command , commandln)

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	0			SD	BI	LU Number						

Control Request: CALL EXEC(3 , cntwd[, parm1[, parm2]])

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	0	UE	0	SubFunction					LU Number						

Subfunction Codes

- 0B Rewind
- 1B Write “n” File Mark(s)
- 2B Backward space “n” record(s)
- 3B Forward space “n” record(s)
- 4B Rewind
- 5B Rewind and go off line (unload tape)
- 6B Dynamic Status Request
- 7B Write “n” Set Mark(s)
- 10B Backward space “n” Set Mark(s)
- 11B Forward space “n” Set Mark(s)
- 13B Forward space “n” file(s)
- 14B Backward space “n” file(s)
- 15B Set tape density, enable/disable compression
- 16B Enable/disable driver request sense after check condition

where “n” is a positive integer from 1 to 32767, inclusive.

Logical Unit Status (DVT Word 6)

DVT word 6 Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV	Device type = 24B					EOF	DB	EOT	BOT	R	WP	OF	E		

DDQ24 Driver Error (DVT Word 16)

<u>Dec.</u>	<u>Octal</u>	<u>Error and Solution</u>
0	0B	= Normal request completion.
1	1B	= Illegal request. Check the syntax of your EXEC request call. The EXEC call could have illegal parameters or specify an unused LU or an unimplemented command. Retry the request after you check it.
3	3B	= Device timeout. Use the TO command to increase the timeout period.
6	6B	= Device write protected. Remove the tape's write protection.
7	7B	= Address error. Ensure that the SCSI address is correct.
10	12B	= Tape fault. Check the tape for any problems.
12	14B	= Insufficient driver table space generated. Generate more driver table space.
22	26B	= Incompatible cartridge. Use the correct type of tape cartridge.
23	27B	= Positioning error detected. First try ejecting, then reloading the tape. If this does not solve the error, a hardware error is indicated.
24	30B	= Hardware error. Check all connections, and ensure that tape drive is functioning correctly.
25	31B	= Unknown error. This message should not appear. If it does, decode the error by checking DVT 19 which contains specific SCSI error codes.
26	32B	= End of Data.
62	76B	= Device busy.
63	77B	= Driver retry request. This message can appear only if the Control Word user error bit (UE bit 13) is set. Control Word bit 13 enables the user to interpret errors and take action on them instead of allowing the system to handle errors. If you receive this message, check your request's syntax and try the request again.

All codes of 28 decimal (34B octal) or higher are interface driver error codes, except for 63 (77B), which indicates a driver retry request.

DDQ24

DDQ24 SCSI Status and Transaction Status (DVT Word 18)

DVT word 18 contains the SCSI status and transaction status of device driver DDQ24. The SCSI status returned in \$DV18 (bits 7 – 0) is as follows:

xx00000xB = No error.
xx00001xB = Check condition.
xx00010xB = Condition met.
xx00100xB = Busy.
xx01000xB = Intermediate.
xx01010xB = Intermediate condition met.
xx01100xB = Reservation conflict.
xx10001xB = Command terminated.
xx10100xB = Queue full.

The transaction status returned in \$DV18 (bits 15 – 8) is as follows:

0 = No error.
1 = SCSI error. Check the sense code and additional sense code to identify the specific error.
2 = Selection/reselection timeout. Check to ensure that the device address is correct.
3 = Data error. This indicates a hardware error.
4 = Bus parity error. This indicates a hardware error.
5 = Reset.
6 = Illegal bus free. This indicates a hardware error.
7 = Abort request. The driver has sent an abort request.
8 = Illegal request. This indicates an illegal script type.
9 = Firmware error.

DD*30 Disk Device Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln, track, sector)

Write Request: CALL EXEC(2, cntwd, bufr, bufln, track, sector)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	0	0	UE	0	77B				Device LU				>>DD*30			

Control Request: None

Status Request: CALL EXEC(13, cntwd, stat1 [,stat2 [,stat3 [,stat4]])

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	0		Z	0				Device LU								

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<	AV		Device Type = 30-32				0		DB	EM	0	SE	0	E		
stat2 <<	AV		Interface Type = 37				0		I/O Select Code							

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0																
stat3<<	First Word of the Driver Parameter Area															
stat4<<	Second Word of the Driver Parameter Area															

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1																
stat3 >>	Buffer to Return the Driver Parameter Area															
stat4 >>	Length of Buffer Described by stat3															

DD*30 Driver Parameter Area

Parameter	Parameter Description
+1	HP-IB Address
+2	Unit Number
+3	Starting Head Number
+4	Starting Cylinder
+5	Number of Spares
+6	Number of Tracks
+7	Number of Sectors/Track
+8	Number of Surfaces

DDM30

DDM30 Disk Device Driver

Read Request: CALL EXEC(1 ,cntwd ,bufnr ,bufln ,track ,sector)

Write Request: CALL EXEC(2 ,cntwd ,bufnr ,bufln ,track ,sector)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0				UE				0			
								77B				Device LU			

>>DDM30

Control Request: None

Status Request: CALL EXEC(13 , cntwd , stat1 [,stat2 [,stat3 [,stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0				Z				0			
								Device LU							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<		AV		Device Type = 32,34,35				0		DB	EM	0	SE	0	E
stat2 <<		AV		Interface Type = 37				0		I/O Select Code					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0															
stat3<<				First Word of the Driver Parameter Area											
stat4<<				Second Word of the Driver Parameter Area											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1															
stat3>>				Buffer to Return the Driver Parameter Area											
stat4>>				Length of Buffer Described by stat3											

DDM30 Driver Parameter Area

Parameter	Parameter Description
+1	HP-IB Address
+2	Unit Number
+2	Starting Head Number
+4	Starting Cylinder
+5	Number of Spares
+6	Number of Tracks
+7	Number of Sectors/Track
+8	Number of Surfaces

DDQ30 SCSI Disk Device Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln, track, sector)

Write Request: CALL EXEC(2, cntwd, bufr, bufln, track, sector)

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	NT	0					LU Number					

Control Request:: CALL EXEC(3, cntwd, p1, p2, p3, p4)

Control Word Bits:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	0	UE	0	Function Code					LU Number						

Function Codes

16B	Enable/disable driver request sense after check condition
76B	Copy RTE block information

DDQ30 Driver Error (DVT Word 16)

<u>Dec.</u>	<u>Octal</u>	<u>Error and Solution</u>
0	0B	= Normal request completion.
1	1B	= Illegal request. Check the syntax of your EXEC request call. The EXEC call could have illegal parameters or specify an unused LU or an unimplemented command. Retry the request after you check it.
2	2B	= Device not ready. The drive is offline, but media is present in the drive. If this occurs, first try unloading, then reloading the tape. Second, send a SCSI Command Descriptor Block (CDB) to load the tape. If this does not work, ensure that the device is correctly connected, not busy, and correctly generated into the system.
3	3B	= Device timeout. Use the TO command to increase the timeout period.
5	5B	= Transmission error.
6	6B	= Device write protected. Remove the tape's write protection.
7	7B	= Address error. Ensure that the SCSI address is correct.
10	12B	= Disk fault. Check the disk for any problems.
12	14B	= Insufficient driver table space generated. Generate more driver table space.
21	25B	= Wrong media.
62	76B	= Device busy.

All codes of 28 decimal (34B octal) or higher are interface driver error codes.

DDQ30

DDQ30 SCSI Status and Transaction Status (DVT Word 18)

DVT word 18 contains the SCSI status of device driver DDQ30. The SCSI status returned in \$DV18 (bits 7 – 0) is as follows::

xx00000xB = No error.
xx00001xB = Check condition.
xx00010xB = Condition met.
xx00100xB = Busy.
xx01000xB = Intermediate.
xx01010xB = Intermediate condition met.
xx01100xB = Reservation conflict.
xx10001xB = Command terminated.
xx10100xB = Queue full.

The transaction status returned in \$DV18 (bits 15 – 8) is as follows:

0 = No error.
1 = SCSI error. Check the sense code and additional sense code to identify the specific error.
2 = Selection/reselection timeout. Check to ensure that the device address is correct.
3 = Data error. This indicates a hardware error.
4 = Bus parity error. This indicates a hardware error.
5 = Reset.
6 = Illegal bus free. This indicates a hardware error.
7 = Abort request. The driver has sent an abort request.
8 = Illegal request. This indicates an illegal script type.
9 = Firmware error.

DD*33 CS/80 Disk Device Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln, track, sector)

Write Request: CALL EXEC(2, cntwd, bufr, bufln, track, sector)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>				0	0	UE	0	77B				Device LU				>>DD*33

Control Request:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0	0	0	0	Function Code				LU			

Function Codes:

- 26B Set user request timeouts.
- 76B Alter driver parameter table.

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0	Z	0				Device LU					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<		AV	Device Type = 26, 33				Status Bits								
stat2 <<		AV	Interface Type = 37				0	I/O Select Code							

Device type: Disk = 33B, CTD = 26B

Status Bits:

- 0 Severe Error.
- 1 Channel Errors. Set for any Reject error, or DMA length error. If this bit is set, the severe bit (bit 0) is also set.
- 2 Not ready (unit not ready for access). If this bit is set, severe bit is also set.
- 3 Fault (a Fault error has occurred). If this bit is set, severe bit is also set.
- 4 Uninitialized Medium. If this bit is set, severe bit is also set.
- 5 EOF/EOV. If this bit is set, severe bit is also set.
- 6 Unrecoverable Data/(recoverable data), set for uncorrectable or marginal errors. Severe bit set only for unrecoverable data.
- 7 Write protected volume. Severe bit set only if write failed.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
stat3>>																First Word of the Driver Parameter Area			
stat4>>																Second Word of the Driver Parameter Area			

DD*33

DD*33 Driver Parameter Area

	Disk Parameter Description				CTD Parameter Description			
	15	9	8	0	15	9	8	0
DP1	HP-IB Address				HP-IB Address			
DP2	Unit #		Volume #		Unit #		Volume #	
DP3	MSB				C	Unit # Disk		Volume #
DP4	Starting Block Address 3 Words				Starting Block Address 2 Words			
DP5	LSB							
DP6	Number of Tracks				Address of First Cache Block			
DP7	Number of 128-Word Sector/Block							
DP8	Reserved				Reserved			
C = 1 if Cartridge Tape is Cached								

Extended Status:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XSTAT(1)<<DVT16<<	x		0						Error Code							
XSTAT(2)<<DVT17<<	Transmission Log = 0															
XSTAT(3)<<DVT18<<	Extended Status #1 (Error Class 1-4)															
XSTAT(4)<<DVT19<<	Extended Status #2															

Error Codes

- 00 Normal Request Completion
- 01 Illegal Request (track sector error)
- *02 Device Not Read
- *03 Device Timeout
- *05 Transmission Error
- *06 Device Write Protected
- *10B Device Fault (hardware)
- *77B Driver Request Retry

* = code not seen unless bit 13 (UE) is set

ID*00/01 ASIC Interface Card Drivers

Read Request: CALL EXEC(1, cntwd, bufr, bufln, pram3[, pram4])

Write Request: CALL EXEC(2, cntwd, bufr, bufln, pram3)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	BB	NB	UE	x	SP	x	x	x	CRLF	BI	LU				>>ID*00

Control Request: CALL EXEC(3, cntwd[, pram1])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	BB	NB	UE	0	Function Code						LU				>>ID*00

Function Codes

- 00 Clear and Reset Device
- 06 Dynamic Status
- 23B Disable Asynchronous Interrupts
- 43B Enable/Disable Error Checking

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	0		Z	0				LU							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<	AV		Device Type = xx				Status						E		
stat2 <<	AV		Interface Type = 00				0	0	I/O Select Code						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0	First Word of the Driver Parameter Area														
stat3 <<	Second Word of the Driver Parameter Area														
stat4 <<															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1	Buffer to Return the Driver Parameter Area														
stat3 >>	Length of Buffer Described by stat3														
stat4 >>															

ID*00/01

ID*01 Enhancements to ID*00 for Modem Operation

Additional Function Codes

- 31B Activate for Modem Environment
- 32B Deactivate Modem Environment

IDM00 8-Channel ASYNC Multiplexer Interface Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln, pram3[, pram4])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>	BB	NB	UE	x	x	x	KP	x	x	BI	LU					>>IDM00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
pram3>>	EOT	x	x	EM	x	x	x	x	x	x	x	x	x	x	x	>>IDM00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
pram4>>	Prompt Character							Number of Characters								>>IDM00

Write Request: CALL EXEC(2, cntwd, bufr, bufln)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>	BB	NB	UE	x	SP	x	x	HS	CRLF	BI	LU					>>IDM00

Control Request: CALL EXEC(3, cntwd [, pram1])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>	BB	NB	UE	0	Function Code						LU					>>IDM00

Function Codes

- 06 Dynamic Status
- 23B Control Program Scheduling
- 26B Flush Input Buffer
- 30B Set Port ID
- 33B Configure Driver Responses
- 37B Set Read Type

ID*27

ID*27 Disk Interface Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln, track, sector)

Write Request: CALL EXEC(2, cntwd, bufr, bufln, track, sector)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0	UE	Z	77B				LU				

Control Request: None

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0	Z	0				LU					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<		AV	Device Type = 30B				0		EM	SE	0		E		
stat2 <<		AV	Interface Type = 27B				0		I/O Select Code						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<		x	0				Error Code								
DVT17<<		Transmission Log = 0													
DVT18<<		Reserved	Status				Unit # of Disk								
DVT19<<		I	Reserved				0	WP	0	FLT	0	SE	DNR	0	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0		First Word of the Driver Parameter Area													
stat3<<		Second Word of the Driver Parameter Area													
stat4<<															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1		Buffer to Return the Driver Parameter Area													
stat3 >>		Length of Buffer Described by stat3													
stat4 >>															

ID*27 Driver Parameter Area

Parameter	Parameter Description
+1	Drive Address
+2	Unit Number
+3	Starting Head Number
+4	Starting Cylinder
+5	Reserved
+6	Number of Tracks
+7	Number of Sectors/Track
+8	Number of Surfaces

ID*36

ID*36 PROM Storage Module Interface Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln, track, sector)

Write Request: CALL EXEC(2, cntwd, bufr, bufln, track, sector)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	0	0	UE	0	77B						LU					

Control Request: None

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>	0			Z	0						LU					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<	AV		Device Type = 36B						0						E	
stat2 <<	AV		Interface Type = 36B						0		I/O Select Code					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<	x	x	0						Error Code							
DVT17<<	Transmission Log															
DVT18<<	Card Status															
DVT19<<	Undefined															

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0	First Word of the Driver Parameter Area															
stat3<<	Second Word of the Driver Parameter Area															
stat4<<																

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1	Buffer to Return the Driver Parameter Area															
stat3 >>	Length of Buffer Described by stat3															
stat4 >>																

ID*36 Driver Parameter Area

Parameter	Parameter Description
+1	0
+2	0
+3	0
+4	0
+5	0
+6	Number of Tracks (64)
+7	Number of Blocks/Track (4)
+8	Blocking Factor (1)

ID*37

ID*37 HP-IB Interface Card Driver

Note For high-speed devices, special cable-length restrictions apply. Refer to the *12009A HP-IB Interface Reference Manual*, part number 12009-90001, for a description of the high-speed data cable restrictions.

Read Request: CALL EXEC(1 ,cntwd ,bufr ,bufln [,pram3])
 Write Request: CALL EXEC(2 ,cntwd ,bufr ,bufln [,pram3])

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>	BB	NB	UE	0	Subfunction						Device LU						>>ID*37

Control Request: CALL EXEC(3 ,cntwd [,pram1 [,pram2 [,pram3 [,pram4]]]])

Function Codes

- 00 Selected Device Clear (SDC)
- 06 Device Dynamic Status (Serial Poll)
- 16B Set REN True
- 17B Go to Local (GTL)
- 20B SRQ Program Scheduling
- 21B Disable SRQ Program Scheduling
- 22B Set Interface Driver Timeout
- 23B Enable Parallel Poll Interrupt
- 24B Set Device Address
- 27B Group Execute Trigger (GET)
- 30B Disable Code 20B from Allowing SRQ Interrupts
- 31B Restore Ability of Code 20B to Allow SRQ Interrupts
- 40B Enable Parallel Poll Program Scheduling (PPE)
- 41B Disable Parallel Poll Program Scheduling (PPD)

Read Request: CALL EXEC(1 ,cntwd ,bufr ,bufln [,pram3 [,pram4]])
 Write Request: CALL EXEC(2 ,cntwd ,bufr ,bufln [,pram3 [,pram4]])

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>	BB	NB	UE	Z	Subfunction						BUS LU						>>ID*37

Control Request: CALL EXEC(3, cntwd[, pram1[, pram2[, pram3[, pram4]]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>				BB	NB	UE	Z	Function Code				BUS LU				>>ID*37

Function Codes

- 00 Clear and Reset Device
- 06 Dynamic Status
- 16B Set REN True (Remote Enable)
- 17B Go to Local (GTL)
- 23B Parallel Poll Interrupt
- 25B Local Lockout (LLO)
- 27B Group Execute Trigger (GET)
- 40B Enable Parallel Poll Program Scheduling
- 41B Disable Parallel Poll Program Scheduling
- 51B Bail Out (Abort)

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0		Z	0				Device LU				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<		AV		Device Type = xx				Status				E			
stat2 <<		AV		Interface Type = 00				0		I/O Select Code					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0															
stat3<<															
stat4<<															
First Word of the Driver Parameter Area															
Second Word of the Driver Parameter Area															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1															
stat3>>															
stat4>>															
Buffer to Return the Driver Parameter Area															
Length of Buffer Described by stat3															

ID*37 Driver Parameter Area

Any words returned from the driver parameter area to stat3/stat4 or to the status buffer described by stat3/stat4, are device dependent.

ID*50

ID*50 GPIO/Parallel Interface Card Driver

Read Request: CALL EXEC(1, cntwd, bufr, bufln[, pram3])

Write Request: CALL EXEC(2, cntwd, bufr, bufln[, pram3])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>				BB	NB	UE	0	x = 0				LU				>>ID*50

Control Request: CALL EXEC(3, cntwd, pram1[, pram2[, pram3[, pram4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>				BB	NB	UE	x	Function Code				LU				>>ID*50

Function Codes

- 00 Clear and Reset Device
- 06 Dynamic Status of Card
- 20B Enable Program Scheduling
- 21B Disable Program Scheduling
- 40B Configure Card Control Word

Status Request: CALL EXEC(13, cntwd, stat1[, stat2[, stat3[, stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0		Z	0				LU				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<		AV	Device Type				Status				E				
stat2 <<		AV	Interface Type = 50				0	0	I/O Select Code						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0		stat3<<													
First Word of the Driver Parameter Area															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=1		stat3 >>													
Buffer to Return the Driver Parameter Area															
stat4 >>		Length of Buffer Described by stat3													

ID*50 Driver Parameter Area

Any words returned from the driver parameter area to stat3/stat4 or to the status buffer described by stat3/stat4, are device dependent.

ID*52 Parallel Interface Intercomputer Communications Driver

Read Request: CALL EXEC(1 ,cntwd ,bufr ,bufln)
 Write Request: CALL EXEC(2 ,cntwd ,bufr ,bufln [,pram1])
 Write/Read Request: CALL EXEC(2 ,cntwd ,bufr ,bufln ,wbuf ,wbuf1)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>				BB	NB	UE	0	x = 0				LU				>>ID*52

Control Request: CALL EXEC(3 ,cntwd ,pram1 [,pram2 [,pram3 [,pram4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
cntwd>>				BB	NB	UE	x	Function Code				LU				>>ID*52

Function Codes

- 00 Clear and Reset Card
- 01 Send EOF
- 06 Dynamic Status of Card
- 11B Top of Form (same as Send EOF)
- 20B Enable Program Scheduling
- 21B Disable Program Scheduling
- 22B Set Timeout
- 23B Disable Program Scheduling (same as 21)
- 45B Use level mode for Device Command signal
- 46B Use pulse mode for Device Command signal

Status Request: CALL EXEC(13 ,cntwd ,stat1 [,stat2 [,stat3 [,stat4]]])

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cntwd>>				0		Z	0				LU				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stat1 <<		AV		Device Type				Status				E			
stat2 <<		AV		Interface Type = 52				0	0	I/O Select Code					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
For Z=0 stat3<<															
First Word of the Driver Parameter Area															

ID*52

For Z=1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<i>stat3</i> >>	Buffer to Return the Driver Parameter Area															
<i>stat4</i> >>	Length of Buffer Described by <i>stat3</i>															

ID*52 Driver Parameter Area

Any words returned from the driver parameter area to *stat3/stat4* or to the status buffer described by *stat3/stat4*, are device dependent.

ID100, ID101, ID400, ID800, ID801, IDZ00 Serial Interface Drivers

These drivers do not have quick reference information but are documented in Chapter 2 of this manual.

IDQ35 SCSI Interface Driver

This driver does not have quick reference information but is documented in Chapter 2 of this manual.

Comparison of Serial Drivers

Introduction

This appendix explains the differences between %IDM00 and the serial I/O drivers released at Revision 4010. It will help you convert from a Revision C, 8-channel MUX to a revision D, 8-channel MUX, or to an HP 12100A A400 4-channel MUX.

The Revision 4010 serial I/O drivers were modified to increase driver compatibility and black box support. The following drivers have been added:

- DDC00.REL, which replaces %DD*00
- DDC01.REL, which replaces %DD*20
- ID100.REL, interface driver for the HP 12005A/B ASIC Card
- ID101.REL, interface driver for the HP 12005A/B ASIC Card with modem support
- ID800.REL, interface driver for the HP 12040D 8-channel MUX
- ID801.REL, interface driver for the HP 12040D 8-channel MUX with modem support
- ID400.REL, interface driver for the A400 4-channel MUX
- IDZ00.REL, interface driver for HP Telnet Pseudo Terminal LUs

For the differences between ID800/01.REL and ID400/01.REL, refer to the “Serial I/O Drivers” documentation in Chapter 2 of this manual in the sections on Function Codes 30B through 34B.

The changes to the read and write requests are minor. All valid binary and ASCII reads and writes using %IDM00 work on the new driver set. The following sections explain the subtle changes in control requests, and how to utilize the enhancements of the drivers.

Read Requests

Among the enhancements to read requests are the following:

- One-character turnaround in write/read requests (using the Z bit).
- Special status read.
- Auto-home in block page mode.
- CRLF echo inhibit during block reads.
- Pseudo full-duplex reads (on 4-Channel MUX only).
- User-defined terminators.

Special Status Read

To enable a program to know the identification of the interface card, a special status read can be used to determine the driver being used (ID800/ID400), the revision, the current control configurations (for the restoration of parameters) and other vital data.

Auto-Home

When in block page mode, setting bit 7 enables the DC1-DC2-DC1 to include a keyboard lock, auto-home (DC1 DC2 ESC c ESC H DC1). With this bit set, only one read is required to implement a forms package read. This allows you to modify your HP terminal, then type “ENTER”.

CRLF Echo Inhibit

For applications performing reads in block mode, the TR bit assumes control if a CRLF is echoed when the read has completed.

Pseudo Full Duplex (4-Channel MUX Only)

This feature is useful for “black box” support. If the application tries to write a very large block of data, no read can be posted until the write has completed. However, the hardware FIFO buffer can overflow during the write. This type of read allows the application to pull the data from the card in small blocks and place it in a specified Z buffer, freeing the hardware FIFO buffers on the card to accumulate more data.

User-Defined Terminators

Any 8-bit character can be stored (using a CN 17 request), to redefine the terminator during a transparent ASCII read. Thereafter, all transparent ASCII reads terminate on the new condition. Although this is intended for “black box” support, it also allows a transparent ASCII read terminating on a CR without CRLF echo.

Write Requests

Normal write requests on the new driver set are similar to the previous IDM00 requests. The enhancement of a forced handshake bit allows the write to force an ENQ/ACK at the start of a request. For driver compatibility, bit 10 inhibits CRLF, bit 9 causes forced handshake, and bit 6 inhibits ENQ/ACK handshaking.

Configuration

All configuration parameters can be specified at generation. The values of driver parameters 1 through 4 initialize the CN 17, CN 20, CN 30, CN 33, and CN 40 parameters. These values may be changed in the Welcome file or at any time.

The specified configuration occurs on the first initiation to the interface driver (such as a read or write request) on a per LU basis. Note that this configuration does not occur when initiated from the remote device (such as a carriage return before initializing the port). In the welcome file, ports may be initialized by a “CN *lu* 6 -1” or a write request to the particular port, showing the port is available.

The D-MUX defaults to the equivalent C-MUX that has been set with “CN *lu* 33B 100000”. The D-MUX does not down the device unless there is an error by a slaved device or a driver defined error.

Note that slaved devices should not be configured in the Welcome file.

CN 6: Dynamic Status

This request returns how much FIFO data is available on the card, the card ID, revision code of the interface driver, and revision code of the firmware.

CN 11: Line Spacing/Page Eject

Conditional page ejects are added.

CN 16: Set Baud Rate Generator (8-Channel MUX Only)

This control call allows you to define your own range for either baud rate generator, instead of being forced to a default range. It also allows for user-defined baud rates using the Z80 processor's timing circuitry.

CN 30: Speed Sensing

Two methods of speed sensing (using CN 30) are available: Automatic (using ENQ/ACK on HP terminals), or CR detection (for non-HP terminals). This is determined by the protocol used (see CN 34). If speed sensing fails (due to no response/no terminal), it will be retried on every unsolicited keystroke from the terminal, until it succeeds.

CN 31: Modem Environment

As an enhancement, the program HPMDM monitors all modem functions. It is available in source form, or any program called HPMDM may be used. This is very useful because the 4-channel MUX contains two built-in modem ports.

CN 32: Generate Break (8- and 4-Channel MUX)

Break generation (a 250-ms space condition online) has been added as an enhancement.

CN 33: FIFO Control (Type-Ahead)

Type-ahead has been renamed FIFO control, and the new name helps to explain the latest changes. If FIFO buffering is enabled, data will be buffered on the card until a read is posted. Data that is entered in this mode (no read pending) will not be echoed or processed, just stored. The maximum size of the FIFO buffer is 1024 bytes on the 8-channel MUX, or 95 bytes on the 4-channel MUX.

Although 95 bytes appears small in type-ahead terms, FIFO is actually a 40ms buffer for incoming data at 9600 baud. This is enough time to post a read or retrieve the data. Just as in the CN33B call, the primary program can be scheduled when data enters the FIFO buffer.

To find out how full the buffer is, a CN06B call will return the byte count in DV17; therefore all data may be removed in one binary read. To remove data from the FIFO buffer, you must read it (as if all the data became available at once), or flush it by re-issuing the CN33B call. The bits in the request are different from that of %IDM00.

FIFO Mode and Type-Ahead Mode

FIFO mode, or its predecessor type-ahead mode, are not available on all interface cards. Programs that depend upon these modes restrict users to using only certain ports on the system. Depending on the complexity of your application, you may be able to write code to interchangeably use FIFO mode or type-ahead mode. More complex applications will require code that handles FIFO mode and type-ahead mode differently. The Revision C multiplexers have type-ahead and the Revision D multiplexers have FIFO. The following paragraphs describe the two modes; refer to Figure C-1 for conceptual block diagrams.

Type-ahead allows input from a MUX port to be saved in a buffer until a read is posted to retrieve the data. This feature permits a limited amount of full-duplex I/O so that input and output occur simultaneously. The unusual aspect is that type-ahead has two buffers of up to 255 bytes each. When input data is received, it is saved in the active buffer until a terminator is received. Because the type-ahead is active when a read is not active, the terminator must be defined by a method other than the usual function bits in the EXEC call control word. The method used is two control calls: CN 37 (Set Read Type) and CN 36 (Set Read Length). Assuming that carriage return is selected as the terminator, the input data is saved in buffer 1 until the first carriage return and then in buffer 2 until the next one. If more data is received after that, it is discarded. This causes

two carriage returns in a row consuming both buffers, even though only two bytes were received. In addition, the data is echoed as it is put in the type-ahead buffer even though the future read might not indicate to echo the data.

To summarize, type-ahead mode has the following characteristics:

- Two buffers up to 255 bytes long between the receiver and the CPU.
- Buffers are ‘filled’ when a terminator byte is received.
- Data echoes as received.
- The type of buffer termination can be selected.

On the Revision D MUX, however, a one-kilobyte buffer is implemented between the output of the UART and the user. The UART (Universal Asynchronous Receiver/Transmitter) on the MUX card serves as a serial-to-parallel converter for incoming data and a parallel-to-serial converter for outgoing data. When incoming data is received, it is put into the FIFO without echo. No interpretation of the data is done, as it is not yet known what kind of read will be removing the data from the FIFO. Thus the full kilobyte can be used without regard to ‘record’ lengths. When a read is posted by a program, the bytes are removed from the FIFO and interpreted according to the rules established by the current request. Each byte is also echoed, if echo is enabled, and compared against the current terminating conditions.

To summarize, FIFO mode has the following characteristics:

- The 1024-byte circular queue is logically between the output of the UART and the receiver.
- Characters are not echoed until a read gets the data.
- The concept of a FIFO buffer terminator does not apply, as the data is not examined until a read is posted and the receiver section retrieves the data from the FIFO.

Note that the Revision C MUX can be upgraded to the Revision D MUX by installation of new firmware. Contact your local Hewlett-Packard Sales Representative for ordering information.

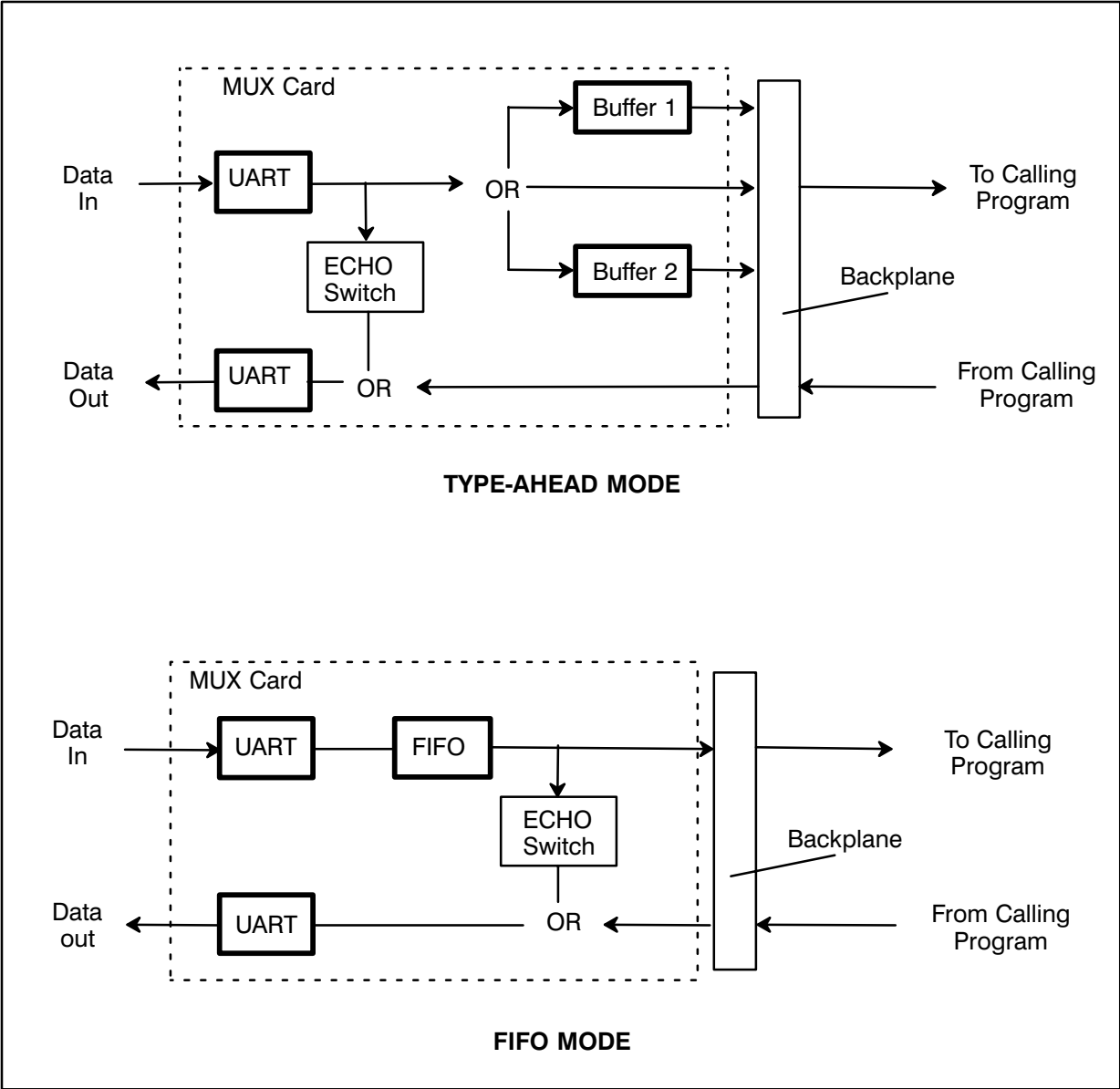


Figure C-1. Conceptual Block Diagram of Type-Ahead Mode and FIFO Mode

CN 34: Set Port Protocol

This enhancement sets the port protocol. Protocols include (bidirectional) Xon/Xoff, HP's ENQ/ACK, TTY (no handshaking), hardware handshake (Xon/Xoff implemented using RTS-CTS), and no LF appending to carriage returns. Also available in the CN 34 call is a timeout bit, which instructs the card to return all of the data on a timeout (set by CN 22).

CN 36: Set Read Length

Not implemented. Use the algorithm illustrated in the section "Function Code 33B: FIFO Buffer Mode Control" in Chapter 2.

CN 37: Set Read Type

Not implemented.

Example Protocol Charts

For the following examples, we will assume that the user has a buffer called ALPHA that contains the upper and lowercase alphabet sequences:

```

...../.....1...../.....2...../.....3...../.....4...../.....5..
ABCDEF GHIJKL MNOPQR STUVWZYX abcdefghi jklmnopqr stuvwxyz

```

← character number
 ← buffer contents

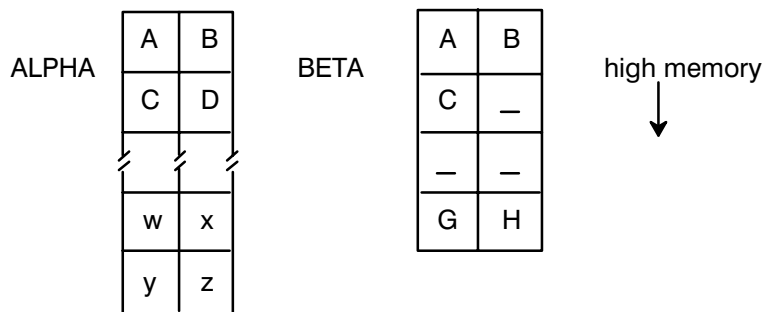
We will also make use of another buffer called BETA that has some underscore characters (octal 137) as follows:

```

...../....
ABC_ _ _ GH

```

In memory, the data would look like this:

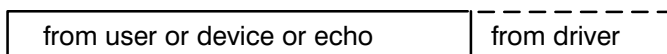


For the read call examples, a 40-word (80-byte) input buffer called IBUF is assumed.

The protocol charts given are representative of what would be observed on the serial data lines using an HP 4953, 4955, or 4951 serial data analyzer. The top line of each chart is the data from the CPU to the device, and the bottom line is the data from the device back to the CPU.

The blank character (octal 40) is represented by “●” in the protocol charts. A character whose value is not known is represented by “■”.

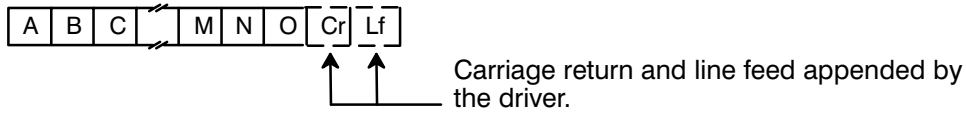
Characters from the user buffer or from the external device or echoed by the driver are shown in solid outline boxes. Characters generated by the driver are shown in dotted outline boxes.



Example 1:

Call EXEC(2,LU,Alpha,-15)

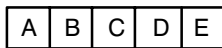
normal ASCII write of 15 bytes to a protocol 0 (TTY) device



Example 2:

Call EXEC(2,ior(LU,2000b),Alpha,-5)

transparent ASCII write of 5 bytes to a protocol 0 (TTY) device

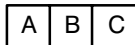


This time the CRLF was not appended by the driver because the transparency bit was set.

Example 3:

Call EXEC(2,LU,Beta,2)

normal ASCII write of 2 words to a protocol 0 (TTY) device

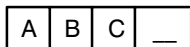


This time the CRLF was not appended by the driver because the last data byte was an underscore. Note that the underscore was also suppressed.

Example 4:

Call EXEC(2,LU,Beta,-5)

normal ASCII write of 5 bytes to a protocol 0 (TTY) device

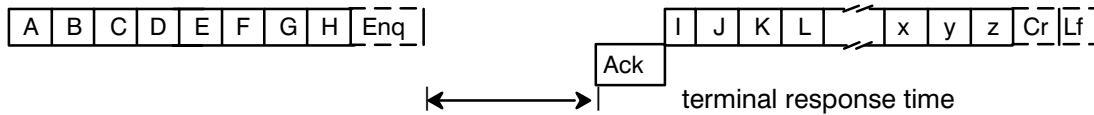


Again the CRLF was suppressed by the driver because the last data byte was an underscore. Note that only the last underscore was suppressed. An underscore in any position other than the last character position is simply data.

Example 5:

Call EXEC(2,LU,Alpha,26)

normal ASCII write of 52 bytes to a protocol 2 (HP) device.

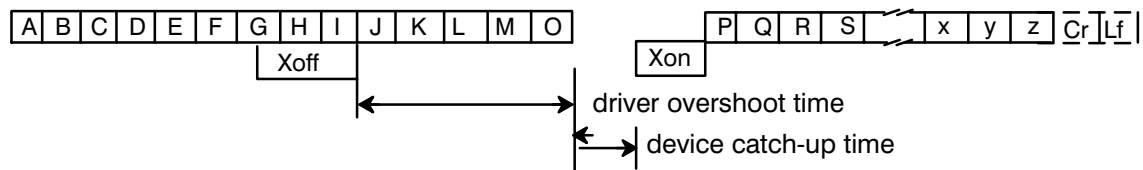


At least once every 80 bytes of output, the driver will suspend data transmission to perform an ENQ/ACK handshake with the terminal. The terminal will not respond with an ACK until it has processed all the characters preceding the ENQ, so the driver is assured that it is okay to resume transmission of data. The handshake does not necessarily occur in a fixed location in each record, because the drivers keep a running 80-byte counter, without regard to the records. This uncertainty can be eliminated by using the Force Handshake bit in the write call. The handshake can be suppressed for one line by doing the write in binary mode.

Example 6:

Call EXEC(2,LU,Alpha,26)

normal ASCII write of 52 bytes to a protocol 1 (Xon/Xoff) device



The Xoff character is asynchronous to the transmitted data. It indicates that the buffer of the receiving device is becoming full and that the transmitter should pause. When the receiving device has emptied its buffers to a sufficient degree, it sends an Xon character to the transmitting device to resume the data flow.

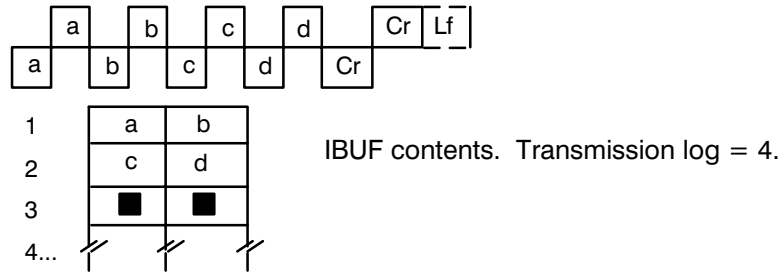
Alternate names:

- Xon ↔ DC1 ↔ Control-Q ↔ ^Q ↔ octal 21
- Xoff ↔ DC3 ↔ Control-S ↔ ^S ↔ octal 23

Example 7:

```
Call EXEC(1,ior(LU,400b),Ibuf,-80)
Call Abreg(iStatus,iTransLog)
```

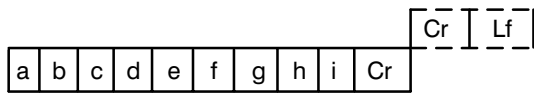
normal ASCII read for 80 bytes with echo from protocol 0 (tty) device



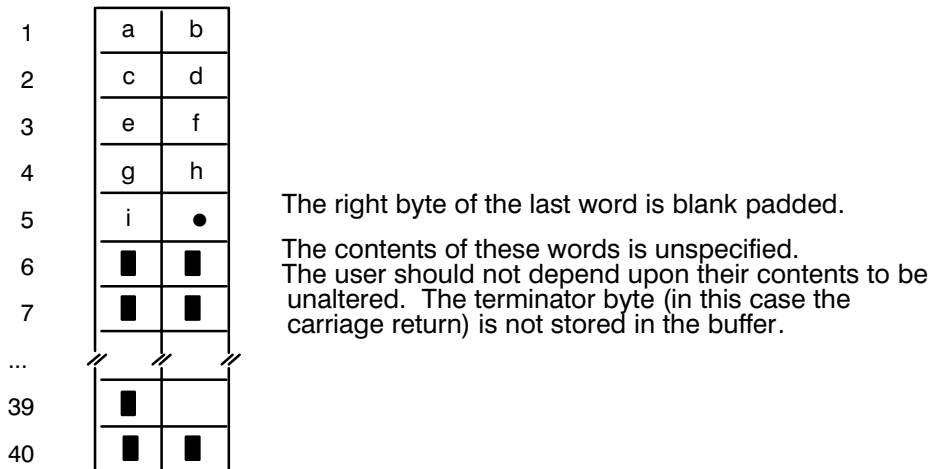
Example 8:

```
Call EXEC(1,LU,Ibuf,-80)
Call Abreg(iStatus,iTransLog)
```

normal ASCII read for 80 bytes without echo from protocol 0 (tty) device



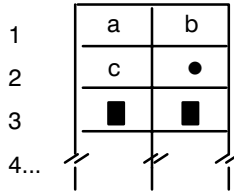
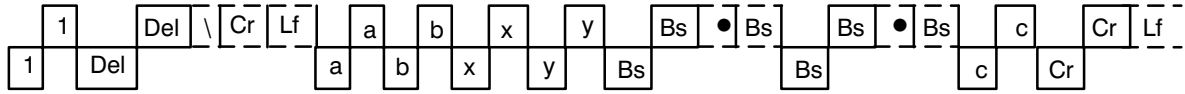
The resulting transmission log would be 9, and IBUF would contain:



Example 9:

```
Call EXEC(1,ior(LU,400b),Ibuf,-80)
Call Abreg(iStatus,iTransLog)
```

normal ASCII read for 80 bytes with echo from protocol 0 (tty) device showing editing characters

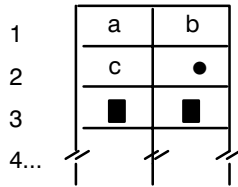
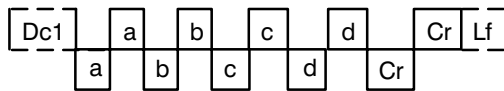


IBUF contents. Transmission log = 3.
Right byte is blank padded because this is an ASCII read.

Example 10:

```
Call EXEC(1,ior(LU,400b),Ibuf,-80)
Call Abreg(iStatus,iTransLog)
```

normal ASCII read for 80 bytes with echo from protocol 2 (HP) device

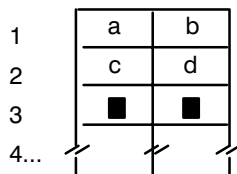
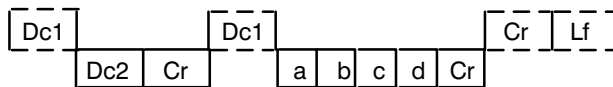


IBUF contents. Transmission log = 3.
Right byte is blank padded because this is an ASCII read.

Example 11:

```
Call EXEC(1,LU,Ibuf,-80)
Call Abreg(iStatus,iTransLog)
```

normal ASCII read for 80 bytes without echo from protocol 2 (HP) device Terminal strapped for line mode and cursor on a line containing "abcd".

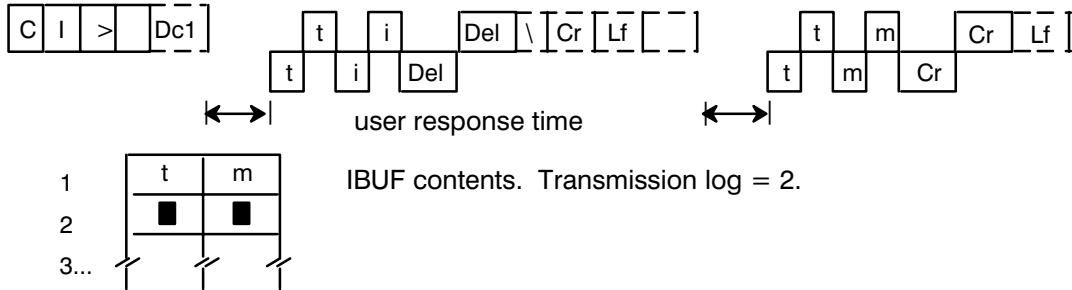


IBUF contents. Transmission log = 4.

Example 12:

```
Call EXEC(1,iOr(LU,10400b),ibuf,-80,5hCI> _,-5)
Call Abreg(iStatus,iTransLog)
```

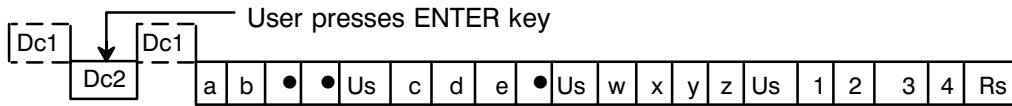
normal ASCII write/read for 80 bytes with echo from protocol 2 (HP) device



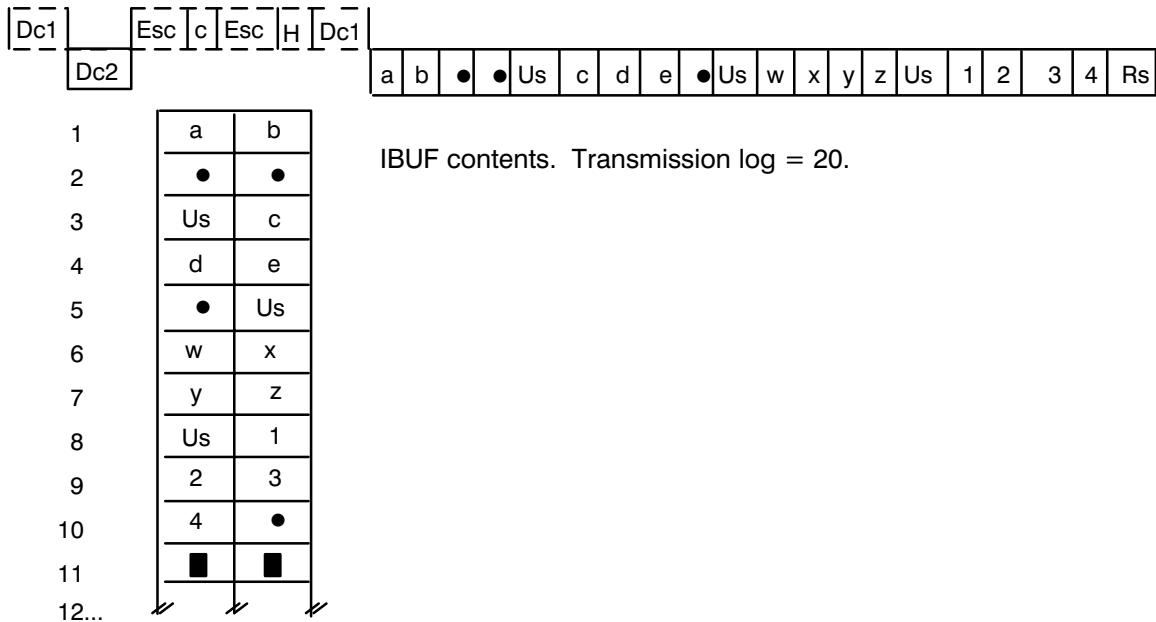
Example 13:

```
Call EXEC(1,Cntwd,Ibuf,-80)
Call Abreg(iStatus,iTransLog)
```

normal ASCII read for 80 bytes without echo from protocol 2 (HP) device terminal strapped for page mode, with a form containing 4 unprotected fields of 4 bytes each.



Same read with Auto-Home bit set.



The fields can be accessed individually by calling either `HpCrtGetField_S` or `HpCrtGetField_I`.

For example,

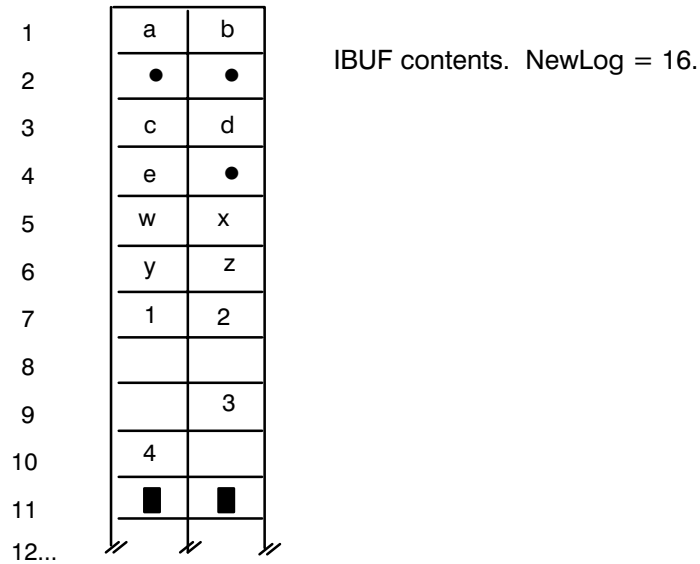
```
Flag = HpCrtGetField_S(ibuf,iTransLog,3,String)
```

would set `String` to “wxyz” and `Flag` would be `.TRUE.` because the third field contained data.

In addition, for backwards compatibility with DVA05, the call:

```
NewLog = HpCrtStripChar(ibuf,iTransLog,37b)
```

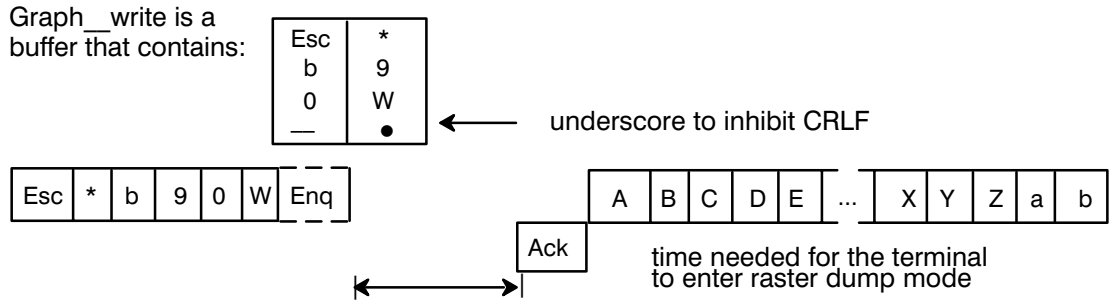
performs an in-place conversion of the data buffer by deleting the Unit Separator characters (octal 37).



Example 14:

```
Call EXEC(2,iOR(LU,11100b),Alpha,-90,Graph_write,-7)
Call Abreg(iStatus,iTransLog)
```

binary write/write for 90 bytes forcing handshake to protocol 2 (HP) device. Note that the graphics escape sequence is in the Z buffer.



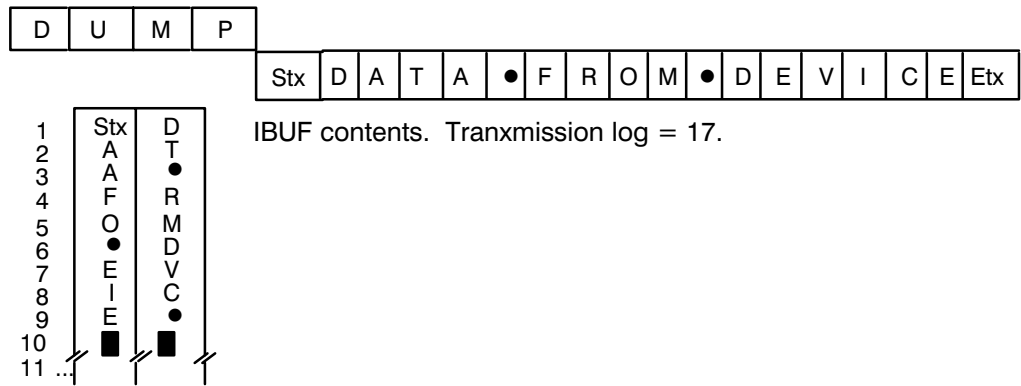
The 90 bytes will be sent without any intervening handshakes that would spoil the graphics data.

See Chapter 3 of the *HP 2648 Graphics Terminal Reference Manual*, part number 02648-90002, for information on raster dumps.

Example 15:

```
Call EXEC(3,iOR(LU,1700b),403b)
Call EXEC(1,iOR(LU,12000b),ibuf,-30,5hDUMP_,-5)
```

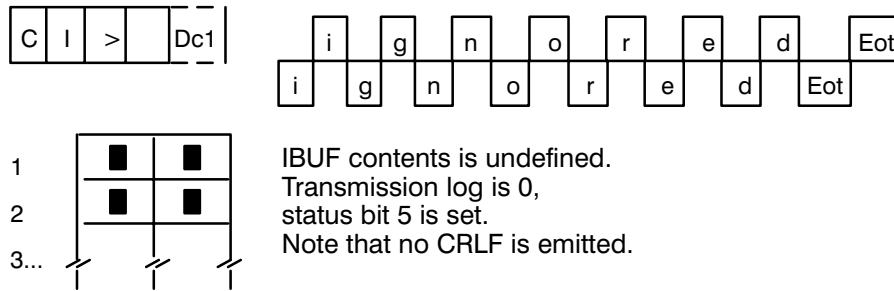
ASCII transparent write/read of 30 bytes from a protocol 4 (CPU to CPU) device with terminator configured to be an ETX.



Example 16:

```
Call EXEC(1,iOr(LU,10400b),ibuf,-80,5hCI>_,-5)
Call Abreg(iStatus,iTransLog)
```

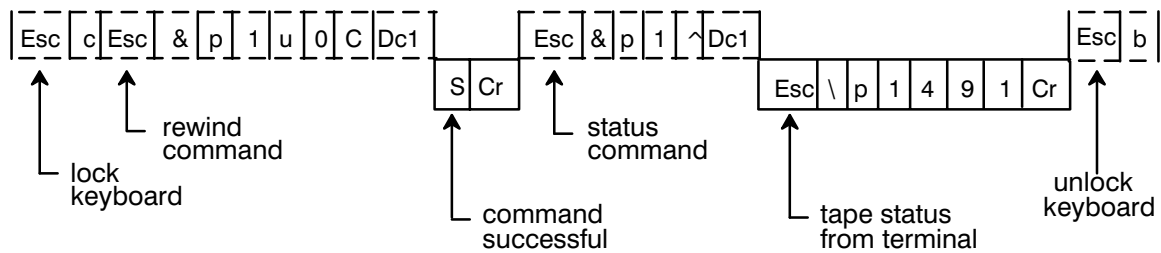
normal ASCII write/read of 80 bytes with echo from protocol 2 (HP) device. The user enters "ignored", which echoes, and then enters a ctrl-D. The data is ignored.



Example 17:

```
Call EXEC(3,iOr(LU,400b))
```

rewind cartridge tape (example is for left tape)

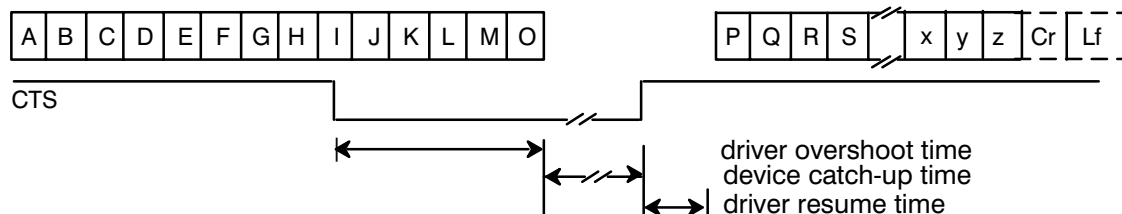


All motion commands are followed by a status check. An ENQ/ACK handshake could appear anywhere in the driver output sequences with no ill effect.

Example 18:

```
Call EXEC(2,LU,Alpha,26)
```

normal ASCII write of 52 bytes to a protocol 8 (Hardware handshake) device



When the receiving device sets the CTS line false, it indicates that the receiving device's buffers are becoming full, and that the transmitter should pause for a while. When the receiving device has emptied its buffers to a sufficient degree, it sets the CTS line high to signal the transmitting device to resume the data flow.

System Generation Considerations

It is convenient to be able to preconfigure the drivers ID800/01, ID400, and ID100/01 so that they “come up” in the right state when the system is booted. The most important parameters to control are the driver protocol and the prompt programs. To set the protocol, set driver parameter 1 to the values that are passed in the function code 34 call. Function code 34 is where the device driver and interface drivers keep these values.

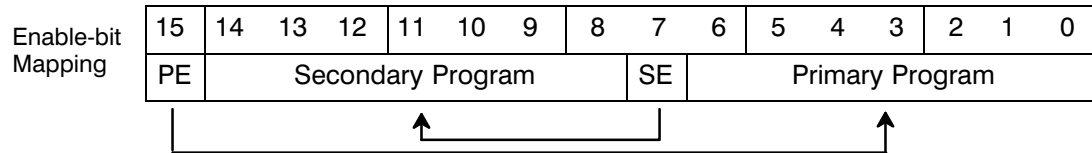
The revision 4010 drivers will keep a unique program name to be scheduled for each DVT (similar to the pre-4010 drivers), but the revision 4010 drivers will only use one word of each DVT, instead of 12 words. This is accomplished by having one master schedule table that contains the program name(s) to be scheduled upon unsolicited interrupt. Each DVT has a one-word (one byte for the primary program and one byte for the secondary) pointer into this table that points to the program name.

The schedule table has enough room for 15 unique entries, with the first five being used (PROMT, CM, CI, FMGR, COMND). This leaves ten entries for the user to define.

To keep the table space that is used to hold the interrupt program names to a minimum, the following method has been adopted: the name “HPMDM” for line supervision has been hardcoded into the interface drivers. The device drivers have an internal table that can hold up to 15 program names. The drivers have preassigned names in 5 of the entries, with 10 entries available for online assignment. The assignments are:

1	PROMT	used in VC+ systems
2	CI	used in RTE-A non-VC+ systems
3	CM	”
4	FMGR	used in RTE-A systems
5	COMND	”
6		spares for online assignment
7		”
8		”
9		”
10		”
11		”
12		”
13		”
14		”
15		”

The programs that have been predefined can be assigned as the primary and secondary programs by setting driver parameter 4 to the correct value. The primary program index number is in bits 3-0, and the secondary in bits 11-8. The enable bits are bits 15 and 7, respectively.



For example, to declare CI and CM enabled at bootup, driver parameter 4 should be set to 101602B.

```

CM/CI      = 1      000      0011      1      000      0010
            = 1      000      001      110      000      010
            = 1      0      1      6      0      2 octal
  
```

The value to use PROMT is 100001B, and FMGR/COMND is 102604B. The generation records supplied in the drivers are generic in nature, specifying a class of device, rather than a single model number. For example, MHP_Term:A can be used to specify any of the HP terminals connected to an ASIC port. If it were connected to port 3 on an 8-channel MUX, it would be MHP_Term:3. Likewise, MHP_Term:1 could be used if it were connected to port B of a 4-channel MUX. The general form is M<generic_class>:<A or 0..7>. Some examples are given below. Refer to Appendix F of the *RTE-A System Generation and Installation Manual*, part number 92077-90034, for a complete list of the generation records supplied with DDC00/01.

```
MHP_Term:A DT:05 BL:bu:40:400 DP:1:2:0:0B:100001B
```

- Generation records for HP CRTs
- Use ENQ/ACK handshake
- Speed sense enabled on MUXs
- PROMT = primary program

```
MTerm:a DT:0 BL:bu:40:400 DP:1:1:0:0:100001B
```

- Generation records for non-HP terminals
- Use Xon/Xoff handshake except on ASIC
- Speed sense enabled on MUXs
- PROMT = primary program

To support TELNET for HP TELNET LUs, add the following:

```
MHP_Telnet DT:05 DP:1:2:0:20000B:100001B TX:16
```

To support TELNET for non-HP terminal TELNET LUs, add the following:

```
MTelnet DT:0 DP:1:1:0:20000B:100001B TX:16
```

MHP_Printer:A DT:12B BL:bu:40:400 DP:1:4402B:0:0:0

- Generation records for HP non-interactive printers
- Use ENQ/ACK handshake
- Printer and hardcopy bits set
- 9600 baud on MUXs

Note that no primary or secondary interrupt programs are specified.

MPrinter:A DT:12B BL:bu:40:400 DP:1:4401B:0:0:0

- Generation records for non-HP noninteractive printers
- Use Xon/Xoff handshake except on ASIC
- Printer and hardcopy bits set
- 9600 baud on MUXs

Note that no primary or secondary interrupt programs are specified.

MPlotter:A DT:0 BL:bu:40:400 DP:1:1:0:0: TX:12

- Generation records for plotters
- Use Xon/Xoff handshake except on ASIC
- Device type 0
- 9600 baud on MUXs

Note that no primary or secondary interrupt programs are specified.

MQTD_Port7 DT:00 BL:UN:0:0 DP:1:100004B:0:77b:0

- Generation record for the 37214A Queensferry Modem Card cage
- CPU-CPU protocol with FIFO enabled
- 1200 baud port 7
- Device is unbuffered
- Program scheduling disabled

MHP_2635:A DT:06 BL:bu:40:400 DP:1:4002b:0:0:100001B

- Generation records for the HP 2635 printing terminal
- Use ENQ/ACK handshake
- The hardcopy bit is set, so the device type is 6
- Speed sense enabled on MUXs
- PROMT = primary program

In addition, there are five generation records to support slaved devices using DDC01. They are:

MHP_Ctu:L	DT:20B	DP:4:1	BL:UN:0:0
MHP_Ctu:R	DT:20B	DP:4:2	BL:UN:0:0
MHP_Slaved_Serial	DT:12B	DP:4:4	BL:UN:0:0

```

MHP_Slaved_HPIB      DT:12B DP:4:5 BL:UN:0:0
MHP_Internal_Prtr    DT:12B DP:4:6 BL:UN:0:0

```

Note that DP4 has a different meaning for slaved devices. It is the subchannel number. If you have a device that is not in the tables above, try to use one of the generic devices that is predefined. If you cannot, then refer to the control code section of this manual. You can experiment with the system online to test your proposed values. Once the values for the various control calls are known, refer to the Driver Parameter Area section to see how the bits are mapped into the words stored there. Put those words in your answer file during the DVT definition to complete the process.

For example, assume that you have a black box that requires CPU-to-CPU protocol at 1200 baud, 2 stop bits, 7 data bits, and even parity. It terminates transmission with an ETB character. Port 5 of your MUX is available and the 1200 baud rate does not conflict with any other port sharing the same baud rate generator. In this case, the CN 34 parameter to set up the protocol is 4B. The CN 17 parameter to terminate on ETB is 4027B. The CN 30 parameter for 1200 baud, 2 stop bits, 7 data bits, and even parity on port 5 is 043475B. Your black box does not require a program to be scheduled on interrupt, so the driver parameters should be: DP:1:4:4027B:43475B:0. If your device requires an interrupt program, and none of the standard ones are appropriate, then issue a CN 20 call in the Welcome file. In such a case, specify the port to be a terminal port in the generation, then issue all the calls from the Welcome file. This could also be done from the program via EXEC calls.

Choosing the Correct Driver

To use this section, first examine the flowchart in Figure E-1 to determine the proper course of action. Then consult the proper section below for more details.

For an A400 computer, relocate ID400.REL during the driver relocation phase of the generation.

For an HP 12040 8-channel MUX that has been upgraded to D-level firmware, relocate either ID800.REL or ID801.REL. Use ID801 only if you have an HP 37214 modem card cage connected to the MUX. Relocate only one of the two, since ID800 is a subset of ID801.

If you relocate any of the drivers given above, you must relocate either DDC00.REL or DDC01.REL. Use DDC01 if any of your terminals has cartridge tape units (HP 2644, 2645, 2647, or 2648) or if they have either an internal printer or an external printer slaved from the terminal. A slave printer is one which is connected to the terminal rather than directly to the A-Series computer. Relocate only one of the two, since DDC00 is a superset of DDC01.

Thus, the relocation phase could include up to five of the following:

```

REL, ID400.REL, , , ,    A400 Onboard 4-channel MUX driver
REL, ID800.REL, , , ,    8-channel MUX driver
REL, ID801.REL, , , ,    8-channel MUX with HP 37214 support
REL, ID100.REL, , , ,    ASIC Card Driver
REL, ID101.REL, , , ,    ASIC Card Driver with modems
REL, DDC00.REL, , , ,    serial port device driver
REL, DDC01.REL, , , ,    device driver with CTU and printer support
REL, IDZOO.REL, , , ,    TELNET driver

```

In the device table definition phase, you must include an IFT definition for each serial interface driver, followed by the DVT definitions for the devices connected to that interface. Note that the LU numbers in the example are arbitrary, as are the select codes, except for the 4-channel MUX.

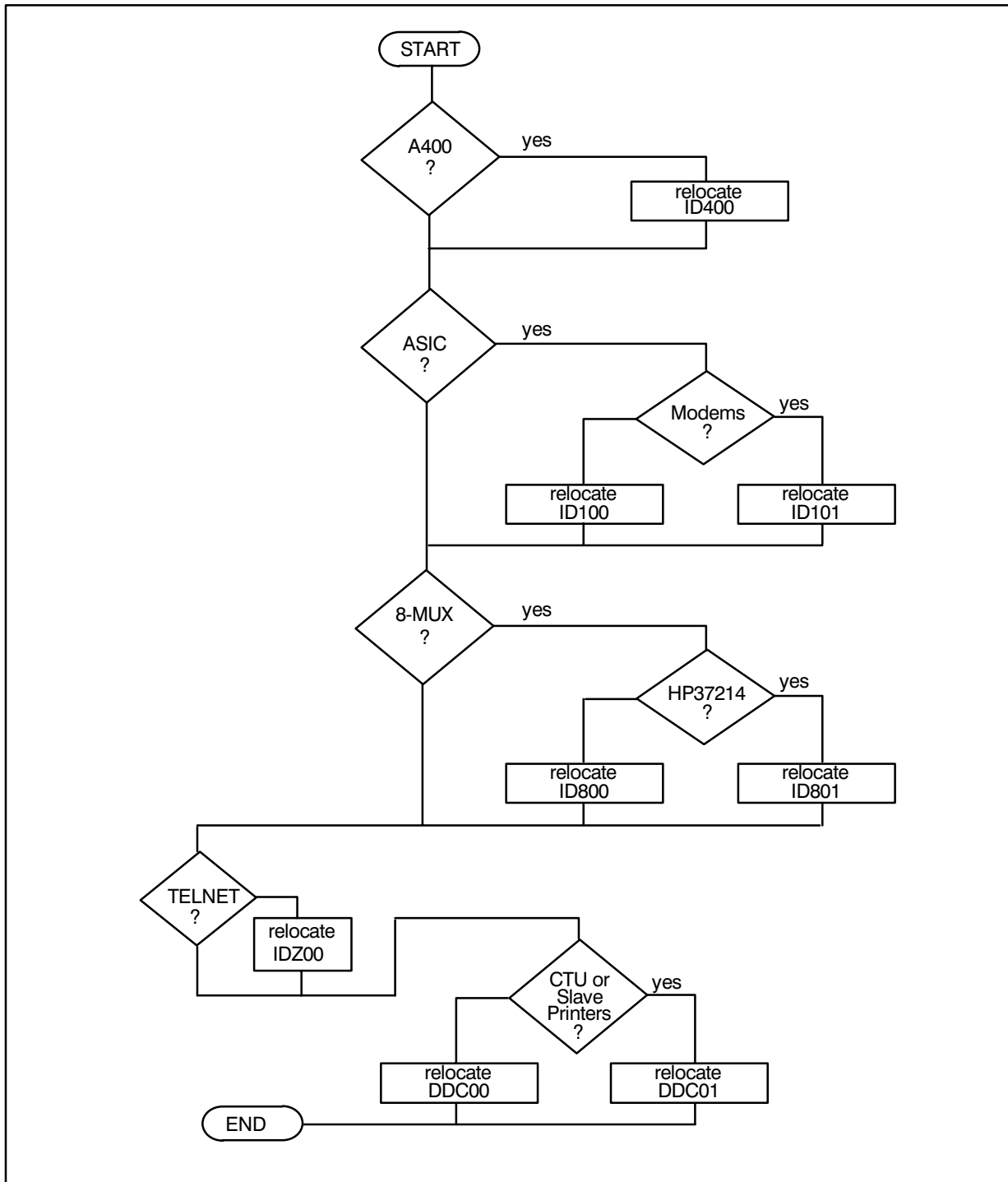


Figure E-1. Flowchart for Selecting Serial Interface Drivers

```

* The 4-channel MUX on an A400
ift id400.rel ← the select code is always 77B and should
dvt ddc01.rel mHP__Term:0 LU:40 NOT be supplied in the IFT specification
dvt ddc01.rel mHP__Term:1 LU:41 line. It is defaulted.
dvt ddc01.rel mHP__Term:2 LU:42
dvt ddc01.rel mHP__Term:3 LU:43
*
* An 8-channel MUX
ift id800.rel sc:23b
dvt ddc01.rel mHP__Term:0 LU:40
dvt ddc01.rel mHP__Term:1 LU:41
dvt ddc01.rel mHP__Term:3 LU:43
dvt ddc01.rel mHP__Term:4 LU:44
dvt ddc01.rel mHP__Term:5 LU:45
*
* this is the black box from the example above
dvt ddc01.rel,,LU:46 DT:72b DP:1:4072b:43475b:0

*
* this is the system printer
dvt ddc01.rel mHP__Printer:7 LU:6
ift id100.rel
dvt ddc01.rel mHP_Term LU:48
*
* A TELNET Pseudo Interface with 4 Pseudo LUs
ift idz00.rel ← The select code is always zero
dvt ddc00.rel mHP_Telnet LU:110 (pseudo driver) and should NOT
dvt ddc00.rel mHP_Telnet LU:111 be supplied in the IFT specification line.
dvt ddc00.rel mHP_Telnet LU:112 It is defaulted.
dvt ddc00.rel mHP_Telnet LU:113

```

Sample Page Mode Application

The following is an example of a page mode application. Note that it requires the terminal driver to support the Auto-home bit in an EXEC read.

```

character*80 FieldString
integer*2 Clear(3)
integer*2 Ibuf(200)
data Clear /15510b,15512b,15542b/

100  call EXEC(2,LU,Form,FormLength)
      call HpCrtPageMode(LU,.true.)

200  call EXEC(2,ior(LU,2000b),Clear,-6)

      call EXEC(1,ior(LU,200b),
>          ibuf,-400)

      call abreg(istatus,itrans_log)
      if ( iand(istatus,1).eq.1 .or.
>        itrans_log.eq.0 ) then
          go to 100
      endif

      flag = HpCrtGetField_S(ibuf,itrans_log,
>          1,fieldstring)
      if ( flag ) then
          if ( fieldstring.eq. 'EN' ) then
              go to 300
          endif
      endif

*****
*  Process the data from the screen here  *
*****

300  go to 200
      call HpCrtCharMode(LU)
      call EXEC(2,ior(LU,2000b),Clear,-6)

      end

```

```

! to receive one field
! escape sequence
! to receive raw screen data
! home cursor, clear form,
! and unlock keyboard
! (Esc H Esc J Esc b)
! put the form on the screen
! put terminal in block page
! mode, protected forms enabled
! clear the unprotected fields,
! enable the keyboard
! do the block mode read of the
! unprotected part of the form
! auto-home will lock the keys
! get driver status
! check for timeout
! or zero transmission log
! which can happen only if
! the user messed up the CRT
! so go through set-up again
! get data from first
! unprotected field

! look for end command
! to escape application

! do another screen
! back to normal terminal mode
! clear the screen and enable
! keyboard before terminating

```


Program Scheduling

This state diagram shows the transitions caused by the program schedule control calls (CN 20, 21, 40, 41) in RTE-A.

PR – primary program
 SE – secondary program
 RT – RTE prompt

DI – disabled
 EN – enabled

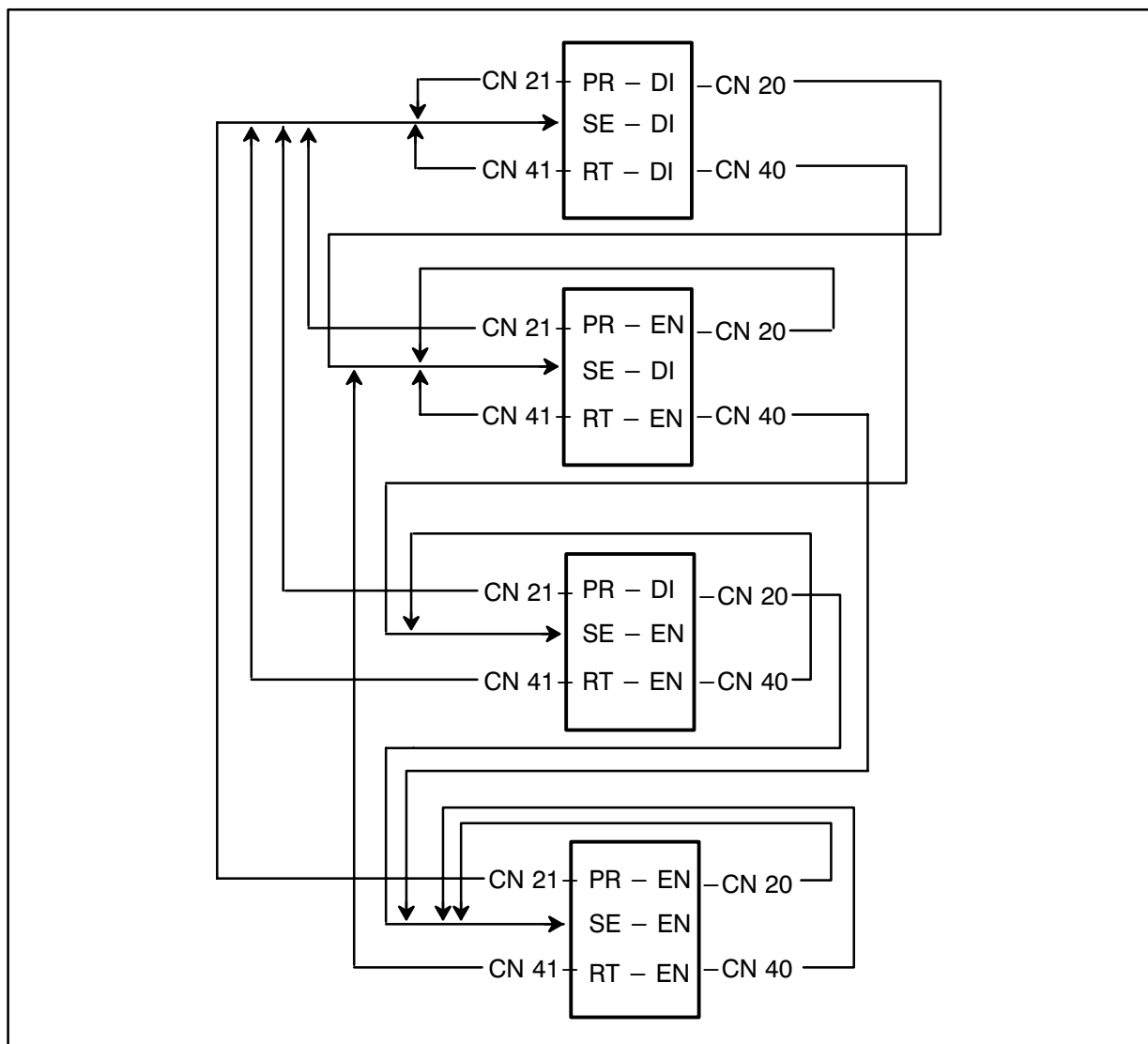


Figure G-1. State Diagram for Program Scheduling

Source Code Programs

The following HP-provided subroutines are supplied as source code with the RTE-A Operating System for use with the serial I/O drivers.

RMTERM - Remote Terminal Download

RMTERM.FTN is the source code for RMTERM.RUN, which allows a remote terminal to be configured into a system. This program only runs on an HP 12040D 8-channel MUX or HP 12100A 4-channel MUX. The program requires two ports, one connected to the terminal on the host system, and one port connected to the remote system via a serial line (RS-232/RS-422). RMTERM.RUN acts as a data router; it reroutes data from the terminal to the remote system, and data from the remote system to the terminal. This allows the terminal on the local system to appear to be physically connected to the remote system. Using this method, ENQ/ACK handshaking is done by the terminal on the local system. The two ports must be connected with TX and RX (bits 2 and 3) reversed.

This program contains a download subroutine that allows a type 1 file to be downloaded to a remote system via the VCP of the remote system. This allows memory-based systems and those not using DS to be booted via RS-232/RS-423.

To start RMTERM, enter:

```
CI> RMTERM communicationlu baud
```

where *communicationlu* is the port on the local system that is connected to the distant system, and *baud* is the baud rate of the link (for example, 19.2K or 9600). To terminate remote terminal mode, press the BREAK key which will enable you to:

```
generate a break  
leave RMTERM  
enter the download subroutine, or  
continue
```

Note that overrun problems can occur if the baud rate of the remote system is much faster than the local system, and the remote sends a block of data, filling up the FIFO buffer.

The source can be modified. The subroutine included downloads a system via a VCP port, but the remote system may be another computer, a robot, or a black box that communicates in ASCII format. Then a specific download subroutine can be written to perform a file transfer or any other special task required for the application.

The included download subroutine allows 0.5 Mbytes of memory to be downloaded into a remote HP 1000 in about seven minutes at 19.2K baud, or five minutes at 76.8K baud (A400 only).

AUTOR - Powerfail Automatic Restart

AUTOR.FTN is the source code for AUTOR.RUN that allows system recovery after a power failure, printing out a powerfail message to all terminals on the system. It is called by the system upon restoration of power. The program must be RP'd in the system session.

The source is designed to be easily modified, and may be changed to start any processes required for system restoration.

HPMDM - Modem Control

HPMDM.FTN is the source code for HPMDM.RUN, which allows control of serial ports on the HP 12005B (ASIC) and 12040D (8-Channel MUX) interface cards, and the HP 12100A (A400 4-Channel MUX) computer board. The drivers for these cards all attempt to schedule program HPMDM whenever a modem control event occurs on the modem ports. These events are: incoming call, timeout on a connect request, or modem asynchronous disconnect. Although no harm occurs if this program does not exist, it allows the removal of sessions if the user hangs up without logging off, or has telephone line problems.

This program is for use with modems that are controlled using the modem control lines, namely DTR and RTS (controlled by the interface card), and DSR, CTS, and CD (controlled by the modem). The modem should be configured so that DTR is controlled by the computer, not forced true. This allows HPMDM to control connections and disconnections.

HPMDM utilizes a block in system common, which must be generated into the system. This block is labeled HPMDM_Table, and contains information concerning which modems are controlled by HPMDM, and which configuration parameters are to be used upon modem connection. This table may be altered if additional information is needed for a specific application.

The program is scheduled by either the user, or the system (driver for line changes, AUTOR for powerfail, LOGOF program for session disconnect). Both of these methods are described below.

Method 1: Scheduled by User

This allows the user to configure which parameters to use upon connection, which modems are controlled by HPMDM, and to display current parameters. If HPMDM is scheduled by the user with no parameters, a help menu is displayed:

```
Usage:                                     <yymmdd.hhmm>
HPMDM <lu> AD [OK] - Add (re-init) LU to HpMdm control
HPMDM <lu> DE [OK] - Delete LU from HpMdm control
HPMDM <lu> CO      - Configuration for an LU
HPMDM AL          - Configuration for all LU's
HPMDM TO <1..655> - New connection timeout value in seconds
HPMDM TO          - Display current connection timeout value
HPMDM VE [AL]    - Display logon/logoff messages to console
                  - "AL" display even if not in HPMDM table
HPMDM QU         - Don't display logon/logoff messages
HPMDM <lu> COMMAND=OPERAND [COMMAND=OPERAND] ...
HPMDM ?          - this help again
HPMDM ??         - more extensive help
```

where *yymmdd.hhmm* is the last modification time of the program.

To change connection parameters, enter:

```
CI> HPMDM lu com1=op1 com2=op2 com3=op3 . . .
```

where *com* is a user command, and *op* is a user option. For example:

```
CI> hpmdm 25 baud=1200 program=prompt fifo=enable
```

would set the baud rate (CN 30) to 1200 baud, the primary program (CN 20) to PROMT, and enable FIFO buffering (CN 33).

The available options are:

Option	Choice
TImeout	user timeout [1..653 seconds]
BAud	[50, 75, 110, 134, 150, 300, 1200, 1800, 2400, 4800, 9600, 19.2K, 38.4K, 76.8K, SPeedsense on logon, NOT programmable, AUto-speed sense]
LOgof	YEs/NO (disconnect on logoff)
FRame	[5,6,7,8] bits per character
STop bits	[1,1,.5,2] stop bits per character
PARity	[ODd, EVen, NOne]
HAndshake	[CP (CPU to CPU), TT (dumb terminal), protocols HP (enq/ack), XO (xon/xoff), HX (HP and XO combined), H2: Half HP (stat mux), X2 (Half HP and XO combined)
T5	[ENable, DISable] Force type 5
FIfO	[ENable, DISable] FIFO buffering
CHaracter	[ENable, DISable] Character by character program scheduling
SAve data	[ENable, DISable] on break key entry
PRogram	[XXXXX], XXXXX is RP'd program
VERbose	display new connection configuration

For example:

```
CI> hpmdm 25 baud=300 parity=none ha=hp pr=prompt verbose
```

In addition, you may display the current configuration of a particular LU by entering:

```
CI> hpmdm lu co
```

```
HPMDM CONFIGURATIONS:  
User Timeout: 122 seconds (2.0 minutes)  
CN30: modem, 8 bits/char, 1 stop bit(s), NO parity, 1200 baud, port #1  
CN33: FIFO buffering  
CN34: HP protocol  
PRIMARY PROGRAM: PROMT
```

The connection parameters are the CN 30, CN 33, CN 34, CN 20, and CN 22 parameters to be used to configure the port. Upon each connection, these commands are re-issued to correct anything that the previous user may have done (for example, enabling FIFO buffering, if most users do not want that feature). This is useful, because a program such as CALLB could be initialized as the primary program. This way, when a user dials in, CALLB could query them for a name and password, disconnect, then call the user back using a number in an internal table, and resetting the primary program (CN 20) to PROMT.

To change which modems are controlled by HPMDM, enter:

```
CI> hpmdm lu add
```

or

```
CI> hpmdm lu delete [ok]
```

ADD or DELETE may be abbreviated as AD or DE, respectively. If a session exists, "OK" must be added or HPMDM replies that the session is in use, and does not disconnect it. Adding an existing LU re-enables the LU, killing any existing session.

To change the length of time HPMDM allows for a modem to connect after it enables the connection, enter:

```
CI> hpmdm 0 <new connection time value in seconds>
```

To display which LUs are controlled by HPMDM, enter:

```
CI> hpmdm all
```

All LUs controlled by HPMDM are displayed, and the current status and connection parameters are also displayed.

Method 2: Scheduled by Driver

The driver schedules HPMDM with parameter 5 being the special key of 12345B. Although you may include this key when HPMDM is scheduled, doing so causes the session to be disconnected from the current user. HPMDM will take appropriate action. This is included here only if you need to modify the HPMDM code.

- Pram1: LU (or -1 if scheduled by AUTOR)
- Pram2: 0
- Pram3: 0
- Pram4: Reason for scheduling HPMDM:
 - 2: Scheduled by LOGOF to indicate removal from session
 - 1: Scheduled by AUTOR to indicate recovery from a power fail
 - 1: Incoming call
 - 2: Asynchronous (modem initiated) disconnect
 - 3: Timeout upon connection request (negative response to a CN31)
 - 4: Successful completion of connection request (positive response to a CN31)
- Pram5: Special Key: 12345B

HPMDM is used with a program such as the source program CALLB, to implement a security system. Using the runstring:

```
CI> hpmdm lu pr=callb
```

causes CALLB to become the primary program. When a user dials in, CALLB is rescheduled. It queries the user for an ID and password, and if correct, dials the user back, and sets the primary program to PROMT. When the user disconnects, and another incoming call is received, HPMDM changes the primary program back to CALLB, allowing another user to use the security feature.

CALLB – Security Callback

CALLB.FTN is the source code for CALLB.RUN. It is supplied as a starting point to create a security callback system for the HP 1000. It works with ID801 and ID101 to provide an increased level of security for dial-in lines.

CALLB works such that the dial-in user does not get full access to the system, only to the CALLB program. CALLB verifies the user access by an ID and keyword that are independant of the user name and password maintained by GRUMP. Once the user has been identified, the system breaks the connection and uses the telephone number from an internal data base to place a call back to the user to set up a normal session.

Although no form of dial-in access is completely secure, this method makes it more difficult to break into the system.

It may also be an advantage that the telephone toll charges, if any, are charged to the originator. The user pays only for a short call for identification, then the organization that owns the HP 1000 pays for the outgoing call.

Creating a Special Character PROM

When using the ASIC card interface drivers (ID100, ID101, and ID200), you may specify a user-defined termination character using a CN 17 call. The card detects the termination character by using a look-up PROM.

If you defined CR (15b) as the termination character, the driver enables type 3 character detection. The standard PROM, supplied with the ASIC card, has CR defined as a type 3 special character.

If you defined RS (36b) as the termination character, the driver enables type 2 character detection. The standard PROM supports RS as a type 2 special character.

If you defined any other character as your termination character, the driver enables type 2 character detection, but you must supply a special PROM to detect that character. Note that you will lose the ability to use the card with HP block mode terminals, because they depend on RS being a type 2 termination character.

The special character types are:

- | | | |
|--------|-----------------------|--|
| type 1 | CR, BS, DEL, EOT, DC2 | (normal ASCII reads) |
| type 2 | RS | (block page mode and user-defined terminator, except CR, enabled) |
| type 3 | CR | (transparent ASCII reads, no user-defined terminator or user-defined terminator is CR) |

Refer to the *HP 12005 ASIC Installation and Reference Manual*, part number 12005-90002.

The contents of the standard HP look-up PROM are:

<u>Location</u>	<u>Contents</u>
00 – 17:	17 17 17 17 16 17 17 17 16 17 17 17 17 12 17 17
20 – 37:	17 17 16 17 17 17 17 17 17 17 17 17 17 17 15 17
40 – 57:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
60 – 77:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
100 – 117:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
120 – 137:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
140 – 157:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
160 – 177:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 16
200 – 217:	17 17 17 17 16 17 17 17 16 17 17 17 17 12 17 17
220 – 237:	17 17 16 17 17 17 17 17 17 17 17 17 17 17 15 17
240 – 257:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
260 – 277:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
300 – 317:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
320 – 337:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
340 – 357:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
360 – 377:	17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 16

To create a special character PROM for a user-defined type 2 termination character other than RS, do the following:

1. Change the 15 in locations 36B and 236B (note that locations 36B and 236B correspond to the octal value of RS) in the table above to 17.
2. Clear bit 1 in the locations that correspond to the octal value of the character you want to specify as your termination character.

For example, to define the termination character as EOT, change locations 4B and 204B to 14. Similarly, for ETX, change locations 3B and 203B to 15, and for tilde (~), change locations 176B and 376B to 15.

DO NOT define more than one character in this way.

3. After you have defined the table, burn it into a 256-by-4 bipolar PROM that is equivalent to a Harris 7611.

Before installing your custom PROM onto the ASIC card, note the following:

- Mark the PROM in some way so that it is easily recognizable as being a custom PROM. This will alert service personnel so that if the ASIC card must be replaced, the PROM will be transferred to the new card.
- You must keep the HP standard PROM for service personnel to use when running diagnostics.
- Be aware that if you call for HP service because the card does not work in block mode, and the altered PROM is the cause, then you will be charged for the service call. Or, if the card is damaged due to mishandling (such as electrostatic damage while performing this process), you will incur the cost of replacement.

Serial I/O Drivers (prior to Rev. 4010)

This appendix documents the older serial I/O device and interface drivers (prior to Revision 4010). See Chapter 2, “Serial I/O Drivers” section for information on serial I/O drivers that were introduced at Revision 4010.

Serial I/O Device Drivers

As of Revision 4010, serial I/O device driver DD*00 (for terminals) has been replaced in RTE-A by driver DDC00; driver DD*20 (for terminals with CTU) has been replaced in RTE-A by driver DDC01.

DD*00 and DD*20 will continue to exist as relocatable masters and can be made available when you regenerate your system. Although not supported, they are documented here for reference purposes.

Terminal Device Driver DD*00

DD*00 is the terminal device driver for the following devices:

- HP 2621A/P Terminal
- HP 262xA/B Terminals
- HP 264x Terminals
- HP 2382A Terminal
- HP 2392A/93A/97A Terminals
- HP 2635 Terminal/Printer
- HP 256x Line Printers
- HP 2631 Line Printer
- HP 2686A/B Laser Printers
- HP 2687A Laser Printer
- HP 2932A/33A/34A Line Printers
- HP 2225C Thinkjet Printer
- HP 744x Plotters (using serial interface)
- HP 755x Plotters (using serial interface)
- HP 758x Plotters (using serial interface)
- HP 759x Plotters (using serial interface)
- HP 45610B HP Touchscreen Terminal
- HP 700/41 Terminal
- HP Vectra

DD*00 provides the software interface to the device, while the asynchronous serial interface driver ID*00 performs the actual I/O instructions to the interface card.

Read and Write Modes

Any of the following modes may be used in reading and writing to a terminal:

Normal ASCII write
Transparent ASCII write
Character mode, normal ASCII read
Character mode, transparent ASCII read
Block mode, keyboard read

Normal ASCII Write Mode

In this mode, the output to the terminal is a word string from a buffer with each word containing two ASCII characters. The driver terminates the character string by supplying a carriage return (CR) and line feed (LF). If an underscore (_ , 137B) is the last character of the buffer, the underscore character, CR, and LF are not output to the display.

Transparent ASCII Write Mode

As with normal ASCII write mode, a string of ASCII characters is output to the display; however, the driver does not supply the CR or the LF at the end of the buffer. It is the programmer's responsibility to supply the CR and/or LF to the buffer where needed. The underscore character is output to the display if it is the last character of the buffer.

The driver outputs the buffer to the display in 78-character increments so that the time required for processing to the terminal will not cause some of the characters to be lost. For baud rates greater than 4800, the terminal ENQ/ACK (5B/6B) handshake sequence should always be used or characters will be lost in the transmission.

Terminal functions such as line spacing, margins, and page size can be controlled by escape code sequences. The escape code sequences can be embedded in the output buffer for the terminal, so write requests can perform some control functions.

Character Mode Normal ASCII Read

Characters are echoed to the terminal when keys are pressed. The record terminator (CR) must be entered to complete the request, but it is not sent to the user's buffer. DD*00 returns the transmission log in the B-Register. If CR is the first character transmitted, the transmission log is zero. If there is no CR, the request completes by character count. If the buffer fills before a CR, the request is completed immediately.

Note For HP 12005A cards, the read always terminates on character count, regardless of any CR that is transmitted.

The following characters are processed by DD*00 for a normal ASCII read request:

CARRIAGE RETURN	(CR, 15B) A carriage return is echoed to the display but is not sent to the user buffer.
DEL	(RUBOUT, 177B) DEL deletes the current record and puts a backslash at the cursor position. DEL is used to delete a line and start a new one.
BACKSPACE	(10B) BACKSPACE deletes the last character and moves the cursor back one position on the CRT display.
CONTROL D and 7 (^D, 4B)	CONTROL D terminates the transmission and sets bits 5 of DVT6 to indicate end-of-medium and end of transmission.
CONTROL R	(DC2, 22B) CONTROL R should not be entered as the first character in a line. It has special significance in terminal protocol.

Character Mode Transparent ASCII Read

The special characters listed above, except CR, are not processed by DD*00, but are passed directly to the user buffer. The transfer is stopped by a CR, which is not transmitted to the user buffer. If there is no CR, the request completes by character count. If the buffer fills before CR, the Asynchronous Serial Interface Card (ASIC) will terminate the request. The Multiplexer (MUX) card, however, will not terminate the request unless it is specifically configured to terminate on character count.

For keyboard read requests, the driver sends a DC1 (21B) to the terminal before setting the interface card to the receive mode. Therefore, when reading cursor status or other special terminal information, do not explicitly send a DC1 in the output buffer; send only the escape sequence up to the DC1.

The DC1 to trigger the terminal transfer is supplied by ID*00 in the read request. The driver distinguishes between block and character modes by examining the first character received from the keyboard. If it is a DC2 (22B), the driver assumes that the ENTER key was pressed and a block transmission is pending; the driver responds with DC1 to trigger the block transfer. If the first character is not a DC2, the driver assumes that a character transfer is pending.

If a CONTROL R (DC2) is entered as the first character of a line, only CONTROL ^ (Record Separator RS, 36B) will terminate the request. Echoing is suppressed (making the terminal appear dead), and the RUBOUT (DEL) is not recognized or processed.

Block Mode Keyboard Read

For terminals that permit block-mode keyboard reads, the BLOCK MODE key must be activated to permit transmissions either line-by-line or page-by-page, because the terminal is strapped for page mode. The DEL, BACKSPACE, LINE FEED, and CNTL D keys will have no special meaning and are not processed. The ENTER key must be pressed to perform the read.

Line separators (CRLF) are passed to the user buffer, as they appear in the data when more than one line is entered. The data terminator RS, generated by pressing the ENTER key, is not passed to the buffer.

If only one line is entered from the terminal, neither CRLF or RS will be passed to the user buffer.

DD*00 Read/Write Request

The read and write request call sequences take the form

Read: CALL EXEC (1, CNTWD, BUFR, BUFLN[, PRAM3 [, PRAM4]]

Write: CALL EXEC (2, CNTWD, BUFR, BUFLN[, PRAM3 [, PRAM4]]

CRT Terminal Control Word CNTWD

Figure J-1 shows the format of the control word CNTWD used for a read or write request to DD*00 for CRT terminals.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	Z	X	TR	PR	EC	X	BI	X					

Figure J-1. DD*00 Read/Write Request CNTWD Format

Note For binary transfers, make sure that the card and driver are configured for 7-bit or 8-bit operation. The default is 7-bit.

The NB, UE, EC, and LU bits are defined in Chapter 1 under the Read/Write Parameter Definitions section. If the TR or BI bit is set, DD*00 will use the transparent definitions for the read or write. In order to do true binary reads from the ASIC card, your program should access ID*00 and set the special termination bits (bits 15 and 14) in register 31 to zero. The X-bits are “don’t care” bits and can be set to either 1 or 0. To do a program-enabled block read, set the PR bit after previously sending an ESCd to the terminal and setting bit 3 of DVP1.

Take care using bit 9 (the PR bit). MUX interface driver IDM00 interprets the PR bit as the keep (KP) bit, as described under CNTWD for IDM00 in Chapter 2. To get a program-enabled block read, first ensure that bit 3 of DVP1 is set (page strapping), then set the PR bit. To exercise the

keep option of the MUX interface, ensure that bit 3 of DVP1 is NOT set (line strapping), then set the PR bit.

The Z-bit defines control buffers. If Z is set to 1 on a read request, it indicates a double-buffered request and the program can write a buffer to the display before processing the data buffer. The control buffer to be sent to the display is defined by parameters PRAM3 and PRAM4.

PRAM3 is the name (or address, if using Macro) of the output buffer to be written to the display. PRAM4 is the length of the output buffer, defined as the positive number of words or the negative number of characters in the buffer. Similarly, when requesting a WRITE to the terminal and the Z-bit is set, the program is allowed to write a control buffer to the display before processing the data buffer. The control buffer to be sent to the display also is defined by parameters PRAM3 and PRAM4.

Hard-Copy Terminal Control Word CNTWD

Figure J-2 shows the format of the control word CNTWD used for a read or write request to DD*00 for hard copy terminals.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	Z	X	TR	X	X	CC	BI	Device LU					

Figure J-2. DD*00 Write Request CNTWD for Hard Copy Terminal

The NB, UE, BI and Device LU bits are as defined in Chapter 1 under the Read/Write Parameter Definitions section. The X-bits are don't care bits and should be set to zero. The Z-bit is defined above for CRT terminals.

If the CC bit is 1, the driver uses single spacing when the entire data buffer is output to the printer. If CC is 0, the driver interprets the first character of the output buffer as a carriage control character; this character will not be output to the printer. Starting with the second character, the contents of the buffer will be output to the printer. The recognized carriage control characters have the following meanings:

Control Character	Printer Action
0	Double Spacing
1	Eject One Page
*	Suppress Spacing (Overprint). The driver will not issue a line feed after a carriage return after the buffer is printed.

If TR is set to 1 and BI is 0, the driver will process data in transparent ASCII mode. When in transparent ASCII mode, you must provide carriage return (CR, 15B), line feed (LF, 12B) and form feed (FF, 14B) controls in the output buffer. (A form feed embedded in the output buffer will cause a carriage return before the form feed executes; a line feed (LF) embedded in the buffer will not cause a carriage return before the LF executes.)

The driver will not interpret the CC bit (bit 7) when in transparent ASCII mode.

The double-buffered request is useful for writing a message to the display before reading a user response to a data buffer. The following FORTRAN program fragment will illustrate:

```
      IMPLICIT INTegeR (A-Z)
      DIMENSION BUFR (40)           ! INPUT BUFFER
      DIMENSION PRMT (4)           ! OUTPUT PROMPT
      DATA PRMT / 'PROMPT>>' /
      :
      READ = 1    ! DO AN EXEC READ
C   SET THE Z-bit TO 1 FOR A DOUBLE-BUFFERED REQUEST
C   ALSO SET THE ECHO BIT AND DIRECT IT TO LU 1 (THE CRT)
      ZCRT = 010401B
      BUFLN = -80                    ! INPUT A MAX OF 80 CHARS
      PRMTLN = -8                    ! THEN OUTPUT 8 CHARS
      CALL EXEC (READ, ZCRT, BUFR, BUFLN, PRMT, PRMTLN )
      :
      END
```

The result of the program at terminal LU 1 will be:

```
PROMPT>> (DD*00 waits for you to enter up to 80 characters.)
```

BUFR and BUFLN

The I/O buffer used in the EXEC request is described by parameters BUFR and BUFLN, and are defined as in the section on Read/Write Request Conventions in Chapter 1.

PRAM3 and PRAM4 Optional Parameters

For a double-buffered request ($Z=1$), parameters PRAM3 and PRAM4 define the buffer to be output to the display before the read or write request is performed. PRAM3 is the name of the output buffer (or its address if accessed from Macro), and PRAM4 is the length of the output buffer.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each nonbuffered EXEC read/write request. The format of word 6 is the same as STAT1 returned after an EXEC 13 request; refer to the Status Request description for details.

The B-Register will contain the positive number of words or characters that were transmitted during the last nonbuffered read/write request.

DD*00 Control Request

The control request call sequence takes the form

```
CALL EXEC (3, CNTWD [,PRAM1 [,PRAM2 [,PRAM3 [,PRAM4]]]])
```

Control Request CNTWD

The CNTWD control word format for DD*00 control requests is shown in Figure J-3. The NB, UE, and DEVICE LU bits are as defined in Chapter 1. The PRAM1 through PRAM4 parameters of the EXEC call are used with the available function codes as defined in the following paragraphs.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	X	Function Code					Decive LU					

Figure J-3. DD*00 Control Request CNTWD Format

The control functions are:

Code	Function
00	Clear and Reset Device
06B	Dynamic Status
11B	Line Skipping/Formfeed
20B	Enable Primary Program Scheduling
21B	Disable Primary Program Scheduling
22B	Set System Request timeouts
23B	Disable Asynchronous Interrupts
25B	Read Terminal Straps
27B	Set User Request timeouts
40B	Enable Secondary Program Scheduling
41B	Disable Secondary Program Scheduling
42B	Get Character
44B	Modify Configuration Word
45B	Modify Trigger Character

Function Code 00: Clear and Reset Device

Performs a full reset that has the same effect as turning power on. For HP 2645 terminals, the screen and memory are cleared, then TERMINAL READY is displayed. Format mode, display functions, and all programmable functions (including the softkeys f1 through f8) are turned off or set to their default values. Device assignments are set to their default values, tapes (if present) are rewound to their load point (an end-of-data mark is not written).

For HP 2621 terminals, the display is cleared and device functions are set to their default values.

For HP 2635 terminals, print mode is set to normal, line spacing is set to the hardware setting, horizontal tabs are all cleared, a top-of-form is issued, display functions is turned off, and view mode is set on.

Function Code 06B: Dynamic Status

Causes the driver to be entered to obtain the immediate status of the device, unlike the EXEC 13 status request that returns the status of a previous EXEC request. The dynamic status information can be used to determine if the device is ready to receive a request, or to determine the device configuration before making a read/write EXEC request. When a dynamic status request is made, the driver returns the status into words 16 through 19 of the DVT. Therefore, after completion of the request, a call to RMPAR will retrieve the status information. The format of the returned status information is shown in Figure J-4.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<	x	x	0						Error Code							
DVT17<<	Transmission Log = 0															
DVT18<<	ASIC Card Status															
DVT19<<	ASIC Control Word															

Figure J-4. DD*00 Dynamic Status Format

For DVT16, bits 14 and 15 are system-defined and meaningless to application programs. The decimal error code is 0 for no error.

DVT17 returns the transmission log (always 0).

DVT18 contains the asynchronous serial interface card status. This status is valid at the completion of any control request, unsuccessful read or write request, and timeout or abort. The format of this word is shown in Figure J-5.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DV	BC	PE	FE	OE	SRD	RR	EPE	SCD	MLS	IC	CS	DM	RDL	TR	DR

Figure J-5. DVT18 Dynamic Status Word Format

DV (Data Valid):

- 1 = Received data is valid
- 0 = Interrupt occurred during data input.

BC (Break Character):

- 1 = Break character detected on received data line.
- 0 = No break character detected.

PE (Parity Error):

- 1 = Parity error detected.
- 0 = No parity error detected.

FE (Framing Error):

- 1 = Framing error detected.
- 0 = No framing error detected.

OE (Overrun Error):

- 1 = Overrun error detected.
- 0 = No overrun error detected.

SRD (Secondary Received Data) dynamic state of line:

- 1 = Mark or logic 1.
- 0 = Space or logic 0.

RR (Receiver Ready):

- 1 = Receiver not ready.
- 0 = Receiver ready.

EPE (Even Parity Enable):

- 1 = Even parity.
- 0 = Odd parity.

SCD (Special Character Detect):

- 1 = No special character recognized.
- 0 = Special character recognized.

MLS (Modem Line Status Change):

- 1 = Interrupt not caused by modem change.
- 0 = Modem status change caused flag to be set.

IC (Incoming Call):

- 1 = IC OFF.
- 0 = IC ON.

CS (Clear to Send):

- 1 = CS OFF.
- 0 = CS ON.

DM (Data Mode):

- 1 = DM OFF.
- 0 = DM ON.

RDL (Dynamic Status of the Received Data Line):

- 1 = Mark or Logic 1.
- 0 = Space or Logic 0.

TR (Transmitter Ready):

- 1 = Transmitter has transferred data and is ready for new data.
- 0 = Transmitter is busy.

DR (Data Received):

- 1 = Data has been received from device.
- 0 = No new data has been received since last STC.

Function Code 11B: Line Skip/Form Feed

For CRT terminals, PRAM1 is the only required parameter and should contain the number of lines to be skipped in one request. If PRAM1 is a negative value, the driver will convert it to positive and proceed. If PRAM1 is less than -55 or greater than +55 then only 55 lines will be skipped (the maximum allowed). If PRAM1 is 0, then 1 line will be spaced.

For hard copy terminals, PRAM1 is the only required parameter. If PRAM1 is negative, a page eject/top-of-form is performed. If PRAM1 is zero, line spacing is suppressed on the next operation only. If PRAM1 is a positive integer N, a carriage return and N line feeds (to a maximum of 55) are performed.

Function Code 20B/40B: Enable Primary/Secondary Program

Words 5 through 12 of the Driver Parameter Area contain a description of a primary and secondary program associated with a particular terminal. When a terminal key is struck, the primary program will be enabled. If the primary program does not exist, or if it is currently busy, the secondary program will be enabled, only if it exists and is not busy.

To schedule a primary program and store its name in the driver parameter area for retrieval when a key is struck, use a control request with function code 20B. The program's name is described in PRAM1 through PRAM3. For example to schedule the primary program PRIMA, use:

```
function code = 20B
PRAM1 = ASCII characters PR
PRAM2 = ASCII characters IM
PRAM3 = ASCII A and the blank character
```

PRAM4 contains an optional single-word integer to be passed to the scheduled program when it is enabled. PRAM1 through PRAM4 are stored in words 5 through 8 of the driver parameter area (see Table J-1).

PRAM4 may be retrieved in the program with a call to RMPAR (see the Programmer's Reference Manual for a description of RMPAR). The first three values returned are the interrupting LU, the parameter passed to PRAM4, and the device status.

The rules apply to scheduling a secondary program, except that a function code of 40B is used, and PRAM1 through PRAM4 are stored into words 9 through 12 of the driver parameter area.

Function Code 21B/41B: Disable Primary/Secondary Program

After a primary program has been scheduled, it may be disabled by issuing a control request with function code 21B. After a secondary program has been scheduled, it may be disabled by issuing a control request with function code 41B. To re-enable either program, another control function request (20B or 40B) must be issued with the program name specified in PRAM1 through PRAM3.

Examples:

1. No primary program exists. Schedule the primary program PRIM1 for terminal LU 1. Assume all values are integer.

```
PRAM1 = 2HPR
PRAM2 = 2HIM
PRAM3 = 2H1
CNTWD = 1 + 2000B
CALL EXEC ( 3, CNTWD, PRAM1, PRAM2, PRAM3 )
```

2. Disable the primary program enabled in Example 1, and enable a new primary program PRAM2 for LU 1 and passing it an optional integer value 9999. Assume all parameters are integer.

```
CNTWD = 1 + 2100B
CALL EXEC ( 3, CNTWD )
PRAM1 = 2HPR
PRAM2 = 2HIM
PRAM3 = 2H2
PRAM4 = 9999
CNTWD = 1 + 2000B
CALL EXEC ( 3, CNTWD, PRAM1, PRAM2, PRAM3, PRAM4 )
```

3. Enable a secondary program, SCND2, for LU 1 and pass an integer value 1. Assume all parameters are integer.

```
CNTWD = 1 + 4000B
PRAM1 = 2HSC
PRAM2 = 2HND
PRAM3 = 2H2
PRAM4 = 1
CALL EXEC ( 3, CNTWD, PRAM1, PRAM2, PRAM3, PRAM4 )
```

Function Code 22B: Set System Request Timeouts

PRAM1 will contain a value in tens of milliseconds for updating the device timeout value that was specified at generation, or by a previous set device timeout control request. The value is stored in word four of the driver parameter area and is used to update the device driver timeout value.

This timeout value only applies to system requests, not user program requests. For system requests, the minimum of this timeout value, and the value set by the "TO" operator command will be the effective timeout. This is useful in multi-terminal configurations to prevent one terminal from dominating system attention.

Example: Set the device timeout for LU 6 to 100 seconds. Assume all values are integer values.

```
PRAM1 = 10000
CNTWD = 6 + 2200B
CALL EXEC (3, CNTWD, PRAM1)
```

Function Code 23B: Disable Asynchronous Interrupts

Disables all program scheduling (primary and secondary). Any programs that were scheduled by using a control request with function code of 20B or 40B will be disabled. To re-enable program scheduling (primary and/or secondary), a control request of function code of 20B and/or 40B must be used (the program name does not have to be specified in PRAM1 through PRAM3).

Example: Enable the primary program PRIM2 and the secondary program SCND2 for terminal LU 1, and disable both by disabling asynchronous interrupts. Assume all parameters are integer values.

```
PRAM1 = 2HPR
PRAM2 = 2HIM
PRAM3 = 2H2
CNTWD = 1 + 2000B
CALL EXEC (3, CNTWD, PRAM1, PRAM2, PRAM3)
PRAM1 = 2HSC
PRAM2 = 2HND
PRAM3 = 2H2
CNTWD = 1 + 4000B
CALL EXEC (3, CNTWD, PRAM1, PRAM2, PRAM3)
.
.
.
CNTWD = 1 + 2300B
CALL EXEC (3, CNTWD)
```

Function Code 25B: Inform DD*00 of Terminal Strap Change

If your program changes a terminal line- or page-mode strap, a control request function code 25B must also be done so DD*00 will know the change has occurred. Function code 25B initiates a status request (ESC^) to the terminal on the specified LU. The terminal then reports status to the driver, including transfer mode (character, block line or block page). Only driver types 0 and 5 issue the terminal status request. Other driver types ignore function code 25B.

```
Example 1: INTeGER PAGE(3)
          DATA PAGE/15446B,71461B,42000B/!ESC&s1D
          :
          CALL EXEC(2,LU,PAGE,-5)           !Put terminal into page mode
          CALL EXEC(3,2500B+LU)           !Inform driver of strap change
          :
```

```
Example 2: INTeGER LINE(3)
          DATA LINE/15446B,71460B,42000B/!ESC&s0D
          :
          CALL EXEC(2,LU,LINE,-5)          !Put terminal into line mode
          CALL EXEC(3,2500B+LU)          !Inform driver of strap change
```

If your program issues function code 25B to a MUX terminal, the driver will first flush the MUX input buffers if bit 5 of the terminal configuration word is set to 1 (see also DVP1 under function code 44B).

Function Code 27B: Set User Request Timeouts

Function code 27B changes the timeout on a device in the same way that the operator TO command does. Code 27B is effective for user program requests in contrast to code 22B, which only applies to system requests.

```
:CN,LU,27B,P1           Where P1 = timeout in tens of milliseconds
```

Substituting values:

```
:CN,30,27B,2000        Changes the timeout of lu 30 to 20 seconds.
```

This is equivalent to :TO,30,2000; however, function code 27B can be done programmatically, whereas the TO operator command cannot.

If you change the timeout programmatically using function code 27B, you can obtain the old value by doing the EXEC 3 with the NB bit (nonbuffered bit 14) set, then calling RMPAR. The old value before changing to the new value will be in DVT17, the second RMPAR buffer word.

Example:

```
CALL EXEC(3,42700B+LU,1000)      ! Change timeout to 10 seconds
CALL RMPAR(IBUF)
IOLD = IBUF(2)                   ! Old timeout is in second word
```

Note that ENQ/ACK must be enabled to make timeouts work on write requests. The system issues an ENQ to which the device must ACK within the specified timeout period.

Function Code 42B: Get Character

This function read one ASCII character from the display. When the request completes, DVT17 and the B-Register will contain the ASCII character in the low byte and the transmission log in the high byte. If the device times out before the character is transmitted, a null character will be transmitted, and the transmission log will be set to zero.

If PRAM1 = 0 the character can be echoed to the display. If PRAM1 is not zero, the character is not echoed.

The NB bit should always be set since a buffered control request would complete immediately. The calling program will be swappable during this request.

Example: Get one ASCII character from terminal LU 1, and allow the character to be echoed to the display.

```
CALL EXEC ( 3 , 4201B , 0 )
```

The B-Register after the call reads an ASCII "A" as shown below:

B-Register<<	0 0 0 0 0 0 0 1	0 1 0 0 0 0 0 1
	Trans. Log = 1	ASCII A = 101B

Figure J-6. B-Register Format

Function Code 44B: Modify Configuration Word

This function modifies the terminal configuration word (DVP1) with the new word contained in PRAM1. This word is defaulted to No Page Mode capability at generation time. This allows DD*00 to trigger a read transfer by sending a DC1 character when it is ready to receive the data. In Page Mode (bit 15 set in DVP1), 'ESCDC1' is sent at the completion of a write transfer, thus allowing the ESC to mark the beginning of the page and the DC1 character to trigger a read transfer for an asynchronous interrupt or an actual page mode transfer of data.

Function Code 45B: Modify Trigger Character

This function modifies the Trigger Character (DVP3) with the new character contained in the upper byte of PRAM1. Driver parameter 3 (DVP3) is defaulted to a DC1 in the upper byte (10400B) at generation. This allows DD*00 to trigger a read transfer by sending a DC1 character when it is ready to receive data. This character may be modified to contain a specialized trigger character required for a specific device or a null for no trigger at all.

DD*00 Status Request

The status request call sequence takes the form

```
CALL EXEC (13, CNTWD, STAT1 [,STAT2 [,STAT3 [,STAT4]])
```

Control Word CNTWD

Figure J-7 shows the control word (CNTWD) format for an EXEC status request for devices supported by DD*00. The returned status information will be discussed in the following paragraphs.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	0				LU							

Figure J-7. DD*00 Status Request CNTWD Format

STAT1 and STAT2 Status Parameters

The format of the returned status parameters (STAT1/STAT2) is shown in Figure J-8.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT1 <<	AV		Device Type = 0,5,12				^D	0	^D	0			TO	E		
STAT2 <<	AV		Interface Type = 00				0		I/O Select Code							

Figure J-8. STAT1 and STAT2 Status Words Format

For STAT1, the AV and DEV.TYPE bits are as defined in Chapter 1 under Status Parameter Definitions. The E-bit is an error flag; if set, it indicates an error in DVT16. If set, the ^D bits indicate that a CONTROL D (end-of-transmission, 4B) was entered at the keyboard (bit 5 indicates EOF and bit 7 indicates EOM). Only valid for normal ASCII mode. The TO bit is set to 1 to indicate that a timeout occurred.

For STAT2, the INTERFACE TYPE is zero, corresponding to the asynchronous serial interface card. The I/O SELECT CODE and AV bits are defined as in the section on Status Parameter Definitions in Chapter 1.

STAT3 and STAT4 Status Parameters

If Z is zero, STAT3 contains the first word of the driver parameter area, and STAT4 contains the second word of the driver parameter area.

If Z is set to 1, STAT3 is the buffer to return the driver parameter area, and STAT4 is the length of the STAT3 buffer.

Driver Parameter Areas

Table J-1 defines the 12 words of the driver parameter area for DD*00.

Table J-1. DD*00 Driver Parameter Area

Word	Definition
DVP1	Terminal Configuration
DVP2	Suppress Space Flag (Bit 0)
DVP3	Trigger Character Before Read
DVP4	Device Timeout Value
DVP5	Scheduled Primary Program Name (Characters 1 and 2)
DVP6	Scheduled Primary Program Name (Characters 3 and 4)
DVP7	Scheduled Primary Program Name (Characters 5 and 6)
DVP8	Optional Parameter (Primary Program)
DVP9	Scheduled Secondary Program Name (Characters 1 and 2)
DVP10	Scheduled Secondary Program Name (Characters 3 and 4)
DVP11	Scheduled Secondary Program Name (Characters 5 and 6)
DVP12	Optional Parameter (Secondary Program)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PM	ASC	T	0				FB	SE	PS	LF	FF	ENQ			

Figure J-9. Terminal Configuration Format in DVP1

Bit 15, PM, page marker bit

- 1 = Writes escape underscore on end of each write.
- 0 = Normal.

Note This bit tells the driver to append a non-displaying terminator at the end of each write. It is set using an EXEC 3 with function code 44B.

Bit 14, ASCII bit

- 1 = 8 data bits (Katakana, Roman 8)
- 0 = 7 data bits (Normal, vs ASII)

Bit 13, Termination bit

- 1 = Binary read only terminated by char count
- 0 = Binary read terminated by count or "CR"

Bits 12 - 6, Not used

Bit 5, FB, Flush Input Buffers (Multiplexer):

- 1 = Flush both MUX input buffers before issuing a terminal status request (CN 25B).
- 0 = Do not flush MUX input buffers before issuing a terminal status request.

Bit 4, SE, Suppress Transmission Error Message (Multiplexer):

- 1 = Suppress issuing transmission error message in case of a buffer overflow on this port.
- 0 = Do not suppress issuing transmission error message in case of a buffer overflow on this port.

Bit 3, Page strapping bit

- 1 = page strapping
- 0 = line strapping

Bit 2, LF, controls Line feed for special character CR, if echo bit set in control word (required for multiplexer operation):

- 1 = Disable line feed (the MUX firmware takes care of line feed)
- 0 = Enable line feed

Bit 1, FF, contains status of terminal Form Feed:

- 1 = Form feed enabled for terminal.
- 0 = Form feed not enabled for terminal.

Bit 0, ENQ, contains status of ENQ/ACK Handshake:

- 1 = ENQ/ACK handshake enabled.
- 0 = No ENQ/ACK handshake.

Do not use Bit 0 to enable ENQ/ACK when using DD*00 with MUX driver IDM00. Bit 7 of CN30B should be used on the MUX.

DD*00 Extended Status

The extended status can be obtained by making a call to RMPAR after an unbuffered EXEC read, write, or control request to determine the status of the request. The format of the returned status is the same as the status returned for a dynamic status request (function code 06B). The difference is that the dynamic status request returns the immediate status of the device. The extended status request, however, returns the status of the last EXEC I/O or control request made to the device. Refer to the Function Code 06B section for the format of the extended status.

CTU Device Driver DD*20

DD*20 is the cartridge tape unit (CTU) device driver to drive devices containing CTUs (HP 264x terminals). DD*20 provides the software interface to handle any device concerns, while the asynchronous serial interface card ASIC driver, ID*00, performs the actual I/O instructions to the interface card. Access to DD*20 is achieved via standard EXEC requests as discussed in the following sections.

CTU Read/Write Conventions

The BLOCK MODE key must be latched out (in the UP position) for CTU operation.

A read request will input, via the driver, either a word or character string (depending on the BUFLN parameter) to a buffer of specified length. If the buffer is filled before an End-Of-Record (EOR) mark is read, the driver ignores the remaining data and stops the CTU at the first EOR it reads. (Note that this condition usually results in the remainder of the data appearing on the terminal display as if it had been typed at the keyboard.) If an EOR mark is read before the specified buffer is filled, the data transfer will stop at the EOR mark. The CTU will skip one record if the buffer length is specified as zero.

The maximum buffer length allowed is 128 words (256 bytes) for binary writes, or 127 words (254 bytes) for ASCII writes. If a larger buffer is specified, or if the length is specified as zero for a binary write, the driver will reject the request and the system will issue an error code of IO07.

The driver appends CRLF to ASCII output records and removes it from ASCII input records.

If the End-of-Tape (EOT) mark is sensed during a write operation, an End-of-Data (EOD) mark is automatically written to the tape, and the driver will terminate the current record being written. Further attempts to write will cause DD*20 to issue an EOT error message.

Similarly, the driver will read the current record if an EOD mark is sensed during a read operation. Either condition, EOD or EOT, will be reported in the DVT18 status word (see the status information in this section). Further attempts to read past the EOT mark will cause the driver to issue an EOT error message.

Read requests will be rejected if the tape is at the EOD mark. However, the EOD mark can be overwritten with additional data or a file mark, unless the tape is at the EOT mark.

DD*20 Read/Write Request

The call sequence of the read or write request is

READ Request: CALL EXEC (1, CNTWD, BUFR, BUFLN)

WRITE Request: CALL EXEC (2, CNTWD, BUFR, BUFLN)

Control Word CNTWD

Figure J-10 shows the control word (CNTWD) format used by DD*20 for a CTU read or write request. The NB (Nonbuffered mode), UE (User Error), BI (Binary data), and DEVICE LU bits are defined in the Read/Write Parameter Definitions section in Chapter 1.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	0	TR	0	DH	0	BI	Device LU					

Figure J-10. DD*20 Read/Write Request CNTWD Format

The DH (Disable Handshake) bit is required for multiplexer interface driver IDM00. All other bits should be set to 0. All other bits should be set to 0.

Setting the DH bit disables the ENQ/ACK handshake between the multiplexer and the CTU, which is required for multiplexer operation. DD*20 sets this bit for a write request. Terminal interface driver ID*00 ignores the DH bit.

BUFR and BUFLN

The I/O buffer used in the EXEC request is described by parameters BUFR and BUFLN as defined in the Read/Write Parameters section of Chapter 1.

A- and B-Register Contents

The A-Register will contain word 6 of the device table after each nonbuffered EXEC read/write request. The format of word 6 is the same as STAT1 after an EXEC 13 status request, as discussed below.

The B-Register will contain the positive number of words or characters transmitted during the last nonbuffered read/write request.

DD*20 Control Request

The call sequence for the control request is

```
CALL EXEC (3, CNTWD [ ,PRAM1 ])
```

When programming a CTU control request, the following rules should be noted:

1. If the terminal is page-strapped (a full page of memory is to be transferred at one time), the BLOCK MODE key must be latched out (in the UP position) for CTU operation.
2. When a control function is executing, the keyboard is locked out (except for the return key) during device-to-device transfer operations. The return key allows you to perform an interrupt to the CPU and display what was being read or written to the CTU at the instant the return key is pressed. The request will then be restarted.

Control Word CNTWD

Figure J-11 shows the format of the control word for a DD*20 control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	NB	UE	0	Function Code						Device LU					

Figure J-11. DD*20 Control Request CNTWD Format

The NB (Nonbuffered mode), UE (User Error), and DEVICE LU bits defined in the control word are as described in the Control Request Parameter Definitions section of Chapter 1. The Z-bit (bit 12) is ignored by the driver and should be set to 0. The following control functions are available for DD*20:

Code	Function
00	Clear and reset device (Rewind)
01B	Write End-of-File (EOF)
02B	Backspace One Record
03B	Forward Space One Record
04B	Rewind Tape
05B	Rewind Tape
06B	Dynamic Status
10B	Write EOF if not previously written or not at load point
13B	Forward Space One File
14B	Backspace One File
26B	Write End-of-Data Mark (EOD)
27B	Locate Absolute File on Tape

Function Code 00: Clear and Reset Device

Function code 00 places the CTU in a known state by rewinding the tape to the beginning.

Function Code 01B: Write End-of-File

Function code 01B places an end-of-file mark at the current position of the tape. If a tape is to be repositioned after a write to the tape, you should request that an end-of-file mark (EOF) be written to the tape, to ensure that the data just written will not be appended to another file or overwritten.

When a new file is to be appended to the current position of a tape, and the current position does not have an EOF, an end-of-file should be recorded before starting the new file. If an EOF is not written, the new data will be appended to the end of the last file. However, if the current position is the beginning of the tape, the end-of-file mark is not necessary.

Function Codes for Tape Positioning

The following function codes can be used for tape positioning:

Code	Function
02B	Backspace one record
03B	Forward space one record
04/05B	Rewind
13B	Forward space one file
14B	Backspace one file
27B	Locate absolute file

A rewind (04B, 05B), backspace one record (02B), or backspace one file (14B) control request does not cause any action if the CTU is at the load point (beginning of tape). The load point condition can be determined by examining the status returned by a dynamic status request (refer to Dynamic Status, function code 06B).

For a backspace operation (02B, 14B), if an end-of-file mark is the last record encountered while backspacing, the tape will be spaced forward so that it is positioned immediately after the end-of-file mark (that is, just before the first record of the file). The end-of-file status bit also will be set at this time (see the EXEC 13 status information in this section).

When positioning to an absolute file on a tape (function code 27B), PRAM1 will contain the file number to be located, an integer value between 0 and 256. The tape will be positioned just before the first record of the specified file (that is, just after the end-of-file mark of the previous file).

The following control requests are examples that allow positioning of a cartridge tape. Figure J-12 illustrates the points at which the tape will be positioned (for these examples) after each request.

Forward Space One Record (function code 03):

If at A, will move to B.

If at B, will move to C.

Backspace One Record (function code 02):

If at B, will move to A.

If at C, will move to A

Forward Space One File (function code 13B):

If at A, will move to B.

If at B, will move to E.

Backspace One File (function code 14B):

If at C, will move to A.

If at E, will move to D.

Locate Absolute File (function code 27B):

If locating file 2, will move to E.

If locating file 3, will move to G

If locating file 1, will move to B.

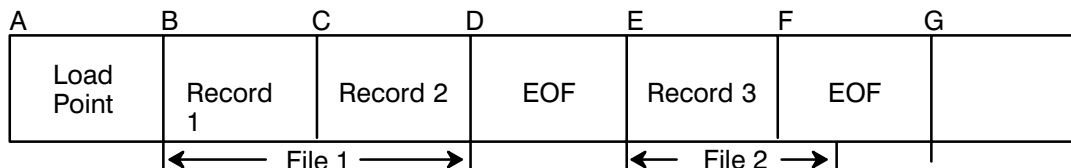


Figure J-12. Cartridge Tape Positioning

When using any control requests for the CTU, the transmission log returned in the B-Register is meaningless.

Function Code 06B: Dynamic Status

Function code 06B returns the current status of the device. This function differs from the EXEC 13 request because it reads the current status, not the stored status of the device. Dynamic status can be used to see if a device is ready for a request. The dynamic status request is returned to DVT16 to DVT19, where it can be read by a call to RMPAR. The format of the returned status information is shown in Figure J-13.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<		X	0						Error Code							
DVT17<<	Transmission Log = 0															
DVT18<<	Device Status															
DVT19<<	ASIC Control Word															

Figure J-13. Extended Status Words DVT16 to DVT19

In DVT16, bits 14 and 15 are used by the system. The decimal error codes returned in bits 0 through 5 are:

- 0 = No error.
- 3 = Device timeout.

DVT17 contains the transmission log.

DVT18 contains the dynamic status. The format of DVT18 is shown in Table J-2.

DVT19 contains a copy of the asynchronous serial interface card (ASIC) control word, which is normally meaningless to application programs. Refer to the section covering the ASIC driver ID*00 for details.

Table J-2. DVT18 Dynamic Status Word Format for DD*20 CTU Driver

Bit	Identifier	Setting
15–12	--	0
11	EOF (End-of-File)	1 = At end of file (positioned after EOF mark) 0 = Not at end of file
10	LP (Load Point)	1 = At load point 0 = Not at load point
9	EOT (End-of-Tape)	1 = At end of tape 0 = Not at end of tape
8	WE (Write Error)	1 = Error during last write 0 = No error
7	CE (Cmd. Executed)	1 = Last command performed 0 = Last command aborted
6	WP (Write Protected)	1 = Tape write protected 0 = Not write protected
5	RE (Read Error)	1 = Error during last read 0 = No error during read
4	TB (Tape Busy)	1 = Device busy 0 = Device not busy
3	SE (Soft Error)	1 = Recoverable error during read/write 0 = No error encountered
2	HE (see note) (Hard Error)	1 = Read/write aborted 0 = No error encountered
1	EOD (End-of-Data)	1 = End-of-data mark found 0 = End-of-data not found
0	TI (Tape Installed)	1 = Tape installed 0 = Tape not installed

Note: A Hard Error occurs when the terminal attempts a read or write request to/from tape more than ten times, then aborts (i.e. HE = 10 soft errors).

Function Code 10B: Write End-of-File

Function code 10B causes the end-of-file (EOF) to be written on the tape if there is not an end-of-file mark directly preceding the current position of the tape or if the device is not at the load point.

Function Code 26B: Write End-of-Data

Function code 26B causes an end-of-data mark (EOD) to be written at the current position of the tape. With this control request, you can record data followed by an end-of-data mark without writing an EOF mark. To find the EOD mark if the tape has been repositioned, a control request (function code 27B) can be used. Specifying PRAM1 as a value greater than the highest numbered file will position the tape just prior to the EOD mark. If data is then added, the EOD mark will be overwritten.

If the last function performed on the cartridge was a record operation, an EOD mark will automatically be written before a locate or rewind operation is performed.

PRAM1 Optional Parameter

PRAM1 is an optional parameter that will be used for the Write End-of-Data and Locate Absolute File control requests (function codes 26B and 27B).

DD*20 Status Reporting

The status request call sequence takes the form

```
CALL EXEC (13, CNTWD, STAT1 [,STAT2 [,STAT3 [,STAT4]]])
```

Control Word CNTWD

Figure J-14 shows the control word format for an EXEC 13 status request for devices supported by DD*20.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z				0						LU		

Figure J-14. DD*20 Status Request CNTWD Format

STAT1 and STAT2 Status Parameters

Figure J-15 shows the format of the returned status parameters (STAT1/STAT2).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV							EOF	DB	EOM	BOM	SE	WP	OF	E	
AV															I/O Select Code

Figure J-15. STAT1 and STAT2 Status Words Format

In STAT1, the AV (device availability) and DEVICE TYPE bits are defined in the Status Request Parameter Definition section in Chapter 1. The status codes returned in bits 0 through 7 are:

- EOF (End-Of-File): Tape is currently positioned at end-of-file mark.
- DB (Device busy): Tape is performing a function that prevents other operations from starting (that is, tape rewind).
- EOM (End-of-Medium): Tape is currently positioned at the end-of-tape.
- BOM (Beginning Of Medium): Tape is currently positioned at the load point (beginning of medium).
- SE (Soft Error): Recovered from read/write error.
- WP (Write Protected): Tape is write protected.
- OF (Off Line): Tape not installed.
- E (Error): Set by system if driver sets any error code in DVT16.

In STAT2, the interface type is 00, corresponding to the asynchronous serial interface card. All other bits are as defined in Chapter 1.

STAT3 and STAT4 Status Parameters

If the Z-bit is zero, STAT3 is the first word of the driver parameter area. If the Z-bit is 1, STAT3 is the buffer to return the driver parameter area, and STAT4 is the length of the STAT3 buffer.

The driver parameter area for the DD*20 CTU driver is returned in STAT3. It is equal to 1 if the left CTU was requested, or 2 if the right CTU was requested.

DD*20 Extended Status

As discussed in the Extended Device Status section of Chapter 1, the extended status of a request can be obtained by making a call to RMPAR after an unbuffered read, write, or control request. The format of the returned status is the same as the status returned for a dynamic status request. However, the dynamic status request returns the immediate status of the device while the extended status request returns the status of the last EXEC I/O or control request made to the device.

Refer to the control request subsection (function code 06) for the format of the extended status parameters DVT17 through DVT19 for DD*20 devices. The format of the DVT16 word for extended status is shown in Figure J-16.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<	x	x	0						Error Code							

Figure J-16. DVT16 Status Word Format

DVT16 bits 14 and 15 are used by the system. The decimal error codes returned in bits 0 through 5 are:

- 0 = No error.
- 2 = Tape not installed.
- 3 = Device timeout.
- 5 = Read error during last read.
- 6 = Write protected. Tape record switch set to write protect.

IDM00 (HP 12040A) 8-Channel Asynchronous MUX Interface Driver

Note: This section describes IDM00 software before rev. code 2301.

This version of IDM00 drives the HP 12040A 8-Channel Asynchronous Multiplexer Card (MUX). It supports device drivers DD*00 and DD*20, and is the functional equivalent of interface driver ID*00. Refer to the *HP 12040A 8-Channel Asynchronous Multiplexer Interface Reference Manual*, part number 12040-90123, for hardware information.

IDM00 performs the actual I/O instructions to the MUX, so it is interface-dependent and device-independent. The driver is accessed via EXEC requests. All I/O transfers are under interface card DMA control.

IDM00 Read Request

The call sequence for the read request is

```
CALL EXEC (1, CNTWD, BUFR, BUFLN [ ,PRAM3 [ ,PRAM4 ] ] )
```

Note All read/write requests use the character format set up by Control Request 30B. The terminal must be strapped accordingly.

Control Word CNTWD

The control word CNTWD is shown in Figure J-17.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	x	TR	KP	EC	X	BI	Device LU					

Figure J-17. IDM00 Read Request Control Word CNTWD

Bit 9 is the Keep Bit. If this bit is set to 1, data past the end of the user buffer is saved until the next read request is encountered. All other bits are standard I/O request conventions as described in Chapter 1, in the description of CNTWD.

BUFR and BUFLN

BUFR is the address of the user buffer to be used in the read request. BUFLN is the length of the user buffer to be used in the read request. If BUFLN is a positive number, the length is in words; if BUFLN is negative, the length is in characters.

PRAM3 Parameter

Bits 15 and 14 select the end-of-transfer character:

00 – no end-of-transfer character. Transfer ends when the buffer is full.

01 – transfer ends on carriage return (CR), control R (DC2), control D.

(EOT), or control ^ (RS). Note: When using the MUX driver, a control D flushed the buffer. When using the ASIC driver, a control D returns the buffer intact.

10 – transfer ends on control ^ (RS).

11 – transfer ends on carriage return.

Bit 11 is the echo control character:

1 – echo on.

0 – echo off.

PRAM4 Optional Parameter

Bits 15 through 8 define the prompt character. If it is not null, then the prompt character is output before the read is started.

Bits 7 through 0 define the number of characters to ignore at the end of a BINARY read. If nonzero, the MUX interface card is configured to read that many extra characters are discarded by the driver. This is used by the CTU driver DD*20 to prevent asynchronous interrupts when the user buffer size is smaller than the size of the record stored on tape.

IDM00 Write Request

The call sequence for the write request is

```
CALL EXEC ( 2, CNTWD, BUFR, BUFLN)
```

Note For all read/write (I/O) requests, the I/O transfers use the character format set up by Control Request 30B. The terminal must be strapped accordingly.

Control Word CNTWD

Bit 6 is for binary data transfer:

- 1 – suppress CRLF and ignore bit 7
- 0 – refer to bit 7 for CRLF control

Bit 7 controls CRLF insertion:

- 1 – suppress addition of CRLF
- 0 – append CRLF to buffer (if bit 6 is also 0).

Bit 8 is for the ENQ/ACK handshake:

- 1 – disable handshake, this write only
- 0 – allow handshake if enabled on card

Bit 11 is for NDT (Non-Display Terminator, ESC _) control:

- 1 – output NDT after printing user buffer
- 0 – do not output NDT

BUFR and BUFLN

BUFR is the address of the user buffer to be used in the write request. BUFLN is the length of the user buffer to be used in the write request. If BUFLN is a positive number, the length is in words; if BUFLN is negative, the length is in characters.

IDM00 Status Request

Whenever an I/O request is made, certain status information is placed in DVT (Device Table) words 16 and 18. The formats of the status returned in DVT words 16 and 18 are as follows:

DVT16 bits 5 through 0 return the error code:

- 3 – timeout.
- 5 – transmission error.
- 77B – request aborted by the system.

DVT18 (read requests only):

Bit 7 = 1 special character was NOT detected (binary read).
= 0 special character was detected (ASCII read).

Bit 0 = 1 data was received.
= 0 data was NOT received.

IDM00 Control Requests

The call sequence for the control request is

```
CALL EXEC ( 3 , CNTWD , PRAM1 )
```

Control Word CNTWD

Bits 11 through 6 contain the function code that defines the control function to be performed.

Bits 5 through 0 define the LU (logical unit number) to be controlled.

The control functions, right-justified in bits 11-6, are:

- 06B Obtain Dynamic Status
- 23B Control Program Scheduling
- 26B Flush Input Buffer
- 30B Set Port ID
- 33B Configure Driver Responses
- 37B Set Read Type

Function Code 6B: Dynamic Status

This control function returns the requested port status in DVT16, DVT17, and DVT18. The character length of any “typed-ahead” data (refer to the section on Type-Ahead) is placed in DVT word 17 (the transmission log). The formats of the status returned in DVT words 16, 17, and 18 are as follows:

DVT16 bits 5 through 0 return the error code:

- 3 – timeout.
- 5 – transmission error.
- 77B – request aborted by the system.

DVT17 indicates the length of the type-ahead data in characters.

DVT18 indicates card status:

- Bit 15 = 1 type-ahead data available on this port.
- = 0 no type-ahead data is available on this port.

Bits 14-0 are not used, and are always zero.

Function Code 23B: Control Program Scheduling

This function sets or resets the Program-Schedule-Enabled flag in the interface driver according to the value of PRAM1:

- PRAM1 = 0 enable scheduling.
- = 1 disable scheduling.

The device driver is entered to schedule the program only if these three conditions are met:

1. Scheduling is enabled.
2. The interface driver is not expecting data from the port (a read operation is not in progress).
3. A. The port is not in type-ahead mode and a key is pressed;
B. The port is in type-ahead mode and the BREAK key is pressed (refer to function 33B regarding type-ahead and the BREAK key);
C. The port is in type-ahead mode with program scheduling upon data available enabled, and a valid record is received.

Function Code 26B: Flush Input Buffer

This function causes the MUX interface to clear any data from the active port input buffer that might have accumulated in the type-ahead mode. PRAM1 determines the nature of the clear:

- PRAM1 = 1 clear all input buffers on this port.
- 0 = 0 clear only the port's active input buffer.

Function Code 30B: Set Port ID

This function establishes a logical connection between the LU to which it is directed and the physical device connected to the MUX interface. It also configures the port according to the value in PRAM1.

Note Control function 30B must be issued before any other request is given to that port. Any requests given to a port prior to function code 30B is ignored by the interface driver. Also, the port number given should specify a unique device on the MUX interface. If a conflict exists, then the new assignment takes precedence and the old assignment is lost.

The value of PRAM1 for this function code is:

Bits 15-14 are the number of bits per character:

- 00 – 5 bits per character
- 01 – 7 bits per character
- 10 – 6 bits per character
- 11 – 8 bits per character

Bit 13 is reserved, and should be zero.

Bit 12 is the baud rate generator used for this port:

- 1 – baud rate generator #1
- 0 – baud rate generator #0

Note Refer to the hardware/installation manual for setting bit 12. The way in which the customer cable has been wired determines how to set this parameter.

Bits 11-10 indicate the number of stop bits:

- 00 – reserved; do not use this bit pattern.
- 01 – 1 stop bit
- 10 – 1-1/2 stop bits
- 11 – 2 stop bits

Bits 9-8 select the type of parity:

- 00 – no parity
- 01 – odd parity
- 10 – no parity
- 11 – even parity

Bit 7 is for the ENQ/ACK handshake:

- 1 – enable handshake
- 0 – disable handshake

Bits 6 to 3 represent the baud rate:

- 0 – no change
- 1 – 50
- 2 – 75
- 3 – 110
- 4 – 134.5
- 5 – 150
- 6 – 300
- 7 – 1200
- 10B – 1800
- 11B – 2400
- 12B – 4800
- 13B – 9600
- 14B – 19200
- 15B – reserved; do not use.
- 16B – reserved; do not use.
- 17B – reserved; do not use.

Note 19200 baud is not supported on all 8 ports simultaneously. The maximum combined baud rate for all 8 ports is 76,800 baud.

A baud rate parameter of ZERO in bits 6-3 results in no change to any of the device's parameters (baud rate, parity, stop bits, etc.).

Bits 2-0 are the port number of this device:

- 0 – Port 0
- 1 – Port 1
- 2 – Port 2
- 3 – Port 3
- 4 – Port 4
- 5 – Port 5
- 6 – Port 6
- 7 – Port 7

Function Code 33B: Configure Driver Responses

This function sets up various parameters and responses strictly for the interface driver. NONE of these parameters are sent to the MUX interface itself. These parameters are defined by the value of PRAM1.

Note All of the two-bit fields defined below default to a value of 01 at system reboot.

Bits 15-14 are reserved, and should be zero.

Bits 13-12 define the type-ahead feature to be used:

- 00 – No change is made in this field.
- 01 – No type-ahead. System attention is gained by pressing any key when no read is pending.
- 10 – Enable type-ahead. If data is received without a pending read, it is saved on MUX interface until a read request occurs. System attention is gained by hitting BREAK key only, unless type-ahead scheduling is enabled.
- 11 – Reserved.

Bits 11–10 define the action to be taken by the interface driver when type-ahead data becomes available:

- 00 – No change is made in this field.
- 01 – Bit 15 of DVT18 is set to 1.
- 10 – Program scheduling is attempted and bit 15 of DVT18 is set to 1.
- 11 – Reserved.

Bits 9-8 define the action to be taken by interface driver when the BREAK key is pressed:

- 00 – No change is made in this field.
- 01 – Attempt program scheduling, if it is enabled.
- 10 – Clear data on ALL ports, then attempt program scheduling, if it is enabled.
- 11 – Reserved.

Bits 7-6 tell the interface driver how to control sending of read configuration information to MUX interface card:

- 00 – No change is made in this field.
- 01 – Always send configuration information on each EXEC read operation AND on each control function 37B command.
- 10 – Only send configuration information on control function 37B commands OR on command from the device driver.
- 11 – Reserved.

Note Since Device Drivers DD*00 and DD*20 send configuration information to the interface driver in the EXEC read parameters, operation with these device drivers is only supported with this two-bit field set to 01.

Bits 5-0 are reserved, and should be zero.

Function Code 37B: Set Read Type

This function sends configuration information to the MUX interface card that will be used in subsequent read (EXEC 1) requests. Normally this information is provided by the interface driver as directed by the function field (bits 11-6) in the EXEC 1 request. This control function allows you to either override the device driver defined values or to configure a read operation on the MUX interface without executing a read request (useful in type-ahead initialization).

Note If PRAM1 bit 7 in control Function 37B is zero, then any read operation will overwrite any of the values set here with the values in the function field of the EXEC 1 request.

The format for the values in PRAM1 are as follows. The bits, when set to 1, enable the following:

- 15: End Transfer on Carriage Return (CR)
- 14: End Transfer on Cntl ^ (RS)
- 13: End Transfer on Cntl D (EOT)
- 12: End Transfer on Cntl R (DC2)
- 11-10: (reserved)
- 9: Enable Input Data Editing
- 8: Enable Input Data Echo
- 7-0: (reserved)

IDM00 Type-Ahead

Type-ahead is the ability of a system to accept data from the user's terminal or device before it is asked for. The MUX card, being buffered, can hold up to two lines of text per port in memory without needing a place in the system to hold it. The advantages of type-ahead over the RTE convention are prevention of data loss and unexpected system prompts when keys are accidentally struck.

While in type-ahead mode, the driver leaves a read request pending on the CARD (not the IFT) at all times. This read allows the user to enter data into the MUX card even though the system does not have a read pending. Upon receiving a record, the card will interrupt the host telling it that a buffer of data is available. If no request has appeared on that port, a flag is set in the status and the driver returns to the system waiting for something to happen. When the request is issued, the driver can get the data from the MUX card and return to the user.

Since keyboard characters would be buffered on the card, system attention cannot be gained by pounding on the terminal keys. The BREAK key, however, is not buffered, and can be used for this purpose. In addition, if type-ahead scheduling is enabled, the user can enter a system command or any other valid keyboard entry (followed by a carriage return) without first having to get the system's attention. The system will execute the command as soon as it is read.

Since multiline type-ahead is possible, two different type-ahead modes are implemented. Full type-ahead, as described above, would cause successive read requests to fetch successive lines of text from the MUX card. This mode would be useful, for example for text editing, using DBUGR, etc. One could type as far ahead of the data processing as allowed by available multiplexer buffer memory (two records).

In situations where system response could alter a user's next command (FMGR error messages, for example) a full multiline type-ahead may cause problems. The following may help illustrate the problem:

User types: *ST,FILE,8*

While tape is moving, user types: *PU,FILE*

Tape runs out; system downs the device.

User hits BREAK, system issues prompt and read.

Driver reads PU command from card buffer and system tries to execute it.

In the above example, the user merely gets back an "ILLEGAL COMMAND" from the system the first time the request for system attention is made. It is possible, however, for the commands stored on the card to have a disastrous effect on the system.

A solution to the above problem is to program the driver to cancel all card data upon receiving a BREAK interrupt. This preserves the multiline type-ahead feature, and reduces the chance of data being read by the wrong process.

Another solution to the above problem is to issue a Flush Input Buffer request (code 26B) with a "fatal" error message to clear extra commands before they are read. For a description of the driver configuration options, see Control Function 33B.

Another advantage of type-ahead is that programs can prompt for responses before they begin processing the last response. While the user types the new response, the program processes the last one. This increases throughput, especially if the program can process each response in less than the time it takes the user to type the next one.

The forms of type-ahead just described are also useful in non-terminal device communication. The buffering on the card eliminates the need for stacking two class read requests on an LU to prevent data loss, which reduces program size and complexity, and the need for a large System Available Memory (SAM) area.

When data is available on the multiplexer card, and there is no pending request to accept it, a bit will be set in the status word and program scheduling attempted. Should the user program decide it does not want the data, it can issue a Flush Input Buffer (Control Function 26B) to remove the data.

In the non-type-ahead mode of operation, the subsystem will appear to act the same as current RTE terminal drivers. The driver, when a port is inactive, will leave an "interrupt on any character" read pending on the card so as to be informed when a key is struck. The appropriate action (system attention, program schedule, etc.) will then be taken.

Since the MUX performs all data processing on input or output data, it needs some configuration information to know just what operation to perform. For output, this consists of whether to add a CRLF (Carriage Return/Line Feed) and whether the ENQ/ACK handshake is enabled. Input (Read) configuration information is what has been referred to as a "Record". This consists of the following:

1. Echo ON/OFF
2. Editing ON/OFF (This includes Backspace and Delete)
3. Terminating Condition:
 - a. Count: Terminate after a defined number of characters
 - b. Character: Terminate on one or more of the following characters:
 - 1) CR (Carriage Return)
 - 2) RS (Cntl ^)
 - 3) DC2 (Cntl R)
 - 4) EOT (Cntl D)

Each of the two input buffers for each port on the MUX can hold one Record, or part of a Record, if the Record is longer than 254 characters. This limitation holds true even if the Records are only a few characters: Records are not packed into the buffers. This places the two-record limitation on type-ahead. If a third Record is received before space is made available on that port of the MUX card, then an overflow condition occurs. This results in the first Record left exactly as it was and the second Record will report an overflow condition (Record Length = 0, Status = 5).

IDM00 (HP 12040B/C) 8-Channel Asynchronous MUX Interface Driver

Note: This section describes IDM00 software Rev. Code 2301 and later.

This version of IDM00 drives the HP 12040B/C 8-Channel Asynchronous Multiplexer Card (MUX) and the HP 37222A Modem Card. It supports device drivers DD*00 and DD*20, and is the functional equivalent of interface driver ID*00. Refer to the *HP 12040B/C 8-Channel Asynchronous Multiplexer Interface Reference Manual* for a multiplexer hardware description and to the *HP 37222A Modem Card Reference Manual*.

IDM00 performs the actual I/O instructions to the MUX, so it is interface-dependent and device-independent. IDM00 is accessed via EXEC requests, and all I/O transfers are under interface card DMA control.

IDM00 Read Request

The call sequence for the read request is

```
CALL EXEC (1, CNTWD, BUFR, BUFLN, PRAM3 [ ,PRAM4 ])
```

Note All read/write requests use the character format set up by function code 30B. The terminal must be strapped accordingly.

Control Word CNTWD

The read request control word CNTWD is shown in Figure J-18.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	x	TR	KP	EC	x	BI	Device LU					

Figure J-18. IDM00 Read Request Control Word CNTWD

KP = keep bit (bit 9 of CNTWD), KP is unique to the HP 12040B multiplexer. When it is set to 1, data beyond the end of the user buffer is saved until the next read request. When using KP, make sure that the terminal straps are not set for page mode. Device driver DD*00 interprets bit 9 as a programmatically initiated read, and if page strap is set, it will continue to read until it finds the block terminator.

The other bits are standard for I/O read requests (refer to in the Read/Write Parameter Definitions section of Chapter 1); however, the value of PRAM3 will override the end-of-transmission and echo characteristics set in CNTWD.

BUFR and BUFLN

BUFR is the name or address of the user buffer to receive data from the read request. BUFLN is the length of the user buffer to be used in the read request. If BUFLN is positive, the length is in words; if BUFLN is negative, the length is in characters.

If the input data ends on an odd (left) byte, the last word is padded with a blank (octal 40) in ASCII mode or a null (octal 0) in binary mode. This byte is *not* included in the transmission log computation. In all cases, the contents of the user buffer beyond the last valid word as indicated by the transmission log is undefined. There is no guarantee that the buffer has not been altered by any editing that may have occurred; neither should the user assume that the terminator byte (if any) is there.

PRAM3 Parameter

Bits 15 and 14 select the end-of-transfer character:

- 00 – no end-of-transfer character. Transfer will end when the buffer is full.
- 01 – transfer will end on carriage return (CR), control R (DC2), control D (EOT) or control ^ (RS). Note: When using the MUX driver, a control D flushes the buffer. When using the ASIC driver, a control D returns the buffer intact.
- 10 – transfer will end on control ^ (RS).
- 11 – transfer will end on carriage return.

Note When the device driver bypass bit (bit 15) is set and a read is requested (EXEC1), input data editing using the backspace and delete keys is recognized only if 01 has been chosen as the end-of-transfer option.

Bit 11 is the echo control character:

- 1 – echo on.
- 0 – echo off.

PRAM4 Optional Parameter

Bits 15 through 8 define the prompt character. If it is not null, then the prompt character will be output before the read is started.

Bits 7 through 0 define the number of characters to ignore at the end of a binary read. If nonzero, the MUX interface card is configured to read that many extra characters which will be discarded by the driver. This is used by the CTU driver DD*20 to prevent asynchronous interrupts when the user buffer size is smaller than the size of the record stored on tape.

IDM00 Write Request

The call sequence for the write request is

```
CALL EXEC ( 2, CNTWD, BUFR, BUFLN)
```

Note For all read/write (I/O) requests, the I/O transfers use the character format set up by function code 30B. The terminal must be strapped accordingly.

Control Word CNTWD

The write request control word CNTWD is shown in Figure J-19.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	Function Code						Device LU					

Figure J-19. IDM00 Write Request Control Word CNTWD

Bit 6 is for binary data transfer:

- 1 – Suppress CRLF and ignore bit 7
- 0 – Refer to bit 7 for CRLF control

Bit 7 controls CRLF insertion:

- 1 – Suppress addition of CRLF
- 0 – Append CRLF to buffer (if bit 6 is also 0)

Bit 8 is for the ENQ/ACK handshake:

- 1 – Disable handshake, this write only
- 0 – Allow handshake if enabled on card

Bit 11 is for NDT (Non-Display Terminator, ESC _) control:

- 1 – Output NDT after printing user buffer
- 0 – Do not output NDT

Bit 15 is the “bypass device” bit:

- 1 – Bypass the device driver
- 0 – Do not bypass the device driver

Note If CNTWD bits 11-6 are set to 35B, the user buffer will be sent to the card without adding anything to it.

BUFR and BUFLN

BUFR is the address of the user buffer to be used in the write request. BUFLN is the length of the user buffer to be used in the write request. If BUFLN is a positive number, the length is in words; if BUFLN is negative, the length is in characters.

IDM00 Status Request

Whenever an I/O request is made, certain status information is placed in DVT (Device Table) words 16 and 18. The formats of the status returned in DVT words 16 and 18 are as follows:

DVT16 bits 5 through 0 return the error code:

- 3 – timeout.
- 5 – transmission error.
- 11B – data communication line down.
- 77B – request aborted by the system.

DVT18 (read requests only):

- Bit 7 – 1 special character was NOT detected (binary read).
 – 0 special character was detected (ASCII read).
- Bit 0 – 1 data was received.
 – 0 data was NOT received.

IDM00 Control Requests

The call sequence for the control request is

```
CALL EXEC ( 3 , CNTWD , PRAM1 )
```

Control requests can also be performed interactively:

```
:CN,LU,CODE,PRAM1
```

Control Word CNTWD

Bits 11 through 6 contain the function code that defines the control function to be performed.

Bits 5 through 0 define the LU (logical unit number) to be controlled.

The function codes, right-justified in bits 11-6, are:

Code	Function
06B	Obtain dynamic status
23B	Control program scheduling
26B	Flush input buffer
30B	Set port ID
31B	Connect modem line
32B	Disconnect modem line
33B	Configure driver responses
34B	Set port configuration
36B	Set terminating count
37B	Set read type
50B	Initiate modem loop test
52B	Terminate receive buffer

Function Code 6B: Dynamic Status

This function code will return the requested port status in DVT16, DVT17, and DVT18. The character length of any “typed-ahead” data (refer to the section on Type-Ahead) will be placed in DVT word 17 (the transmission log). The formats of the status returned in DVT words 16, 17, and 18 are as follows:

DVT16 bits 5 through 0 return the error code:

- 3 – timeout.
- 5 – transmission error.
- 77B – request aborted by the system.

(Refer to “Extended Device Status” in Chapter 1 for an example of how to get DVT16, DVT17, and DVT18 by using RMPAR.)

If PRAM1 = 0, then DVT17 indicates the number of type-ahead characters. If PRAM1 does not equal zero, then the modem status is returned to DVT17.

Bits	Function
15-8	Undefined
7	0 – Connected to systems modem 1 – Hardwired (no systems modem panel)
6	1 – no response from system modem

Bits 15-8	Function Undefined
5	1 – Modem not present
4	1 – Being called
3	0 – Analog loopback 1 – Remote digital loopback
2	0 – Not in loopback mode 1 – Loopback completed
1	0 – Low speed 1 – High speed
0	0 – Line disconnected 1 – Line connected

DVT18 indicates card status:

Bits	Function
15	1 – Type-ahead data available on this port. 0 – No type-ahead data is available on this port.
14-0	Length of type-ahead data.

Function Code 23B: Control Program Scheduling

This function code will SET or RESET the Program-Schedule-Enabled flag in the interface driver according to the value of PRAM1 as follows:

PRAM1	– 0 enable scheduling. – 1 disable scheduling.
-------	---

The device driver will be entered to perform a schedule if the three following conditions are met:

1. Scheduling is enabled.
2. The interface driver is not expecting data from that port (a read operation is not in progress).
3. The port is not in type-ahead mode and any key is hit;

-OR-

The port is in type-ahead mode and the BREAK key is hit (refer to function code 33B regarding type-ahead and the BREAK key);

-OR-

The port is in type-ahead mode with scheduling on data available enabled and a valid record is received.

Function Code 26B: Flush Input Buffer

This function will cause the MUX interface to clear any data from the active port's input buffer which might have accumulated while in type-ahead mode. The value of PRAM1 will determine the action taken:

- | | |
|-------|--|
| PRAM1 | – 1 clear all input buffers on this port |
| | – 0 clear only the port's active input buffer. |

Function Code 30B: Set Port ID

This function will establish a logical connection between the LU to which it is directed and the physical device connected to the MUX interface. This function will also configure the port according to the value in PRAM1.

Note Function code 30B must be issued before any other request is given to that port. Any requests given to a port prior to function code 30B will be ignored by the interface driver. Also, the port number given should specify a unique device on the MUX interface. If a conflict exists, then the new assignment will take precedence and the old assignment will be lost.

PRAM1 is interpreted as explained below.

Bits 15-14 are the number of bits per character:

- 00 – 5 bits per character
- 01 – 7 bits per character
- 10 – 6 bits per character
- 11 – 8 bits per character

Bit 13 is to enable or disable the port as a modem LU:

- 1 – Enable this port as a modem LU. Caution: Setting this bit causes port 7 to be assigned to the HP 37214A Systems Modem Card Cage prohibiting port 7 from being used as a hardwired port. Conversely, if a modem is to be used with the HP 37214A, do not configure port 7 with a function code 30B. When using the 37222A Modem card, this bit must be set.
- 0 – Do not enable this port as a modem LU.

Bit 12 is the baud rate generator used for this port:

- 1 – Baud rate generator #1
- 0 – Baud rate generator #0

Note

Refer to the hardware/installation manual for setting bit 12. The way in which the customer cable has been wired will determine how to set this parameter. Also, refer to note 2, below.

Bits 11-10 indicate the number of stop bits:

- 00 – Reserved; do not use this bit pattern.
- 01 – 1 stop bit
- 10 – 1-1/2 stop bits
- 11 – 2 stop bits

Bits 9-8 select the type of parity:

- 00 – No parity
- 01 – Odd parity
- 10 – No parity
- 11 – Even parity

Bit 7 is for the ENQ/ACK handshake:

- 1 – Enable handshake
- 0 – Disable handshake

Bits 6-3 represent the baud rate:

- 0 – No change
- 1 – 50
- 2 – 75
- 3 – 110
- 4 – 134.5
- 5 – 150
- 6 – 300
- 7 – 1200
- 10B – 1800
- 11B – 2400
- 12B – 4800
- 13B – 9600
- 14B – 19200
- 15B – Reserved; do not use.
- 16B – Reserved; do not use.
- 17B – Reserved; do not use.

Note

1. 19200 baud is not supported on all eight ports simultaneously. A baud rate parameter of zero in bits 6-3 will result in no change to any of the device's parameters (baud rate, parity, stop bits, etc.)
2. The MUX card has two on-board baud rate generators providing baud rates to the eight ports. These eight ports can be divided into two groups and connected to either of the two generators. However, all ports on a given baud rate generator must be initialized to the same baud rate with the exception that (75 and 150) or (300 and 1200) or (2400, 4800, and 9600) can be simultaneously selected.

For example, in the case where ports 0-3 are connected to baud rate generator 0, ports 4-6 are connected to baud rate generator 1, and a modem card is plugged into port 5: If port 0 is set to 9600 baud, then ports 1-3 must be set to either 2400 or 4800 or 9600 since they are on the same baud rate generator. Assuming that the modem is initialized to auto-answer, then it must be set to 1200 baud. But if the modem port is at 1200 baud, then the other ports on baud rate generator 1 (ports 4 and 6) must also be set to either 300 or 1200.

When the MUX is connected to the systems modem in the HP 37214A, the firmware uses port 7 to communicate to the Systems Modem Controller at 1200 baud. Hence, port 7 should be grouped with the modem ports which will be run at 300 or 1200 baud.

3. If a port is a modem port, then the baud rate should be set to 300 or 1200 baud as required while originating a call. If auto answer mode is enabled, the baud rate should be initially set to 1200 baud. The firmware will automatically change it to 300 baud if the system modem detects that the remote modem is calling in at low speed (300). This is referred to as speed sensing.

Bits 2-0 are the port number of this device:

- 0 – Port 0
- 1 – Port 1
- 2 – Port 2
- 3 – Port 3
- 4 – Port 4
- 5 – Port 5
- 6 – Port 6
- 7 – Port 7

NOTE

The 37222A must always be assigned to port 0.

Function Code 31B: Connect Line

If the 37214A Systems Modem Panel or the 37222A Modem Card is being used, issuing function code 31B will establish a modem connection. This function code can also specify the name of a program to schedule when a modem line disconnect takes place. Program MODEM is available (part number 92077-16391) for use as a disconnect program and should suffice in most situations. MODEM can be configured to terminate active programs and reinitialize the modem port for auto-answer. Also, if the application requires it, MODEM can terminate and log off a session upon disconnect.

Once the connect line has been performed on a given port, the user need not do it again unless a power fail occurs or a disconnect request is executed. In the event that the modem gets disconnected accidentally, the user will be able to call back and establish the connection. However, in the case of an auto dialed call, the user will have to make a connect line request again to perform the auto dialing. When the line gets disconnected the action taken by the driver is selected according to the user specification of bits 15 and 14 in function code 33B (that is, down or do not down the device if the line gets disconnected). Connect line can be done interactively or programmatically by EXEC request.

For example:

```
:CN,LU,31B,PRAM1[,PRAM2[,PRAM3[,PRAM4]]]
```

where PRAM1 is as follows:

Bit 15 is the "No Wait" bit

- 0 – Driver will complete the request (that is, return) only after line has been connected.
- 1 – Driver will not wait for the line to be connected before completing the request.

Bits 14 through 6 apply only for program MODEM.

Bit 14 If the modem line goes down,

- 0 – Abort active program;
- 1 – Do not abort active programs.

Bit 13 Meaningless if bit 14 = 1; if bit 14 = 0, then if the modem goes down,

- 0 – Reenable the port for auto-answer;
- 1 – Do not reenable the port for auto-answer.

Bits 12 through 6 Log LU for messages from program MODEM.

Bits 5 to 3 are the configuration straps for the HP 37213A modem card. These straps do not affect the configuration of the external modem connected to the HP 37215A modem interface card.

- Bit 5 0 – guard tone off
 1 – guard tone on

- Bit 4 0 – 212 mode
 1 – V.22 mode

- Bit 3 0 – 10 bits
 1 – 9 bits

- Bit 2 0 – originate
 1 – answer

- Bit 1 0 – manual
 1 – auto-dial

- Bit 0 0 – low speed (300 baud)
 1 – high speed (1200 baud)

PRAM2, PRAM3 and PRAM4 contain the name of the alarm program. If an alarm program is specified, it must have an ID segment assigned (RP command) prior to the 31B request. If PRAM2 through PRAM4 are omitted, the previous (if any) program name will be used.

If the user wants to auto-dial, the number to be dialed is specified before the connect line request by making a write request with function code 35B as follows:

```
CALL EXEC (2, 3500B+LU, BUFR, BUFLN)
```

Where:

LU – LU of the modem

BUFR – buffer that contains the phone number (AUTO-DIAL BUFR)

BUFLN – length of BUFR

Function code 35B tells the driver that this is a special write buffer. The AUTO-DIAL BUFR should start with “T” for DTMF tone (push-button) or “P” for pulse (rotary) dialing, followed by a string of numbers. The format of BUFR is:

T+n+n+n+n+... n+n+n or P+n+n+n+n+... n+n+n

where + is a delimiter. Other valid delimiters are:

space (), - /

A delimiter need not be sent. Consistency of the choice of delimiters is not needed and any number of delimiters may be sent.

n is a digit 0 to 9, or *

An asterisk (*) will cause an access pause of two seconds during dialing. Up to two access pauses may be included. The total number of digits including pauses cannot exceed 125.

Examples:

1. P /031/331 – 1000

Pulse dial the number 031 331 1000

2. T 9 * 0101 (916) 786 2001

Tone dial 9 0101 916 786 2001 with an access pause after the first digit

The 37222A Modem Card has an auto-redial capability for occasions when the number called is either busy or unobtainable. Up to eight redials can be made, with a suitable pause between redials to allow the system or remote station to clear. To specify redial, use the form

RnPnumber

where:

n is an integer between 1 and 8 inclusive, specifying the number of times to redial.

number is the number to dial.

For example:

R5P408*257-7000

specifies pulse dial 408 257 7000 with an access pause after 408. If the number is busy or unobtainable, redial up to a maximum of five times. There are more examples given in the subsection titled “Modems” at the end of this section on driver IDM00.

Function Code 32B: Disconnect Line

Function code 32B is used to disconnect a line that is being used as a modem line. Once a disconnect Line request is done, an incoming call cannot establish a connection until a Connect Line request is performed, except unless the local modem is programmed to auto-answer a call.

The interactive format of disconnect line is:

:CN,LU,32B,PRAM1

where PRAM1 is as follows:

Bit 15: No Wait bit

- 0 – Driver will complete the request (that is, return) only after line has been disconnected.
- 0 – Driver will complete
- 1 – Driver will not wait for the line to be disconnected before completing the request.

Bit 14: Applies only when used with program MODEM.

- 0 – Abort active programs upon disconnect.
- 1 – Do not abort active programs upon disconnect.

Bit 0: 1 – Disable auto-answer.

Function Code 33B: Configure Driver Responses

This function will set up various parameters and responses strictly for the interface driver. None of these parameters are sent to the MUX interface itself. These parameters are defined by the value of PRAM1.

Note All of the two-bit fields defined below default to a value of 01 at system reboot.

Bits 15-14 are used to control responses to device or line failures, such as the modem line going down.

- 00 – Do change
- 01 – If the device or line goes down, or if a timeout occurs during a writer operation, set the device down (for example, IO NR)
- 10 – If the device or line goes down, do not down the device, but abort the request, then return EOT and go into a hard flush mode (that is, ignore all subsequent I/O requests). Function code 6B will return additional information regarding the failure.

Bits 13-12 define the type-ahead feature to be used:

- 00 – No change will be made in this field.
- 01 – No type-ahead. System attention will be gained by pressing any key when no read is pending.
- 10 – Enable type-ahead. If data is received without a pending read, it is saved on the MUX interface until the next read request. System attention will be gained by hitting BREAK key only, unless type-ahead scheduling is enabled.
- 11 – Reserved.

Bits 11-10 define the action to be taken by the interface driver when type-ahead data becomes available:

- 00 – No change will be made in this field.
- 01 – Bit 15 of DVT18 is set to 1 after a dynamic status request.
- 10 – Program scheduling is attempted and bit 15 of DVT18 is set to 1.
- 11 – Reserved.

Bits 9-8 define the action to be taken by interface driver when the **BREAK** key pressed:

- 00 – No change will be made in this field.
- 01 – Attempt program scheduling, if it is enabled.
- 10 – Clear all data on the port, then attempt program scheduling, if it is enabled; any data on the port will be lost.
- 11 – Reserved.

Bits 7-6 tell the interface driver how to control sending of read configuration information to the MUX interface card:

- 00 – No change will be made in this field.
- 01 – Always send configuration information on each EXEC read operation and on each function code 37B command.
- 10 – Get configuration information on function code 37B commands; that is, not from each EXEC read.
- 11 – Reserved.

Note Since Device Drivers DD*00 and DD*20 send configuration information to the I/O driver in EXEC read parameters, typical operations require that bits 7 and 6 be set to 10 for function codes 37B and 52B and set to 01 for all other function codes.

Bits 5-0 are reserved, and should be zero.

Function Code 34B: Set Port Configuration

Function code 34B uses the PRAM1 parameter to specify port configuration in addition to function code 30B. The bit fields in PRAM1 are defined as follows:

Bits 15-2 are reserved for future use and should be set to zero.

Bit 0 is for transmit pacing with XON/XOFF:

- 0 – Disable XON/XOFF handshake
- 1 – Enable XON/XOFF handshake

Bit 1 is to force an XON condition:

- 0 – Do not force an XON condition
- 1 – Force an XON condition

For example if the XON/XOFF handshake is enabled and the printer is powered off, issuing CN,LU,34B,3 will force an XON condition.

Transmit pacing is a mechanism by which the remote device can control (stop and resume) the transmission of data from the multiplexer. If enabled, transmit pacing is performed using XON and XOFF control codes. When the multiplexer receives an XOFF code (ASCII DC3), it stops transmitting data. When the multiplexer subsequently receives an XON code (ASCII DC1), it resumes transmitting data.

For example:

:CN,LU,34B,1	(Enable XON/XOFF)
:DL,CRN	(Causes a long output to your terminal)
CTRL-S	(XOFF will cause your output to stop; for example, if
.	you want to stop to look at something)
.	
CTRL-Q	(XON resumes the output to your terminal)

Caution If XON/XOFF transmit pacing is enabled, all CTRL-S characters received by the MUX are treated as handshake characters and therefore cannot be used as data characters. Similarly, each CTRL-Q preceded by a CTRL-S is treated as a handshake character. A CTRL-Q received without a CTRL-S is treated as a data character. Some of the HP Software Programs (EDIT, BASIC) use XON and/or XOFF characters to perform line and/or page editing. It is advised that you do not use these characters for editing and handshaking at the same time. A CTRL-Q can be used for editing as long as it is not preceded by a CTRL-S. Similarly, it is recommended that the XON/XOFF transmit handshake is disabled while reading binary data (for example from a data cartridge to a multiplexer port).

Note XON and XOFF codes can be issued through the keyboards to pace the data transmitted to your terminal. The CTRL and Q keys (when pressed simultaneously) generate an XON code and the CTRL and S keys generate XOFF.

Function Code 36B: Set Read Length

This function sets the read length in bytes for use in conjunction with the Terminate Receive Buffer (Function 52B).

:CN,LU,36B,254 (Sets the read length to 254 bytes)

Function Code 37B: Set Read Type

This function sends configuration information to the MUX interface card. The information is used by subsequent read requests. Normally, the configuration information is supplied by the interface driver as directed by the function field of EXEC 1 requests. Function code 37B lets you override the driver-defined values or configure a read operation on the MUX interface without executing a read request (as in type-ahead initialization).

Note If PRAM1 bit 7 in function code 33B is 0, then any subsequent read operation as directed by the function field of the EXEC1 request will override any of the values set by function code 37B; that is, if bit 7 of function code 33B is 0, then the read configuration will be determined by the order in which EXEC1 and CN 37B are executed in the program. If bit 7 in function code 33B is 1, then the read configuration will be sent only by the function code 37B command. Any read configurations specified by the function field of subsequent EXEC1 requests will be ignored.

The format for the values in PRAM1 are as follows. The bits, when set to 1, enable the following:

- 15 – End Transfer on Carriage Return (CR)
- 14 – End Transfer on CTRL ^ (RS)
- 13 – End Transfer on CTRL D (EOT)
- 12 – End Transfer on CTRL R (DC2) You must set the bypass device driver bit (bit 15 of CNTWD in EXEC1) if End Transfer is to function. It is strongly recommended that this option be used with extreme caution, since device driver DD*00 recognizes DC2 as an indication of block transfer.
- 11 – End Transfer on Count (refer to function code 36B)
- 10 – End Transfer according to bits 15-12

- 9 – Enable Input Data Editing
- 8 – Enable Input Data Echo 7-0 – reserved

Note Bits 10 and 11 are mutually exclusive. Never set both bits together.

Function Code 50B: Loopback Test

Function code 50B is used to perform a local analog or a remote digital loop test when using the HP 37222A Modem Card or the HP 37213A Modem Card in the HP 37214A Systems Modem Panel. Refer to the manuals for the external modems for instructions on doing loopback tests on them. Function code 50B only applies to the HP 37213A Modem Card and the HP 37222A Modem Card.

ENQ/ACK handshake must be set off to perform the loopback test. After sending the Function 50B request, wait about three seconds and read the port status as described in the Dynamic Status section of this chapter to confirm the loopback.

After the confirmation, put the port in a type-ahead mode and perform the integrity check of the modem line by comparing the received data against the transmitted data. After the test, disable the loopback test by sending another Function 50B request with PRAM1 = 0.

Note that if a port is down, the driver IDM00 will pass only the write request with function code 35B to the card. Therefore, if the loopback port is down, you should use the write request (function code 35B) to send the loopback data to the card.

An example to initiate the loopback test is:

```
:CN, LU, 50B, PRAM1
```

where PRAM1 is

- Bit 2 0 – low speed (300 baud)
1 – high speed (1200 baud)
- Bit 1 0 – analog
1 – remote digital (only valid at high speed)
- Bit 0 0 – disable loopback test
1 – enable loopback test

The following is an example of a loopback test:

```
*** ENABLE TYPE-AHEAD & DO NOT CHANGE READ CONFIGURATION  
*** EVERY TIME MODE  
  
:CN, LU, 33B, 20200B  
  
*** SPECIFY ECHO/EDIT OFF, END TRANSFER ON CARRIAGE RETURN
```

```

:CN,LU,37B,100000B

:** INITIATE THE LOCAL ANALOG LOOPBACK MODE AT HIGH SPEED

:CN,LU,50B,5
.
.
:** WRITE DATA TO THE LU; USE 35B AS THE WRITE FUNCTION
:** CODE; BYPASS THE DEVICE DRIVER (BIT 15)

CALL EXEC(2, 103500B+LU, TBUFF, BUFLLEN)

:** READ BACK THE DATA; USE 35B AS FUNCTION CODE; BYPASS
:** THE DEVICE DRIVER.

CALL EXEC(1, 103500B+LU, RBUFF, BUFLLEN)
.
.
NOW COMPARE TBUFF WITH RBUFF (THEY SHOULD BE IDENTICAL)
.
:** DISABLE THE MODEM LOOPBACK TEST

:CN,LU,50B,0

:** CONFIGURE THE PORT BACK TO ITS NORMAL MODE OF
:** OPERATION

:CN, LU, 33B, 10100B

```

Function Code 52B: Terminate Receive Buffer

Function code 52B instructs the interface card to immediately terminate its active receive buffer. A subsequent read request will read the terminated buffer with the length of the received data returned in the B-Register (that is, transmission log).

If the active receive buffer on the interface card is empty when function code 52B is issued, the buffer will be terminated and the driver will be informed as soon as the first character is received. The subsequent read request issued will be completed with a transmission log of one byte.

This control request is useful to read incoming data which is not in the format expected by the multiplexer, e.g., ending on <CR>, <DC2>, <RS>, CONTROL-D, or count. In other words, if a user does not know how many characters to expect, and if the data does not have a record terminator that is recognized by the MUX, function code 52B can be used to read the data.

The following example shows how to read data from a device connected to the multiplexer. The length of the incoming data is not known ahead of time and/or the record terminator is not recognized by the multiplexer. Note that the port is configured to be in type-ahead mode and the character count is set to 254.

```

      :
      :
C
C   PROGRAM THE MUX PORT IN TYPE-AHEAD MODE & SPECIFY NOT TO
C   RECONFIGURE THE READ OPERATION ON A READ REQUEST.
C
      CALL EXEC (3, 3300B+LU, 22200B)
C
C   SET THE READ CONFIGURATION TO END ON A COUNT OF 254, ECHO
C   AND EDIT OFF
      CALL EXEC(3,3600B+LU,254)
      CALL EXEC(3,3700B+LU,4000B)
C
C   THE FOLLOWING LOOP WILL TERMINATE & READ THE INCOMING
C   DATA BUFFER
C   NOTE: IF THERE IS NO DATA, THE PROGRAM WILL HANG UNTIL THE
C   DEVICE TIMES OUT. THIS TIMEOUT WILL CAUSE THE PROGRAM TO
C   ABORT UNLESS THE NO (NO ABORT) BIT IS SET IN THE EXEC ALL.
C
10   CALL EXEC(3, 5200B+LU, 0)
      CALL EXEC(1, LU, BUFF, -254)
      CALL ABReg(ISTAT, LEN)
C
C   LEN HAS THE LENGTH OF THE DATA RECEIVED
C
      :
      :
C
C   PROCESS THE RECEIVED DATA
C
      :
      :
C
C   READ SOME MORE DATA
C
      GO TO 10

```

Instead of doing a read after the terminate request, a user may choose to schedule a program when type-ahead data is available. This can be done by setting bits 11-10 to "10" in control request 33B. Refer to function codes 20B and 40B in the section on Terminal Device Driver DD*00 for instructions on how to establish the program to be scheduled. After the terminate request is issued, the program will be scheduled as soon as the type-ahead data is sent to the driver by the interface card. The scheduled program will then read the data, and if desired, issue another terminate request to read more data.

IDM00 Type-Ahead

Type-ahead is the ability of a system to accept data from a terminal or device before it is needed. The buffered MUX card can accept up to two lines of text per port of type-ahead. The advantages of type-ahead over the current RTE convention are prevention of data loss and unexpected system prompts when keys are accidentally struck.

While in type-ahead mode, the driver leaves a read request pending on the card (not the IFT) at all times. This read allows the user to enter data into the MUX card even though the system does not have a read pending. When the card receives type-ahead data, it interrupts the system to signal that data is available. If the data is not needed yet, a flag is set in the card status and the driver exits and waits. When the system is ready for the data, the driver gets it from the MUX and returns it.

The BREAK key is not buffered, so it must be used to get the system's attention when it is busy. If type-ahead scheduling is enabled, the user can enter a system command or any other valid keyboard entry (followed by a carriage return) without first having to get the system's attention. The system will execute the command as soon as it is read.

There are two type-ahead modes. In full type-ahead, as described above, successive read requests fetch successive lines of text from the MUX card. This mode is good for text editing, for example.

In situations where the system response could alter a user's next command (FMGR error messages, for example) full type-ahead can cause problems, as in this example:

```
User types: ST,FILE,8
While tape is moving, user types: PU,FILE
Tape runs out; system downs the device.
User hits BREAK, system issues prompt and read.
Driver reads PU command from card buffer and system tries to execute it.
```

In the above example, the user gets back an "ILLEGAL COMMAND" from the system the first time the request for system attention is made. However, the commands stored on the card may have a disastrous affect on the system.

A solution to the above problem is to program the driver to cancel all card data upon receiving a BREAK interrupt. This preserves the multiline type-ahead feature, and reduces the chance of data being read by the wrong process.

Another solution to the above problem is to issue a Flush Input Buffer (code 26B) request with any "fatal" error messages to the user to clear the extra commands before they can be read. For a description of the driver configuration options, refer to function code 33B.

An additional advantage is that applications programs can make the system appear more responsive to the user, increasing total throughput. This is done by having the application program prompt the user for his next response before processing the previous one. By the time the user has finished typing, the system will have caught up and can begin processing again. As long as the processing takes less time than the typing, the user perceives instant response time.

Note that the above forms of type-ahead are also useful in non-terminal device communication. The buffering on the card may eliminate the need for stacking two class read requests on an LU prevent data loss, thus reducing program size and complexity, and the amount of large System Available Memory (SAM) required.

When data is available on the multiplexer card, and there is no pending request to accept it, a bit will be set in the status word and program scheduling attempted. Should the user program decide it does not want the data, it can issue a Flush Input Buffer (function code 26B) to remove the data.

In the non-type-ahead mode of operation, the subsystem will operationally appear the same as other current RTE terminal drivers. The driver, when a port is inactive, will leave an "interrupt on any character" read pending on the card so as to be informed when a key is struck. The appropriate action (system attention, program schedule, etc.) will then be taken.

Since the MUX performs all data processing on input or output data, it needs some configuration information to know just what operation to perform. For output, this consists of whether to add a CRLF (Carriage Return/Line Feed) and whether the ENQ/ACK or XON/XOFF handshake is enabled. Input (Read) configuration information is what has been referred to as a "record". This consists of the following:

1. Echo ON/OFF
2. Editing ON/OFF (This includes Backspace and Delete)
3. Terminating Condition:
 - a. Count: Terminate after a defined number of characters
 - b. Character: Terminate on one or more of the following characters:
 - 1) CR (Carriage Return)
 - 2) RS (CTRL ^)
 - 3) DC2 (CTRL R)
 - 4) EOT (CTRL D)

Each of the two input buffers for each port on the MUX can hold one record, or part of a record, if the record is longer than 254 characters. This limitation holds true even if the records are only a few characters; records are not packed into the buffers. This places the two-record limitation on type-ahead. If a third record is received before space is made available on that port of the MUX card, then an overflow condition occurs. This results in the first record being left exactly as it was and the second record reporting an overflow condition (record length = 0, status = 5).

Buffer Overflow and Transmission Error

When both MUX input buffers are full and a third record is received, an overflow condition occurs. This will cause the first buffer to remain intact, the second buffer to be corrupted, and a transmission error to be reported by the operating system. This might cause problems under certain conditions (that is, in screen edit mode, where transmission error messages erase the records and the message is embedded within the editing file).

This problem can be eliminated by setting bit 4 of the terminal configuration word (DVP1) either at generation time or using control function 44B online (that is, CN,LU,44B,20024B). When bit 4 of driver parameter 1 (DVP1) is set, the transmission error will be suppressed in a buffer overflow condition. The user application program can check for the occurrence of this condition by examining bit 3 of the status word (A-Register). The driver will always set bit 3 of DVT6 (A-Register) in case of a buffer overflow. Note that if a buffer overflow occurs, the second input buffer is always corrupted.

Type-Ahead and Terminal Status Request

Application programs are required to report terminal status to the device driver. This will enable the driver to recognize the mode of data transfer (that is, character, block line or block page) and to take appropriate action.

If the terminal is on a MUX port that is configured for type-ahead, and terminal status is requested (CN,LU,25B), the type-ahead data will be reported to the driver as the terminal status. This bogus status causes the device driver to enter a state of confusion that will hang up that port.

To eliminate this problem, set bit 5 of the terminal configuration word (DVP1). This can be done at generation time or on-line by issuing CN,LU,44B,20044B. This will cause the MUX driver (IDM00) to issue a “flush both input buffers” command (CN,LU,26B,1) to the MUX card and to clear any type-ahead data before executing the CN,LU,25B command (terminal status request).

Virtual Control Panel (VCP)

The HP 12040B/C Multiplexer supports VCP. The VCP terminal must be plugged into port 0. The interface card VCP switch (#1) must be closed. If you use the MUX terminal as the system console, be sure to generate it as LU 1.

The baud rate of the VCP terminal (on port 0) must be set as shown in the following table at bootup. Port 0 and VCP terminal baud rates can be changed programmatically after bootup.

Table J-3. VCP Terminal Baud Rates

Firmware Part Number	Product Number	Cable Part Number	Baud Rate at Bootup
5180-1970	12040B	12828-60002	2400
5180-1970	12040B	12040-60002	9600
5180-7227	12040B	12828-60002	2400
5180-7227	12040B	12040-60002	9600
5180-7228	12040C	12828-60002	9600

Modems

If the user wishes to have full modem capability, the HP 37214A Systems Modem Panel should be used in place of the HP 12828A Multiplexer Panel. The system modem is an eight-slot card cage which includes a modem controller card, a modem card (HP 37213A), an external modem card (HP 37215A), and a terminal interface card (HP 37216A). The modem card (HP 37213A) is compatible with Bell 212 and 103 modems. If your configuration will be communicating via telephone lines, the modem controller card (supplied with the HP 37214A) must always be plugged into port 7, and one modem card (HP 37213A) for each telephone line plugged into any of the other slots. Thus, it is possible to have up to seven modem lines (since one slot is taken by the modem controller card). If the user has his own modems, they can be connected through the external modem card (HP 37215A). If he wishes to connect terminals to the MUX, the terminal interface card (HP 37216A) is used.

Any combination of modem cards, external modem cards, or terminal interface card (up to seven) can be used. If only terminal interface cards are used, port 7 can also be used for a terminal so that it is possible to have all eight ports used for terminals. If one or more modem cards (including the external modem card) is used, then the modem controller card must be inserted into port 7. The control of modems through the system modem panel is described in this manual.

Modem capability can also be achieved with the HP 37222A Modem Card, which is used in place of the 12040B/C MUX Interface Card. The 37222A is a single modem card and is configured as MUX port zero. It is used in the same manner as the 37213A Modem Card, which plugs into the 37214A Systems Modem Panel.

The following is a typical example of the commands in a WELCOME file. This sequence applies for either the HP 37214A Systems Modem Panel or the HP 37222A Modem Card.

```
*
RP MODEM.RUN: :SYSTEM
CN 72 30B 172271B          37213A in port 1; 1200 baud; ENQ/ACK
CN 72 33B 100000B          enabled. Do not down device if modem line
CN 72 31B 105B MO DE M    goes down. Schedule MODEM upon line dis-
*                          connects; abort active programs if modem
*                          line goes down; reenable port for auto-
*                          answer if line goes down; log messages
*                          from MODEM to LU 1; enable port for high-
*                          speed (1200 baud) and auto-answer;
*
CN 72 20B PROMT           Enable LU 72.
*
```

As mentioned earlier, the 37222A Modem Card must always be specified as port 0; therefore, the corresponding control 30B command would be CN 72 30B 162270B.

Also mentioned earlier, in the description of control 30B, the firmware uses port 7 to communicate with the modem controller card (port 7) at 1200 baud when the MUX is connected to the 37214A Systems Modem Panel. Hence, port 7 should be grouped with the modem ports that will be run at 300 or 1200 baud.

For example, assume you are using the standard MUX cable (that is, baud rate generator 0 connected to port 0 and baud rate generator 1 connected to ports 1 through 7). You also have a modem card (for example 37213A) in port 1. When port 1 is initialized with control 30B specifying that it is a modem port (that is, bit 13 set), the modem controller card is activated and ports 1 through 6 can only be used at 300 or 1200 baud. Terminals can still be used in ports 2 through 6 but must operate at 300 or 1200 baud. However, port 0 can run at any speed desired since it is on a different baud rate generator.

Note that if you are using the standard MUX cable and wish to use the 37214A Modem card in port 0, one of the other ports (1 through 6) must still be initialized as a modem port in order to activate the modem controller card in port 7. Therefore, it is recommended that the MUX cable be modified so that the modem ports are connected to the same baud rate generator (no matter which one) and that they include port 7 for the modem controller card. Then connect the non-modem ports to the other baud rate generator so they are not restricted to operating at 300 or 1200 baud.

Program "MODEM" consists of the following:

1. Main program %MODEM (92077-16391)
2. Library %MDMLB (92077-16392)
3. LINK with LI,%MDMLB
 RE,%MODEM
 EN

A typical output message is as follows:

```
MDM: LINE disconnECT DETECTED 5:45 PM TUE., 20 SEPT, 1983
MDM: 12040B ST= 3410 LU 71
```

Refer to the *HP 12040B/C Multiplexer Interface Card Reference Manual* for the meaning of the status (ST) bits.

The following is a sample program compatible with a Racal-Vadic modem model VA3451. The user calls in to the system from a remote site via a VA3451 and logs on. The user then runs the DIAL program to have the system call back so that phone charges are absorbed by the system's phone.


```

FTN7X,L
C
C   PROGRAM "DIAL" IS USED TO CALL BACK THE USER WHO IS LOGGED
C   ON TO A SYSTEM FROM A REMOTE SITE VIA MODEM.  THE SYSTEM
C   MUST HAVE EITHER A 37214A SYSTEMS MODEM PANEL WITH A
C   37213A MODEM CARD OR A 37222A MODEM CARD (BOTH OF WHICH
C   HAVE AUTODIAL CAPABILITIES).  THIS PROGRAM IS AN EXAMPLE
C   FOR USE WITH A RACAL-VADIC MODEM, MODEL VA3451.
C
C   OPERATING INSTRUCTIONS:
C
C       :RU,DIAL,P1[,P2]  NOTE:  P2 IS OPTIONAL
C
C           P1=YOUR TELEPHONE NUMBER
C           P2=DELAY TIME IN SECONDS THAT YOU WANT
C               THE SYSTEM TO WAIT BEFORE CALLING YOU
C               BACK; (GIVES YOU TIME TO SET UP YOUR
C               MODEM); DEFAULTS TO 10 SECS.
C
C   eg.
C
C       ;RU,DIAL,415-997-2702      DIAL will disconnect you and call you
C                                   back at area code 415, phone# 997-2702.
C
C   program dial(3,2000),autodial program TMH <851101.1009>
C   implicit integer (a-z)
C   DIMENSION NAME(3)
C   integer rmparms(5),phonenumberbuff(32),xluexparms(2),IDATE(15)
C   character*64 phonenumber,temp
C   equivalence (phonenumber,phonenumberbuff)
C   DATA NAME/'DIAL' /
C
C-----Print help message if needed
C
C   call rmpar(rmparms)
C   if(rmparms(1).eq.1) then
C
C       write(1,*)      'Usage:  [ru]      dial,PhoneNumber,WaitTimeInSecs'
C       write(1,*)      'Examples:      dial,415-283-7294,10'
C       write(1,*)      'Examples:      dial,736-9732,10'
C       WRITE(1,*)      'Examples:      DIAL,9*257-2702'
C       WRITE(1,*)      '                ^ The * means pause 4 secs between
C   +n dialing the 9 and 2'
C       stop
C       endif
C
C-----PROCESS THE INPUT PARMS
C
C   length = rcpa(1,temp)          ! Process the phone#
C   phonenumber = 'T*' // temp     ! & insert T* in front of it. c
C
C-----
C
C   INSERT YOUR OWN ROUTINE HERE TO VERIFY VALID PHONE NUMBERS.
C
C-----

```

```

CC
C
      IF(RMPARMS(2).EQ.0) RMPARMS(2)=10      !Default wait time to 10 sec.
      IDADDR=IDGET(NAME0)                    !Get the
      I=IXGET(IDADDR+28)                    !LU number of the
      RMPARMS(3)=IANI(I,377B)               !scheduling terminal.
D      WRITE(6,5) RMPARMS,PHONENUMBERBUFF
D      5 FORMAT(527," ",32A2)
C
10     ITIME=RMPARMS(2)+15
      WRITE(1,*)' '
      WRITE(1,*)' '
      WRITE(1,*)' +-----+'
      WRITE(1,*)' + Wait for CXR light to go out then +'
      WRITE(1,*)' + move the DA/VO/MA switch to VO. +'
      WRITE(1,500) ITIME
500    FORMAT(' + Your phone will ring in about 'I3' +')
      WRITE(1,*)' + seconds. Then in another 20 secs, +'
      WRITE(1,*)' + the CXR light will come on and a +'
      WRITE(1,*)' + WELCOME BACK message will appear. +'
      WRITE(1,*)' + If you are not back ONLINE in +'
      WRITE(1,*)' + approx 70 sec, call in again. +'
      WRITE(1,*)' +-----+'
      WRITE(1,*)' '
      CALL FTIME(IDATE)
      WRITE(1,15) IDATE
15     FORMAT(' '15A2)
      WRITE(1,*)' '
      WRITE(1,*)' * * * * * B Y E * * * * *'
      WRITE(1,*)' * * * * * * * * * * * * * * *'
      WRITE(1,*)' '
      WRITE(1,*)' '
      WRITE(1,*)' '
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CC
C
C
C      INSERT YOUR OWN ROUTINE HERE TO LOG THIS CALL TO A LOG FILE.
C
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CC
C
      CALL EXEC(12, 0, 2, 0, -2)      ! Wait for 2 sec. for messages to
complete

c disconnect & wait for WaitTimeInSecs number of seconds before calling back
      IF(RMPARMS(3).EQ.0) RMPARMS(3) = 1
      XLUEXPARMS(1) = RMPARMS(3)
      XLUEXPARMS(3, XLUEXPARMS, 40000B)      ! ISSUE disconnect RQST.
      CALL XLUEX(3, XLUEXPARMS, 40000B)
D      WRITE(6,*) 'ISSUED disconnect -- SLIGHT PAUSE'
      call exec(12, 0, 2, 0, -rmparms(2))

```

```

c Send the phone number to the card
  XLUEXPARMS(2) = 3100b
  call xluem(2,xluexparms,phonenumberbuff,-trimlen(phonenumber))

c Tell the card to dial (high speed is the default)
  xluexparms(2) = 3100b
  call xluex(2,xluexparms,100103b)

C
C --- WAIT FOR LINE CONNECT
C
  ITRYS=1
  XLUEXPARMS(2) = 100600B
700 CALL XLUEX(3,XLUEXPARMS,1)           ! GET DYNAMIC STATUS.
  CALL ABReg(IA,IB)                       ! GET A & B Reg.
D WRITE(6,701) RETRY,ITRY,IA,IB          !
D 701 FORMAT(2I4," IA="@7" ; IB="@7)      !
  IF(IAND(IB,1).EQ.1) GO TO 900           ! CONNECTION COMPLETED
                                           ! YET?
  CALL EXEC(12,0,2,0,-4)                  ! NO, PAUSE ANOTHER 4 SECS.
  ITRYS=ITRYS+1                           !
  IF(ITRYS.LT.10) GO TO 700              ! GET THE STATUS AGAIN.

C
D 777 CALL EXEC(2,6,31HCOULD NOT ESTABLISH CONNECTION,-31)
D WRITE(6,7771) XLUEXPARMS(1)
D 7771FORMAT("ODIAL CALLING CLGOF; XLUEXPARMS(1)="I3)
  CALL CLGOF(XLUEXPARMS(1),1,ERROR)       ! LOG OFF THE SESSION.
  XLUEXPARMS(2)=3100B
  CALL XLUEX(3,XLUEXPARMS,105B)          ! RE-INIT PORT FOR AUTO ANS
  STOP

C
900 WRITE(1,*)' '
  WRITE(1,*)' '
  WRITE(1,*)'
  WRITE(1,*)' * * * * * * * * * * * * * * * '
  WRITE(1,*)' * * *   W E L C O M E   * * * '
  WRITE(1,*)' * * *       B A C K       * * * '
  WRITE(1,*)' * * * * * * * * * * * * * * * '
  WRITE(1,*)' '
  WRITE(1,*)' '
  CALL FTIME(IDATE)
  WRITE(1,15) IDATE
  WRITE(1,*)' '
  END

```

ID*00/ID*01 ASIC Interface Drivers

ID*00 and ID*01 are the interface drivers for the 12005A Asynchronous Serial Interface Card (ASIC) operating under RS-232 protocol. Where these drivers are the same they will be referred to as a combined driver called ID*00/ID*01.

Driver ID*00/01 performs the actual I/O instructions to the ASIC and is therefore a card-dependent and device-independent software module. The driver performs the basic I/O read, write, and control operations via EXEC requests, with all I/O transfers under interface card DMA control.

Driver ID*01 is to be used when 103/202 compatible asynchronous modems are part of the link between the ASIC card and a terminal that is not used as a VCP (Virtual Control Panel). It is an enhanced version of ID*00; thus the information concerning ID*00 and ID*01 have been combined. ID*01 modem capabilities are covered later in this chapter.

When the modem capabilities of ID*01 are invoked, the driver will automatically manage the states of 103/202 compatible asynchronous modems. If not invoked, or if the modem capability is turned off, ID*01 will imitate ID*00.

The following sections cover the programming considerations that an applications programmer should know when making EXEC I/O requests to the ASIC. Refer also to the *HP 12005A Asynchronous Serial Interface Reference Manual*, part number 12025-90001, for hardware-dependent information.

ID*00/ID*01 Read/Write Request

The call sequences for the read and write requests are

READ Request: CALL EXEC (1, CNTWD, BUFR, BUFLN, PRAM3 [,PRAM4])
WRITE Request: CALL EXEC (2, CNTWD, BUFR, BUFLN, PRAM3)

If ID*01 has its modem management capability turned on, the ASIC card control-word bits 3-7 will be forced to zero.

Control Word CNTWD

Figure J-20 gives the format of the EXEC control word (CNTWD) used by ID*00/ID*01 for a read or write request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	X	SP	X	X	X	CRLF	BI	LU					

Figure J-20. ID*00/ID*01 Read/Write Request CNTWD Format

The BB (driver bypass), NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit assignment) bits are as defined in the Read/Write Request Parameter Definitions section of Chapter 1. All unused bits (marked x) should be set to zero.

SP Start of Page. For devices that use the page mode, the ASIC driver must be notified if the device is to operate in the block or character mode for each request:

- 1 – After the user buffer is sent, the driver issues an escape sequence (ESC_DC1) that marks current position on the display as the start of new logical page. The next read operation will only transmit characters entered after that position.
- 0 – The driver need not append the start-of-page sequence to user output buffer. The SP bit should be set to zero for terminals that have no page mode.

CRLF Carriage Return/Line-Feed bit (Write requests only):

- 1 – Disable the CRLF sequence normally transmitted to the terminal when the CR key is struck.
- 0 – Allow driver to transmit a CRLF to the terminal when the CR key is struck. For an ASCII read request to a user buffer, CRLF is transmitted to the terminal when terminal CR key is pressed (CR also is read to the buffer). For a binary read, the CRLF is transmitted to the terminal. For an ASCII write request from a buffer to a display, the driver transmits a CRLF to the display after the user buffer has been written to the display. For binary write requests, CRLF is not transmitted.

BI Binary bit:

- 1 – Enable binary operation.
- 0 – Enable ASCII operation.

When binary operation is enabled, the driver interprets all read and write request data as binary information. The driver transmits the data as bytes of information.

When ASCII operation is enabled, the driver interprets all data for read or write requests as ASCII. The first seven bits are interpreted as the ASCII code. The eighth bit can be interpreted as a parity bit.

BUFR and BUFLN

The I/O buffer used in the EXEC request is described by parameters BUFR and BUFLN, and are as defined in the Read/Write Request Parameter Definitions section of Chapter 1: BUFR is the name (or address) of the buffer, and BUFLN is its length.

PRAM3 Control Word Modifier

PRAM3 is the ASIC control word modifier. For each I/O request to ID*00/ID*01, a card control word must be supplied to the interface driver, as described in the next section.

When PRAM3 is set to zero, the card control word will be set to a default value. Figure J-21 shows the format of the card control word. The control word bits are described in the next section.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	Z	X	TR	KP	EC	X	BI	XX	XX	X	X	X	X

Figure J-21. ID*00/ID*01 Card Control Word Default Format

- PEE – Parity Error Enable.
- FEE – Framing/Overrun Error Enable.
- RCV – Receive Mode.
- XMIT – Transmit Mode.
- CHLN – Character Length Select.

If you pass a zero card control word (DVT18=0), to the driver, the RCV, XMIT, and CHLN bits in the control word are set to the following default values:

- RCV Read request.
- XMIT Write request.
- CHLN The setting of CHLN is determined by bit 6 of the EXEC control word CONTWD. For ASCII requests, CHLN = 0. When a character is received, the hardware forces the high order bit to 0. For binary requests, CHLN = 1. When a character is received, the hardware does not change the high-order bit.
- PEE,FEE 3 For ASCII read or write requests, the settings of the PEE and FEE bits are determined by the error checking mode (see Control Request, Function Code 43B). PEE and FEE are XOR'd into the control word. PEE enables parity bit generation and error checking. FEE enables framing error detection.

If you pass a non-zero card control word to the driver, all bits (except RCV, XMIT, CHLN) set in PRAM3 will override the corresponding bits in the default card control word. This is accomplished by masking out the RCV, XMIT, and CHLN bits and setting them as described above. The PEE and FEE bits are exclusively OR'd as described above.

PRAM4 Optional Parameter, Read Requests Only

If the upper byte of PRAM4 is any value greater than 0, the ASCII character in the upper byte of PRAM4 is output to the terminal before a read request is executed. If the upper byte of PRAM4 is less than or equal to zero, the request is handled as a normal read and the lower byte of PRAM4 is tested.

Either a positive or negative number in the lower byte will control the number of interrupts to be ignored by ID*00/ID*01. This number (n) in the lower byte is used to build a dummy quad to ignore (n-1) interrupts following the actual read. This request is used for the CTU driver (DD*20) during a read when the user buffer length is less than the request length on the tape. The remaining number of interrupts (n) coming in from the tape are stored in the lower byte of DVT19 so as not to cause an asynchronous interrupt. If the lower byte of PRAM4 is equal to zero, the request is handled as a normal read request.

ID*00/ID*01 Card Control Word

There are five output card control words that may be sent to the Asynchronous Serial Interface Card (ASIC). Four words are used for DMA configuration and are set up by ID*00/ID*01. The fifth is used in any mode (DMA or non-DMA) and is the only card control word necessary when DMA is not used.

Figure J-22 shows the format of the card control word that must be provided by the interface driver for all I/O requests. You have the option of setting any bit in the card control word, except for bits 8 through 10 (CHL, XMT, RCV), via PRAM3 for an EXEC read or write request to the ASIC. If PRAM3 is set to zero, the default card control word will be used by the driver.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SC1	SC2	PEE	FEE	EC	RCV	XMT	CHL	MIE	RRR	ICR	CSR	DMR	SSD	RS	TR

Figure J-22. ASIC Control Word Format

SC1

SC2 Special Character Select 1 and 2. ASIC has the hardware capability of monitoring the incoming data stream looking for any one of a set of special characters. The special character can be any 7- or 8-bit pattern designated by the CHL bit. The character select 1 and 2 (SC1/SC2) are

SC1	SC2	Driver Action
0	0	Disable all special character recognition. (Default condition)
0	1	Enable Type 1 special character recognition: CR – Carriage Return BS – Backspace DEL – Rubout ^D – Control D (EOT) DC2 – Handshake code signals block transfer.
1	0	Enable Type 2 special character recognition: RS – Record Separator
1	1	Enable Type 3 special character recognition: CR – Carriage Return

Note

If PEE, FEE are left enabled while the receiver-bit (RCV) is disabled, an interrupt condition could still occur (if erroneous data were received). For example during a transmission, if a key is struck at the terminal, and that character had an error, an interrupt irrelevant to the data transmission would occur.

It is advisable to request a card reset (function code 00) before setting PEE or FEE. This will clear any residual error flags set by previous data transfers.

- PEE** Parity Error Interrupt Enable. The default case is defined by the error checking control function (function code 43B) discussed later in this section. If a error checking control request is not made, the default is PEE = 0.
- 1 – Enable parity bit generation, and set a flag if a parity error is detected. The parity bit is transmitted after seven data bits in the ASCII format, or eight data bits in the binary format.
 - 0 – Inhibit parity generation and parity checking.

- FEE** Framing/Overrun error interrupt enable. A framing error is the absence of the correct stop bits in data transmission. An Overrun error occurs when characters are being received from the terminal faster than the CPU or memory can receive them.

The default setting is defined by the error checking control function (function code 43B) discussed in the next section. If an error checking control request is not made, the default is FEE = 0.

- 1 – Enable Framing/Overrun Error interrupts.
- 0 – Disable Framing/Overrun Error interrupts.

- RCV** Receive mode. The default is determined by the request type passed in the first EXEC request parameter (ECODE). If the I/O request is a read request, RCV will be set to 1. If the request is a write request, RCV will be set to 0.

- 1 – Enable ASIC receiver and associated logic.
- 0 – Disable ASIC receiver.

When RCV = 1, the receiver accepts data when available, or, for direct memory access (DMA), it initiates a write into memory when data is received.

- XMT** Transmit mode. The default setting is determined by the request type passed in the first parameter of the EXEC request (ECODE). If the I/O request is a write, XMIT will be set to 1. If the request is a read, XMIT will be set to 0.

- 1 – Enable ASIC transmitter and associated logic.
- 0 – Disable ASIC transmitter.

The effect of this bit is to enable the transmitter (when it is ready for a transmission) to send data in the case of programmed I/O. When operating under DMA, the transmitter will initiate a memory ready cycle (when ready for a transmission).

- CHL** Character length select. The default setting is the same as the setting of the BI bit (bit 6) in EXEC request control word (CNTWD) (BI=1=CNLN; BI=0=CNLN):

- 1 – Interpret all 8 bits of data byte as the character length, as with binary code.
- 0 – Interpret 7 LSB of the data byte as the character length, as with ASCII code.

- MIE Modem interrupt enable:
- 1 – Enable interrupt if any incoming modem control states changes from reference defined by RRR, ICR, CSR, and DMR bits.
 - 0 – Disable all modem interrupts (default). The next four bits set up the reference status to which the corresponding device status input lines will be compared. If any of the device lines differ from the reference bits, and if MIE (bit 7) is set, an interrupt will occur.
- RRR Receiver ready reference:
- 1 – Receiver ready OFF.
 - 0 – Receiver ready ON (default).
- ICR Incoming call reference:
- 1 – Incoming call OFF.
 - 0 – Incoming call ON (default).
- CSR Clear to send reference:
- 1 – Clear to send OFF.
 - 0 – Clear to send ON (default).
- DMR Data mode reference:
- 1 – Data mode OFF.
 - 0 – Data mode ON (default).
- SSD Secondary send data:
- 1 – Mark
 - 0 – Space (default).
- RS Request to send:
- 1 – Request to send OFF.
 - 0 – Request to send ON (default).
- TR Terminal ready:
- 1 – Terminal ready OFF.
 - 0 – Terminal ready ON (default).

ID*00/ID*01 Control Request

The call sequence for the control request is:

```
CALL EXEC (3, CNTWD [,PRAM1])
```

Figure J-23 illustrates the format of the control word (CNTWD) used by ID*00/ID*01 for an ASIC control request.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BB	NB	UE	0	Function Code						LU					

Figure J-23. ID*00/ID*01 Control Request CNTWD Format

The BB (driver bypass), NB (Nonbuffered mode), UE (User Error), and LU (Logical Unit designator) bits are as defined in the Control Request Parameter Definitions section of Chapter 1. The following control functions are available through ID*00/ID*01:

Code	Function
00	Clear and Reset Card
06B	Dynamic Status
23B	Control Asynchronous Interrupt
*31B	Enable Modem Environment
32B	Disable Modem Environment
43B	Enable/Disable Error Checking

*ID*01 only, refer to the ID*01 Modem Capabilities section of this chapter for more information.

Function Code 00: Clear and Reset Card

The clear and reset performs a full reset of the asynchronous serial interface card, as follows:

Feature	Setting
Special character select:	Disabled
Parity, framing, overrun interrupts:	Disabled
Echo:	Disabled
Transmit mode:	Disabled
Receive mode:	Disabled
Seven-bit character length:	Enabled
Modem interrupts:	Disabled
All 449 output control lines:	ON
Parity generation, checking:	Disabled
Break character flag:	ON
Baud rate:	Set to hardwire select lines
PE, FE, OE flags:	Cleared
Data Received (DR) flag:	Cleared
Transmission in progress:	Aborted

Function Code 06B: Dynamic Status

A dynamic status request returns the current device status to DVT16 through DVT19, where it can be read by a call to RMPAR. The format of the status information is shown in Figure J-24.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<	x	x	0						Error Code							
DVT17<<	Transmission Log = 0															
DVT18<<	ASIC Status															
DVT19<<	Card Control Word															

Figure J-24. DVT16 to DVT19 Status Words Format

DVT16 Bits 14 and 15 are used by the system. The following error codes can be contained in bits 0 through 5:

- 03 Timeout. The transmission log (DVT17) is set to zero; DMA operation is terminated; card status is updated in DVT18; ID*00/ID*01 determines whether or not to re-enable asynchronous interrupt on completion; the device is set down.
- 05 Transmission error. One of the overrun, framing, or parity error bits is set, except in the case of control requests, or when a break character is detected. The transmission log is set to zero; DMA operation is terminated; card status is updated in DVT18; ID*00/ID*01 determines whether to re-enable asynchronous interrupt upon completion; the device is set down.

77B Abort. The transmission log is set to zero; DMA operation is terminated; card status is updated in DVT18; ID*00/ID*01 determines whether or not to re-enable asynchronous interrupt on completion.

If a system powerfail is detected, and if the card was busy at powerfail, DMA operation is terminated and the request is reissued. If the card was not busy at powerfail, ID*00/ID*01 determines whether or not to re-enable asynchronous interrupts, and waits for a new request.

DVT17 is the transmission log (always 0).

DVT18 is a status word. It is often necessary to interrogate a terminal (or interface card) as to its status, in order to obtain such information as the cause of an interrupt or the state of a control circuit. Sixteen bits of information are available in a software accessible status word, shown in Figure J-25.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DV	BRK	PE	FE	OE	SRD	RR	EP	SCD	MSC	IC	CS	DM	RDC	TR	DR

Figure J-25. ASIC Status Format and Definition (DVT 18)

DV Data Valid:

- 1 – Received data is valid.
- 0 – An interrupt occurred during data input.

BRK Break Character:

- 1 – Break character detected on received data line.
- 0 – No break detected.

PE Parity Error:

- 1 – Parity error detected.
- 0 – No parity error detected.

FE Framing Error:

- 1 – Framing error detected.
- 0 – No framing error detected.

OE Overrun Error:

- 1 – Overrun error detected.
- 0 – No overrun error detected.

SRD Secondary Received Data:

- 1 – Mark (logic 1).
- 0 – Space (logic 0).

- RR Receiver Ready:
- 1 – Receiver ready OFF.
 - 0 – Receiver ready ON.
- EP Enable Parity:
- 1 – Even parity enabled.
 - 0 – Odd parity enabled.
- SCD Special Character Detect:
- 1 – Special character not detected.
 - 0 – Special character detected.
- MSC Modem Line Status Change:
- 1 – MIE bit not set; all modem input lines equal reference status.
 - 0 – MIE bit set; at least one input line differs from reference state.
- IC Incoming Call:
- 1 – Incoming call line OFF.
 - 0 – Incoming call line ON.
- CS (Clear to Send):
- 1 – Clear-to-send line OFF.
 - 0 – Clear-to-send line ON.
- DM (Data Mode):
- 1 – Data mode line OFF.
 - 0 – Data mode line ON.
- RCD (Received Data Line):
- 1 – Mark (logic 1).
 - 0 – Space (logic 0).
- TR (Transmitter Ready):
- 1 – Transmitter has transferred data; ready for new data.
 - 0 – Transmitter busy.
- DR (Data Received):
- 1 – Data has been received from device.
 - 0 – No new data has been received since last STC.
- DVT19 contains a copy of the card control word last used by the ASIC.

Function Code 23B: Control Asynchronous Interrupts

This request will enable asynchronous interrupts when PRAM1 is set to 0, or will disable asynchronous interrupts when PRAM1 is set to any other value. Enabling asynchronous interrupts allows you to interrupt by hitting any key on the terminal. Refer to the appropriate device driver documentation for further information on asynchronous interrupts at the device driver level.

Function Code 43B: Control Error Checking

This request will enable or disable framing and/or parity error checking. The default mode is no error checking. PRAM1 will contain the enable and/or disable conditions in bits 12 and 13:

Bit 12: 1 – Framing error enable.
0 – Framing error disable.

Bit 13: 1 – Parity error enable.
0 – Parity error disable.

ID*00/ID*01 Status Reporting

The call sequence for the status request is

```
CALL EXEC (13, CNTWD, STAT1 [,STAT2 [,STAT3 [,STAT4]]])
```

Control Word CNTWD

Figure J-26 shows the control word format for an EXEC status request for ID*00/ID*01.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			Z	BI						X					

Figure J-26. ID*00/ID*01 Status Request CNTWD Format

STAT1 and STAT2 Status Parameters

Figure J-27 shows the format of the returned status parameters STAT1 and STAT2.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT1 <<	X	X	Device Type = xx						Error Code							
STAT2 <<	AV		Interface Type = 00						0	0	I/O Select Code					

Figure J-27. STAT1 and STAT2 Status Words Format

In STAT1, the AV (device Availability) bits and Device Type are defined in the Status Request Parameter Definitions section in Chapter 1. The error code information contained in bits 0 through 7 is defined on a device level; refer to the associated device documentation for information about any returned error codes.

In STAT2, the Interface Type is 00, corresponding to the asynchronous serial interface card. The I/O select code is defined at system generation for each specific device, and is set through switches on the card itself.

STAT3 and STAT4 Status Parameters

If the control word Z-bit is 0, STAT3 returns the first word of the driver parameter area, and STAT4 returns the second word of the driver parameter area.

If Z is 1, STAT3 is the buffer to return the driver parameter area and STAT4 is the length of the STAT3 buffer.

Driver Parameter Area

Any words returned from the driver parameter area to STAT3/STAT4, or to the status buffer described by STAT3/STAT4, are device dependent. To determine the definition of the returned information, consult the appropriate device driver documentation.

ID*00/ID*01 Extended Status

As discussed in the Extended Device Status section of Chapter 1, the extended status can be obtained by making a call to RMPAR after an unbuffered EXEC request to determine the status of the request. Figure J-28 shows the format of the returned status.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVT16<<	X		0						Error Code							
DVT17<<	Transmission Log = 0															
DVT18<<	ASIC Status															
DVT19<<	Card Control Word															

Figure J-28. DVT16 to DVT19 Status Words Format

The content of DVT16 through DVT19 is identical to that defined for Function Code 06B, Dynamic Status. The only difference between extended status and dynamic status is that the dynamic status obtains the immediate status of the card while the extended status obtains the status after the last EXEC read, write or control request completed.

ID*01 Modem Capabilities

The ID*01 driver is an enhanced version of the ID*00 for the purpose of providing communication between a host computer and a terminal through a modem link. The ID*01 modem control capabilities are described below.

ID*01 Alarm Program Scheme

ID*01 will interact with an alarm program for the effective management of modems. Although the user can substitute his own alarm program, program “MODEM” is supplied with ID*01 and should suffice in most situations.

ID*01 will schedule program “MODEM” if any situation occurs that requires the user be notified and/or active programs be cleaned up. If, for instance, the user experiences a line disconnect, program “MODEM” will optionally log the event on a designated log device as well as optionally abort any programs associated with the modem terminal IFT.

Note Since the execution time of some functions may depend upon the time out value of the remote terminal, a low time out value such as 6000 (1.0 minute) or less is recommended.

ID*01 Modem Control Requests

ID*01 control requests can be issued programmatically or interactively. Programmatically, the call sequence is

```
CALL EXEC (3, CNTWD [,PRAM1 [,PRAM2 [,PRAM3 [,PRAM4]]]])
```

Interactively, the CN command is used.

The control requests could be included in the WELCOM file for automatic execution upon boot-up so that the modems on the system are immediately armed for dial-in. In the following descriptions of function code parameters the term “auto-answer” refers to the case where a user dials the host. The term “manual dial-out” refers to the case where the host dials a remote user.

Function Code 31B: Activate for Modem Environment

The calling sequence is the following:

:CN,modem terminal lu,31B,PRAM1,PRAM2,PRAM3,PRAM4

Where PRAM1 – PRAM4 are described as follows:

PRAM1

PRAM1 contains the modem operating information, as shown in Figure J-29.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AA	BB	R/D	FC	HU	Unused			Log LU							

Figure J-29. PRAM1 Format For Calling Sequence

- Log LU** The LU number (0 - 377B) of the device to which the alarm program “MODEM” sends messages. So that “MODEM” does not become I/O suspended, the log device should be a buffered, non-modem device. The default value is 0.
- HU** Hang Up. If AA (bit 15) is 1, HU is “don’t care.”
- 0 – Do not hang up on the current user (default). If a control 31B request enters ID*01 after a modem connection has been established, the connection is not interrupted. Only the parameters supplied are updated; the others are unchanged.
 - 1 – Hang up on current user and re-arm for auto-answer dial in.
- FC** Force Clean Up bit.
- 0 – Force clean up (default). If a user has a mode connection with a terminal in use or is primed for an auto-answer dial-in with the BB bit set, then a subsequent control request 32B executes as usual, except that the BB bit is forced to 0. BB = 0 schedules “MODEM”, which terminates any active programs associated with the terminal IFT before it deactivates the modem environment (refer to request 32B). This will avoid program I/O to an uninitialized LU which could down an LU or create an I/O suspend condition.
 - 1 – Do not force clean up. In the same conditions as above, the BB bit will not be forced to zero. ID*01 will be deactivated (as if no control request 23B had been issued). Any active programs still active that attempt I/O to the terminal may be I/O suspended, the LU may be set down, or both. If this happens, activate the LU (with control request 23B, and set the DVT up. If an important program must not be aborted, clean up should not be forced.

- R/D Rearm/Disarm Port bit. If BB = 1, R/D is a “don’t care.”
- 0 – If the alarm program is needed but is unavailable, disarm the port and wait for operator intervention, as when security is more important than port availability. (default)
 - 1 – If the alarm program is needed but it is unavailable, re-arm the port for an auto-answer re-dial anyway, as when port availability is paramount.
- BB The Benign Bit.
- 0 – If there is a line disconnect, “MODEM” aborts any programs associated with the IFT and notifies the log LU. (default)
 - 1 – Benign handling of user programs is in effect. In the event of a line disconnect the alarm program “MODEM” will only notify the designated log LU. Active programs associated with the modem terminal IFT will not be affected.
- AA (default) is the Auto Answer bit.
- 0 – Select auto answer. ID*01 arms the 12005A/modem for automatic answer.
 - 1 – Manual dial out. ID*01 allows up to 30 seconds for the user to dial a remote terminal from the computer modem and complete the modem connect sequence. Any current connection will be terminated.

PRAM2, PRAM3, PRAM4

PRAM2, PRAM3, and PRAM4 contain the name of the alarm program. If an alarm program is specified, it must have an ID segment assigned prior to the 31B request.

A sample command to initialize ID*01 for modem operation is shown below. In the example, LU4 is armed for auto-answer, the alarm program “MODEM” is enabled, the log LU is 6B, and BB is set.

```
:CN,4,31B,40006B,MO,DE,M
```

Once the modem connect sequence has completed (either manually or automatically), ID*01 will enable asynchronous interrupt handling for the user terminal as if a control 20B (enable primary program scheduling) had been sent to DD*00 with FMGR as the default primary program.

If, for example, the user wants to assign dedicated primary and secondary programs to a modem connected terminal (LU 4 in this example), FMGR4 and COMD4 could be made available as follows:

- a. In the boot file include:

```
:RP,FMGR,FMGR4  
:RP,COMND,COMD4
```

- b. In the WELCOM file include:

```
:CN,4,20B,FM,GR,4 A  
:CN,4,40B,CO,MD,4 A  
:CN,4,31B,1,MO,DE,M
```

With these file entries, when the user dials in and establishes the modem connection, the primary and secondary programs will be FMGR and COMD4, respectively.

Once a valid connection has been established, another control 31B will have the effect of updating the information in PRAM1 through PRAM4 as long as PRAM1 bit 11 (HU bit) is set to zero. (If PRAM2 through PRAM4 are omitted, the alarm program name that ID*01 had already saved will still be used.) In other words, once a valid connection has been established either a physical disconnect (that is hang up or alteration of modem switches), or a control 31B with bit 11 in PRAM1 set to 1, or a control 32B will terminate the connection.

If a physical disconnect occurs, or a control 31B with the HU bit (bit 11) set to one is executed, ID*01 will attempt to re-arm the host modem for auto-answer. If a control 32B was executed, ID*01 will convert to an uninitialized ID*00 and will no longer be aware of the modem environment.

Caution

Since the 12005A powers up with the DTR signal high, it is possible to establish a modem connection (in the U.S.A.) even though ID*01 is not enabled for modem operation (control 31B was not executed).

If at any time another user executes a control 31B, the line will be “hung up” and re-armed for a new modem connection while the non-valid connection via power-up will be interrupted. Also the special features of ID*01 for modem operation will not be available to the user that made a connection via the power-up DTR signal.

Note

1. In the event of an untimely disconnect, the current DMA transfer will be aborted. Critical operations such as I/O to CTU's should be restarted after the re-dial. (Some terminals may require a soft reset to recover.)
 2. If the CPU modem has the ability to force the DTR signal active, the CPU capability must not be used. Instead, ID*01 will programmatically manage the DTR signal.
-

Function Code 32B: Deactivate Modem Environment

The interactive calling sequence is:

```
:CN,modem terminal lu,32B
```

A control 32B will hang up the line if a user is connected. All modem environment indicators used by ID*01 are cleared, the modem is disarmed, and the handling of asynchronous interrupts is discontinued.

After a control 32B is executed, a control 20B (enable primary program scheduling) will activate the port for a non-modem environment as if ID*00 were being used.

ID*01 Error Message

ID*01 will issue an error message 21B and disable the port if conditions indicate a modem, hardware problem or a missing alarm program during control 31B execution. After the condition has been rectified by operator intervention, a control 31B will re-arm the port.

ID*01 Manual Dial-Out Considerations

ID*01 allows for the special case in which a computer operator at the CPU can dial out and establish a modem connection with an outside user terminal (that is control 31B PRAM1 bit AA bit=1). This would allow the computer site to absorb telephone charges.

The procedure given below was established for the 103/202 compatible Vadic 3451 modem. The procedure may or may not be exactly the same for other 103/202 compatible modems. The procedure and special precautions are as follows:

At a terminal near the CPU modem, type in the following:

```
:CN,remote modem terminal lu,140001B,MO,DE,M<cr>  
(BB = 1, manual answer, Log LU = 1)
```

The operator has 30 seconds to dial the remote terminal modem (on which DTR must be forced active for auto-answer), hear the responding tone, set the VO-DA switch to DA and hang up. The remote user can then use his terminal as usual.

When the remote user wants to end his connection, he must inform the computer operator, who sets the VO-DA switch back to VO. ID*01 senses the line disconnect, re-arms the port for auto-answer and schedules "MODEM."

The program "MODEM" prints a disconnect message on the log LU and terminates the user programs according to the BB setting passed with the last 31B control request.

If 30 seconds passes before the operator completes the connection:

- a. ID*01 tries to hang up the CPU modem. If successful,
- b. ID*01 re-arms the port for auto-answer, and schedules
- c. "MODEM," which sends a CONNECT SEQUENCE TIME OUT message.

Note

If the computer operator moves the DA-VO switch from its normal VO position to DA for manual dial-out and does not switch it back, ID*01 can not hang up the CPU modem, and assumes a hardware failure. It prints a message on the log LU and disables the port. Move the switch back to VO and issue another control 31B request.

ID*01 Alarm Program MODEM

Program MODEM used in the ID*01/modem scheme consists of the following:

- a. Main program %MODEM (92077-16391)
- b. Library %MDMLB (92077-16392) (assembled for RTE-A)

Program “MODEM” provides the designated log LU with an identification of the interface card involved (12005A) along with its status and the terminal LU.

A typical output message is as follows:

```
MDM: CONNECT SEQ. TIMEOUT
MDM: 12005A ST= 105777 LU 4
```

There are four different messages provided by “MODEM” as follows:

```
MDM: ****MODEM DOWN****
MDM: disconnect SEQ. TIMEOUT
MDM: 12005A ST=XXXXXX LU XX
```

```
MDM: CONNECT SEQ. TIMEOUT
MDM: 12005A ST=XXXXXX LU XX
```

```
MDM: LINE disconnect DETECTED
MDM: 12005A ST=XXXXXX LU XX
```

```
MDM: POWER FAILURE ERROR
MDM: 12005A ST=XXXXXX LU XX
```

Referen to the *HP 12005A Asynchronous Interface Card Reference Manual* for the meaning of the status (ST) bits.

ID*01 Example of Modem Link Set-up

At boot-up, the following commands should be executed:

```
:RP,MODEM (make alarm program available to be scheduled)
:CN,4,31B,40001B,MO,DE,M      (arm modem/terminal LU 4 for auto-answer
                                "dial in", and the BB bit is set)
```

At this point a user may dial the phone number and establish the modem link. If the connection is then lost, for whatever reason, ID*01 modem driver will do the following:

- a. Post a message to the log device (1 in this example).
- b. Re-arm the port for auto-answer redial.
- c. Hold off all requests to that LU except redial or shut down (function 32B).

Since the BB bit was set, the user's FMGR will do continuous prompting to the terminal. The driver will flush each request after a 1.5 second wait until the connection is re-established. At this time, all I/O activity except the one DMA transfer that was aborted will continue. This scheme is designed to allow a control request 32B to be able to reach the driver. At this point, any user is free to dial in and continue where the previous user left off.

An alternative is to set the BB bit (14) to zero either at boot-up time or by first establishing the modem connection and executing the following command:

```
:CN,lu,31B,20001B      (as long as PRAM1 bit 11=0, the existing connection
                        will not be disturbed. Only the parameters will be
                        updated)
```

Since the BB bit has now been set to zero, the program doing I/O (e.g., FMGR) will be commanded to turn OFF by "MODEM" when the user hangs up. By setting bit 13 in the control word, port availability has priority over file security if the alarm program is not available.

This appendix contains the serial device and interface drivers that are no longer actively supported in the RTE-A operating system as of Revision 5000.

Index

Symbols

\$DTCLB library routines, 2-83

- XCOMP, 2-89
- XCOPY, 2-88
- XDESC, 2-87
- XINMD, 2-87
- XRELD, 2-87
- XRELS, 2-87
- XRQST, 2-87

A

- A- and B-Register contents, 1-9, 1-15
- AH: auto-home bit, serial I/O drivers, 2-154
- alarm program schedule, ID*01, J-78
- ASCII vs. binary read modes, 2-147
- ASIC interface drivers, ID*00/ID*01, J-66
- auto-home bit, read request, 2-152
- AUTOR program, H-2

B

- baud rates, supported, 2-165
- binary bit, read request, 2-152
- binary vs. ASCII read modes, 2-147
- block mode read, serial I/O drivers, 2-153
- BRG range, 2-165
 - ID800/ID801 default, 2-175
- buffered read, 2-179
- bypass bit (bit 15), 2-149

C

- CALLB, H-5
- carriage control, capabilities, 2-181
- choosing, correct driver, E-4
- CNTWD parameter, 1-5, 1-12, 1-17
- Command Descriptor Block (CDB), 2-35
- comparison, serial drivers, C-1
- control calls (DDQ30), 2-62
- control parameter definitions, 1-12
 - CNTWD, 1-12
 - ecode, 1-12
 - KEYWD, 1-15
 - PRAM1 through PRAM4, 1-14
 - ZERO, 1-15
- control request calling format, 1-11
- control request conventions, 1-11
- control request, serial I/O drivers, 2-158
 - control word CNTWD, 2-158
 - dynamic status special forms, 2-164
 - ID400/ID800 firmware revision codes, 2-164
 - ENQ/ACK handshake details, 2-185

- function code 11B: line spacing/page eject, 2-164
- function code 16B: configure baud rate generator, 2-165
- function code 17B: definable terminator, 2-166
- function code 20B: enable prog scheduling, 2-167
 - pass program name, 2-167
 - program scheduling conditions, 2-167
 - RTE compatibility: pass program name, 2-168
- function code 21B: disable prog scheduling, 2-168
- function code 22B: set device timeout, 2-169
- function code 25B: read HP terminal straps, 2-169
- function code 26B: flush input buffers, 2-169
- function code 30B: set port configuration, 2-170
 - default BRG ranges (ID800/ID801 only), 2-175
- function code 31B: modem environment, 2-177
- function code 32B: generate break, 2-177
- function code 33B: FIFO buffer mode control, 2-178
- function code 34B: set port protocol, 2-180
- function code 35B: reset baud rate generator, 2-186
- function code 40B: enable prog scheduling, 2-186
- function code 41B: disable prog scheduling, 2-186
- function code 6B: dynamic status, 2-159
 - driver defined errors, 2-162
 - DVT16: error code, 2-161
 - DVT17: transmission log, 2-163
 - DVT18: interface card status, 2-163
 - DVT19: interface driver information, 2-163
 - DVT20: driver communication area, 2-164
 - DVT6: device status, 2-160
 - system defined errors, 2-161
- CS/80, disk driver DD*33. *See* DD*33 CS/80 disk device driver
- CTU device driver, DD*20, J-18

D

- DD*00, 2-145
 - terminal device driver, J-1
- DD*12 line printer driver, 2-2
 - control request, 2-3
 - control word CNTWD, 2-4
 - function code 00: clear & reset, 2-4
 - function code 11B: line skipping & form feed, 2-4
 - extended status request, 2-6
 - status request, 2-5

- control word CNTWD, 2-5
- STAT1 and STAT2 parameters, 2-6
- STAT3 and STAT4 parameters, 2-6
- write request, 2-2
 - A- and B-Register contents, 2-3
 - BUFR and BUFLN, 2-3
 - control word CNTWD, 2-2
- DD*20, 2-145
- DD*20 CTU device driver, J-18
- DD*23 magnetic tape driver, 2-22
 - control request, 2-23
 - extended status request, 2-25
 - read/write request, 2-22
 - A- and B-Register contents, 2-22
 - BUFR and BUFLN, 2-22
 - control word CNTWD, 2-22
 - status request, 2-24
 - control word CNTWD, 2-24
 - STAT1 and STAT2 parameters, 2-24
 - STAT3 and STAT4 parameters, 2-25
- DD*24 magnetic tape driver, 2-27
 - control request, 2-28
 - A- and B-Register returns, 2-30
 - control word CNTWD, 2-28
 - extended status request, 2-31
 - read/write request, 2-27
 - A- and B-Register returns, 2-28
 - BUFR and BUFLN, 2-28
 - control word CNTWD, 2-27
 - status request, 2-30
 - control word CNTWD, 2-30
 - STAT1 and STAT2 parameters, 2-30
 - STAT3 and STAT4 parameters, 2-31
- DD*30 disk device driver, 2-46
 - control request, 2-47
 - error information, 2-52
 - extended status request, 2-49
 - read/write request, 2-46
 - A- and B-Register contents, 2-47
 - BUFR and BUFLN, 2-46
 - control word CNTWD, 2-46
 - TRACK and SECTOR parameters, 2-47
 - status request, 2-48
 - control word CNTWD, 2-48
 - driver parameter area, 2-49
 - STAT1 and STAT2 parameters, 2-48
 - STAT3 and STAT4 parameters, 2-49
- DD*33 CS/80 disk device driver, 2-68
 - access errors field (Class 3), 2-77
 - bad tape indicator, 2-91
 - control request, 2-70
 - control word CNTWD, 2-70
 - CTD control request, 2-80
 - clear cache, 2-81
 - close cache, 2-81
 - unload tape, 2-82
 - using the cartridge tape drive (CTD), 2-80
 - CTD read/write request, 2-79
 - BUFR and BUFLN, 2-79
 - CTD A- and B-Register contents, 2-80
- HIBLK, LOBLK, 2-79
- direct disk control, 2-83
 - calling format, 2-83
 - complementary command array (ICOMP), 2-84
 - ICOMP array, 2-84
 - XCOMP routine, 2-89
 - XCOPY routine, 2-88
 - XDESC routine, 2-87
 - XINMD routine, 2-87
 - XRELD routine, 2-87
 - XRELS routine, 2-87
 - XRQST routine, 2-87
- driver parameter area, 2-73
- error information, 2-89
- EXEC function and subfunction codes, 2-81
- extended status, 2-73
- fault errors field (Class 2), 2-76
- information errors field (Class 4), 2-78
- read/write request, 2-68
 - A- and B-Register contents, 2-70
 - BUFR and BUFLN, 2-69
 - control word CNTWD, 2-69
 - TRACK and SECTOR, 2-69
- read/write request (CTD), 2-79
- reject errors field (Class 1), 2-75
- retry after error, 2-90
- status request, 2-71
 - control word CNTWD, 2-71
 - STAT1 and STAT2 parameters, 2-71
 - STAT3 and STAT4 parameters, 2-72
- DDC00, 2-145, 2-148
- DDC01, 2-145, 2-148
- DDC12 printer driver, 2-8
 - control request, 2-12
 - control word CNTWD, 2-12
 - error information, 2-20
 - extended status request, 2-19
 - powerfail recovery, 2-8
 - printer, bringing up automatically, 2-8
 - status request, 2-17
 - control word CNTWD, 2-17
 - STAT1 and STAT2 parameters, 2-17
 - STAT3 and STAT4 parameters, 2-18
- VFC definition request, 2-11
 - BUFR and BUFLN, 2-11
- write request, 2-9
 - BUFR and BUFLN, 2-11
 - control word CNTWD, 2-9
- DDM30 disk device driver, 2-54
 - control request, 2-55
 - driver parameter area, 2-57
 - error information, 2-59
 - extended status request, 2-57
 - read/write request, 2-54
 - A- and B-Register contents, 2-55
 - BUFR and BUFLN, 2-54
 - control word CNTWD, 2-54
 - TRACK and SECTOR parameters, 2-55
 - status request, 2-56

- control word CNTWD, 2-56
- STAT1 and STAT2 parameters, 2-56
- STAT3 and STAT4 parameters, 2-56
- DDQ24 SCSI tape device driver, 2-34
- calling sequence parameters, control word bits, 2-34, 2-36
- control calls, 2-36
- control commands
 - control 0B/4B rewind, 2-37
 - control 10B backward space set mark, 2-40
 - control 11B forward space set mark, 2-40
 - control 13B forward space file, 2-40
 - control 14B backward space set mark, 2-41
 - control 15B set tape density, compression, 2-41
 - control 16B enable/disable request sense, 2-42
 - control 1B write file mark, 2-37
 - control 2B backward space record, 2-38
 - control 3B forward space record, 2-38
 - control 5B rewind and unload tape, 2-38
 - control 6B dynamic status, 2-39
 - control 7B write set mark, 2-39
- control word bits
 - BUFLN buffer length, 2-35
 - BUFR buffer address, 2-34
- driver parameter table, 2-42
- read and write calls, 2-34
- read SCSI command and sense data, 2-35
- status, 2-42
 - communication word – DVT20, 2-45
 - driver error – DVT16, 2-43
 - logical unit status – DVT6, 2-43
 - SCSI & transaction status – DVT18, 2-44
 - sense key & additional sense code – DVT19, 2-45
 - transmission log – DVT17, 2-44
- Z-buffer calls, 2-35
 - COMMAND descriptor block (CDB), 2-35
 - length of command descriptor block, 2-35
- DDQ30 SCSI disk device driver, 2-61
- control calls, 2-62
 - control 16B enable/disable request sense, 2-63
 - control 76B copy RTE block information, 2-63
- driver communication word – DVT20, 2-67
- driver parameter table – DVP, 2-64
 - driver error – DVT16, 2-65
 - logical unit status – DVT6, 2-65
 - SCSI & transaction status – DVT18, 2-66
 - sense key & additional sense code – DVT19, 2-66
 - transmission log – DVT17, 2-66
- read and write calls, 2-61
 - BUFLN buffer length, 2-62
 - BUFR buffer address, 2-62
 - control word bits, 2-61
 - control word CNTWD, 2-61

- TRACK and SECTOR addresses, 2-62
- read/write requests, 2-61
- Z-Buffer calls, 2-62
- DDQ35 SCSI disk device driver, driver parameter table – DVP, driver error – DVT16, 2-100
- dev/intfc driver comm, serial I/O drivers, multibuffer linked list structure, 2-189
- device
 - driver
 - serial I/O, J-1
 - terminal, J-1
 - drivers, 2-1
 - table, 1-2
- device and interface
 - driver communication, serial drivers, 2-187
 - driver separation, 1-1
- DIAL program, J-62
- direct, disk control, DD*33, 2-83
- disk
 - device driver, DD*30. *See* DD*30 disk device driver
 - interface driver, ID*27. *See* ID*27 disk interface driver
- disk device driver DDM30. *See* DDM30 disk device driver
- driver
 - choosing correct serial driver, E-4
 - comparison of serial drivers, C-1
 - DD*00, J-1
 - DD*20, J-18
 - ID*00/ID*01, J-66
 - IDM00, J-39
- drivers, comparison of serial drivers, 2-145
- DVT, definition, 1-2
- DVT 20, bit definitions, serial driver, 2-187

E

- echo bit, read request, 2-151
- ecode parameter, 1-5, 1-12, 1-17
 - no-abort bit, 1-22
- ENQ/ACK handshake, serial I/O driver, 2-185
- error information
 - DD*30 driver, 2-52
 - DD*33, 2-89
 - DDC12 driver, 2-20
 - DDM30 disk device driver, 2-59
 - ID*27 disk interface driver, 2-98
 - ID*36 PROM storage module driver, 2-104
- example
 - page mode application, F-1
 - protocol charts, D-1
- EXEC, error returns, 1-21
- extended device status, 1-19

F

- full duplex, pseudo (A400 only), serial I/O drivers, 2-153

G

generating, serial drivers into your system, E-1
GPIO/parallel interface card driver ID*50. *See*
ID*50 GPIO/parallel interface card driver

H

HpCrtParityChk, 2-173
HpCrtParityGen, 2-173
HpCrtReadChar, 2-150
HP-IB
 interface card driver, ID*37. *See* ID*37 HP-IB
 interface card driver
 line printer driver, DD*12. *See* DD*12 line
 printer driver
 magnetic tape driver
 DD*23. *See* DD*23 magnetic tape driver
 DD*24. *See* DD*24 magnetic tape driver
 printer driver, DDC12. *See* DDC12 printer
 driver
HPMDM, H-2

I

I/O, request, conventions, 1-1
ICOMP, complementary command array, 2-84
ID*00, 2-145, J-66
ID*01, 2-145, J-66
 modem capabilities, J-78
ID*27 disk interface driver, 2-92
 control request, 2-93
 error information, 2-98
 extended status, 2-95
 error codes from DVT16, 2-96
 status codes from DVT18, 2-97
 parameter area, 2-95
 read/write request, 2-92
 A- and B-Register contents, 2-93
 BUFR and BUFLN, 2-92
 CNTWD read request example, 2-93
 control word CNTWD, 2-92
 TRACK and SECTOR, 2-92
 status request, 2-94
 control word CNTWD, 2-94
 STAT1 and STAT2 parameters, 2-94
 STAT3 and STAT4 parameters, 2-95
ID*36 PROM storage module driver, 2-101
 control request, 2-102
 error information, 2-104
 extended status, 2-103
 parameter area, 2-104
 read/write request, 2-101
 A- and B-Register contents, 2-102
 BUFR and BUFLN, 2-101
 control word CNTWD, 2-101
 example read request, 2-102
 TRACK and SECTOR, 2-101
 status reporting, 2-102
 control word CNTWD, 2-102

 STAT1 through STAT4 parameters, 2-103
ID*37 HP-IB interface card driver, 2-105
 control request, auto-addressed, 2-109
 function code 00: selected dev clear (SDC),
 2-110
 function code 06B: device dynamic status,
 2-110
 function code 16B: set REN true, 2-111
 function code 17B: go to local (GTL), 2-111
 function code 20B: SRQ prog scheduling,
 2-111
 function code 21B: disable SRQ prog schedul-
 ing, 2-112
 function code 22B: set intfc driver timeout,
 2-112
 function code 23B: parallel poll assignment,
 2-112
 function code 24B: set device address, 2-112
 function code 27B: group execute trigger
 (GET), 2-113
 function code 30B: disable SRQ interrupts,
 2-113
 function code 31B: restore SRQ interrupts,
 2-113
 function code 40B: enable parallel poll sched-
 uling, 2-114
 function code 41B: disable parallel poll sched-
 uling, 2-114
 control request, direct I/O, 2-116
 control word CNTWD, 2-116
 function code 00: clear and reset device, 2-117
 function code 06B: dynamic bus status, 2-117
 function code 16B: set REN true, 2-118
 function code 17B: go to local (GTL), 2-118
 function code 23B: parallel poll configure,
 2-118
 function code 25B: local lockout (LLO), 2-119
 function code 27B: group execute trigger,
 2-119
 function code 40B: enable parallel poll sched-
 uling, 2-119
 function code 41B: disable parallel poll sched-
 uling, 2-119
 function code 51B: bailout (ABORT), 2-119
 PRAM3 and PRAM4 control buffer descrip-
 tors, 2-119
 end-of-record processing, 2-107
 extended status, 2-121
 operating modes, 2-106
 auto-addressing, 2-106
 direct I/O, 2-106
 read/write request, auto-addressed, 2-108
 BUFR and BUFLN, 2-109
 control word CNTWD, 2-108
 PRAM3 – secondary address, 2-109
 read/write requests, direct I/O, 2-114
 BUFR and BUFLN, 2-115
 control word CNTWD, 2-115
 device addresses, 2-115

- PRAM3 and PRAM4 control buffer descriptors, 2-116
 - status request, 2-120
 - control word CNTWD, 2-120
 - driver parameter area, 2-120
 - STAT1 and STAT2 parameters, 2-120
 - STAT3 and STAT4 parameters, 2-120
 - universal commands, 2-121
 - ID*50 GPIO/parallel interface card driver
 - control requests, 2-129
 - control word CNTWD, 2-129
 - function code 00: clear and reset card, 2-129
 - function code 06B: dynamic status of card, 2-129
 - function code 20B: enable program scheduling, 2-130
 - function code 21B: disable program scheduling, 2-130
 - function code 23B: enable/disable async interrupts, 2-130
 - function code 24B: set PIC control lines, 2-131
 - function code 40B: configure card control word, 2-131
 - parallel interface card control register, 2-134
 - parallel interface card status, 2-136
 - program scheduling, 2-137
 - read/write requests, 2-128
 - BUFR and BUFLN, 2-128
 - control word CNTWD, 2-128
 - PRAM1 optional parameter, 2-128
 - status reporting, 2-133
 - control word CNTWD, 2-133
 - driver parameter area, 2-134
 - STAT1 through STAT4 parameters, 2-134
 - ID*52 PIC intercomputer comm driver, 2-138
 - control requests, 2-139
 - control word CNTWD, 2-139
 - function code 00: clear and reset card, 2-139
 - function code 01B: send EOF, 2-139
 - function code 06B: return dynamic status, 2-140
 - function code 11B: top-of-form EOF, 2-140
 - function code 20B: enable program scheduling, 2-141
 - function code 22B: set timeout, 2-141
 - function code 45B: use level mode DVCMD, 2-141
 - function code 46B: use pulse mode DVCMD, 2-141
 - function codes 21B/23B: disable prog scheduling, 2-141
 - parallel interface card control register, 2-143
 - parallel interface card status, 2-144
 - read/write requests, 2-138
 - BUFR and BUFLN, WBUF and WBUFL, 2-138
 - control word CNTWD, 2-138
 - PRAM1 optional parameter, 2-138
 - status reporting, 2-142
 - control word CNTWD, 2-142
 - driver parameter area, 2-142
 - STAT1 through STAT4 parameters, 2-142
 - ID100 interface driver, 2-147
 - ID101 interface driver, 2-147
 - ID400 interface driver, 2-147
 - firmware revision code, 2-164
 - ID800 interface driver, 2-147
 - baud rate generator, 2-165
 - BRG ranges (default), 2-175
 - firmware revision code, 2-164
 - ID801 interface driver, 2-147
 - IDM00, 2-145, J-39
 - for HP 12040A MUX, J-27
 - IDQ35 SCSI interface driver, 2-99
 - DVT status, 2-99
 - DVT transformation, 2-99
 - IDR27 RAM disk interface driver, 2-123
 - configuration request, 2-125
 - control request, 2-124
 - deallocate request, 2-125
 - extended status, 2-127
 - parameter area, 2-127
 - read/write request, 2-123
 - A- and B-Register contents, 2-124
 - BUFR and BUFLN, 2-123
 - CNTWD read request example, 2-124
 - control word CNTWD, 2-123
 - TRACK and SECTOR, 2-123
 - status request, 2-126
 - control word CNTWD, 2-126
 - STAT1 and STAT2 parameters, 2-126
 - STAT3 and STAT4 parameters, 2-126
 - IDZ00 interface driver, 2-147
 - input editing, 2-151
 - interface, drivers, 2-1
 - interface table (IFT), 1-2
- ## K
- KEYWD parameter, 1-8, 1-15
- ## L
- language, identity codes, 2-15
 - line printer driver DD*12. *See* DD*12 line printer driver
- ## M
- magnetic tape, driver
 - DD*23. *See* DD*23 magnetic tape driver
 - DD*24. *See* DD*24 magnetic tape driver
 - modem, J-61
 - control, H-2
 - serial I/O drivers, 2-177
 - MODEM alarm program, J-83

P

- page mode application, F-1
- PIC intercomputer comm driver ID*52. *See* ID*52
 - PIC intercomputer comm driver
- PRAM1 through PRAM4 parameters, 1-14
- PRAM3 and PRAM4 parameters, 1-8
- printer configuration, automatic bring-up, 2-8
- printer/paper position bit, 2-181
- program(s), scheduling, G-1
- PROM storage module driver ID*36. *See* ID*36
 - PROM storage module driver
- protocol chart examples, D-1

R

- RAM disk, driver IDR37. *See* IDR27 RAM disk interface driver
- RD bit, 2-180
- read request, buffered, 2-179
- read request, serial I/O drivers, 2-150
 - block mode read, 2-153
 - AH: auto-home bit, 2-154
 - BUFR and BUFLN, 2-150
 - control word CNTWD, 2-151
 - auto-home bit, 2-152
 - binary bit, 2-152
 - echo bit, 2-151
 - pseudo full duplex (A400 only), 2-153
 - transparency bit, 2-151
 - Z: write/read bit, 2-151
 - special status read, 2-154
- read/write parameter definitions, 1-4
 - BUFLN, 1-8
 - BUFR, 1-8
 - CNTWD, 1-5
 - ecode, 1-5
 - KEYWD, 1-8
 - PRAM3 and PRAM4, 1-8
 - ZERO, 1-8
- read/write request conventions, 1-4
- remote terminal download, H-1
- return, data on timeout (RD bit), 2-180
- RMTERM, H-1

S

- sample, page mode application, F-1
- scheduling, a program, G-1
- SCSI
 - disk device driver DDQ30. *See* DDQ30 SCSI disk device driver
 - interface driver IDQ35. *See* IDQ35 SCSI interface driver
 - tape device driver DDQ24. *See* DDQ24 SCSI tape device driver
- security, callback, CALLB program, H-5
- serial I/O device drivers, 2-148, J-1
 - DDC00, 2-145
 - DDC01, 2-145

- slave device support using DDC01, 2-148
- serial I/O drivers, 2-145
 - bypass bit (bit 15), 2-149
 - prior to Rev. 4010, J-1
 - special characters, 2-146
- serial I/O interface drivers, 2-147
 - ID100, 2-145, 2-147
 - ID101, 2-145, 2-147
 - ID400, 2-145, 2-147
 - ID800, 2-145, 2-147
 - ID801, 2-145, 2-147
 - IDZ00, 2-145, 2-147
- special characters, serial I/O drivers
 - backspace, 2-146
 - break, 2-147
 - carriage return, 2-146
 - delete, 2-146
 - EOT, 2-146
 - line feed, 2-146
- STAT1 and STAT2 parameters, 1-17
- STAT3 and STAT4 parameters, 1-18
- status parameter definitions, 1-17
 - CNTWD, 1-17
 - ecode, 1-17
 - STAT1 and STAT2, 1-17
 - STAT3 and STAT4, 1-18
- status request
 - calling format, 1-16
 - conventions, 1-16
- supported baud rates, 2-165
- system, generation, serial drivers, E-1

T

- table
 - device (DVT), 1-2
 - interface (IFT), 1-2
- TELNET pseudo terminal LU, 2-149
- timeout, return data on, 2-180
- transparency bit, read request, 2-151
- type-ahead, IDM00, J-58

U

- UART (Universal Asynchronous Receiver/Transmitter), C-5

W

- write request, serial I/O drivers, 2-156
 - BUFR and BUFLN, 2-156
 - CNTWD, 2-156

X

- XCICL, \$DTCLB library, 2-83
- XCNCL, \$DTCLB library, 2-83
- XCOLD, \$DTCLB library, 2-83
- XCOMP, \$DTCLB library, 2-83, 2-89
- XCOPY, \$DTCLB library, 2-83, 2-88

XDESC, \$DTCLB library, 2-83, 2-87
XDIAG, \$DTCLB library, 2-83
XFMRK, \$DTCLB library, 2-83
XINMD, \$DTCLB library, 2-83, 2-87
XLCRD, \$DTCLB library, 2-83
XLCVF, \$DTCLB library, 2-83
XLCWR, \$DTCLB library, 2-83
XRELD, \$DTCLB library, 2-83, 2-87
XRELS, \$DTCLB library, 2-83, 2-87
XRLPB, \$DTCLB library, 2-83
XRQST, \$DTCLB library, 2-83, 2-87

XSDCL, \$DTCLB library, 2-83
XSPRE, \$DTCLB library, 2-83
XUNLD, \$DTCLB library, 2-83
XUTIL, \$DTCLB library, 2-83
XWLPB, \$DTCLB library, 2-83

Z

Z: write/read bit, serial I/O drivers, 2-151
Z-Buffer, 2-151
ZERO parameter, 1-8, 1-15

