MPG

USER'S MANUAL

MD-11-DTUMA-C

EP-DTUMA-C-DL-A

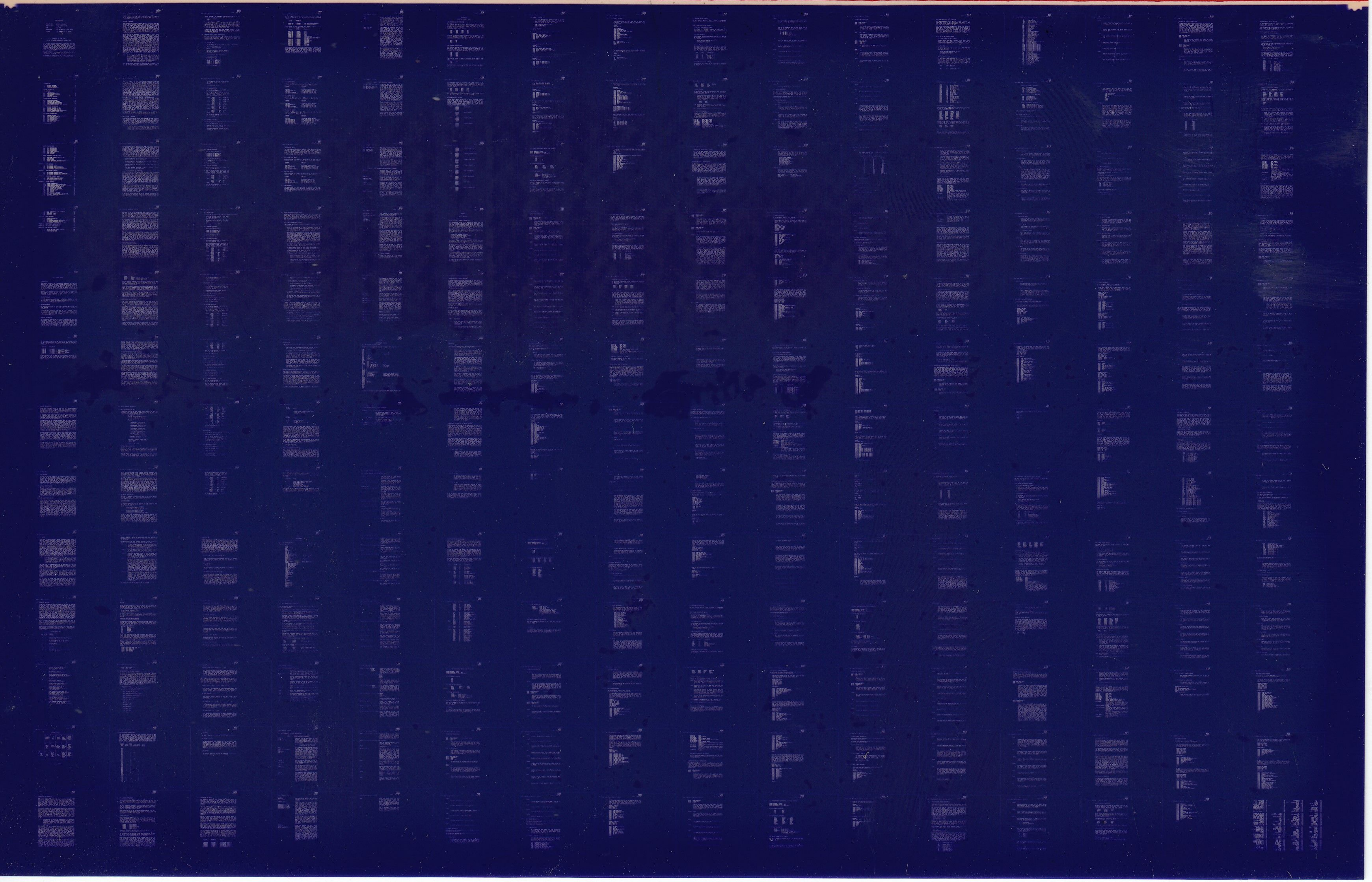
COPYRIGHT © 1976

FICHE 1 OF 2

NOV 1976

digital

MADE IN U.S.A.



MPG

USER'S MANUAL

MD-11-DTUMA-C

EP-DTUMA-C-DL-A

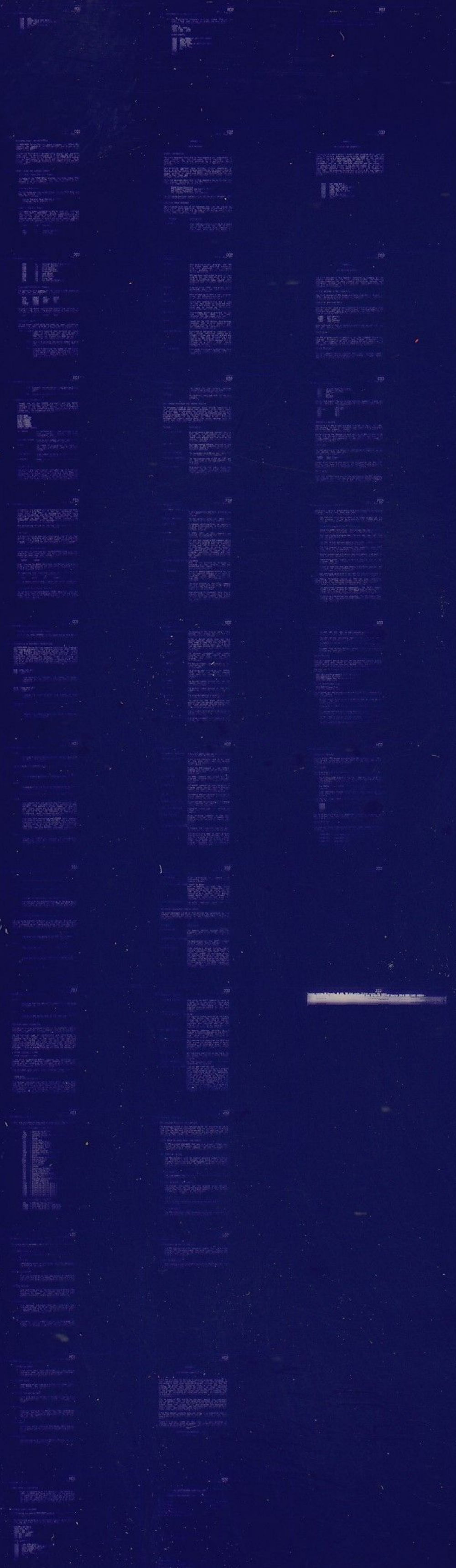
COPYRIGHT © 1976

FICHE 2 OF 2

NOV 1976

digital

MADE IN U.S.A.



IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DTUMA-C-D
PRODUCT NAME:	M.P.G. USERS' MANUAL
DATE:	JULY 1976
MAINTAINERS:	SYS. RELIABILITY / DIAG. GROUP
AUTHORS:	C. E. HARPER / W. R. GREENE / A. W. LEIGH

COPYRIGHT (C) 1975, 1976

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS

This software is furnished under a license for use only on a single computer system and may be copied only with the inclusion of the above copyright notice. This software, or any copies thereof, may not be provided or otherwise made available to any other person except for use on such system and to one who agrees to these license terms. Title to and ownership of the software shall at all times remain in DEC.

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation.

DEC assumes no responsibility for use or reliability of its software on equipment which is not supplied by DEC.

Table Of Contents Page

1.0 INTRODUCTION 5

1.1 RELATED PROGRAMS 5
1.2 RELATED DOCUMENTS 5
1.3 ORDERING INFORMATION 5

2.0 GENERAL INFORMATION 7

3.0 MPG FEATURES 8

3.1 USER PROGRAMS 8
3.2 MULTIPLE USER PROGRAMS 8
3.3 USER PROGRAM DATA AREAS 8
3.4 I/O AREAS 9
3.5 OPERATION SWITCHES 10

4.0 DESCRIPTION OF OPERATION 13

4.1 ENTERING A PROGRAM 14
4.2 EXECUTION OF A PROGRAM 15
4.3 INTERRUPTING USER PROGRAMS 17
4.4 USER PROGRAM MODIFICATIONS 18
4.5 SAVING AND FETCHING OF USER PROGRAMS 19

5.0 DEVICE SUPPORT INFORMATION 20

5.1 DEVICES REQUIRED FOR MPG 20
5.2 OPTIONAL DEVICES FOR MPG 20
5.3 MPG DEVICE USAGE DETAILS 21
5.4 DEVICES SUPPORTED FOR USER PROGRAMS 24
5.5 DEVICE SUPPORT SUMMARY TABLE 25

6.0 MPG LOADING INSTRUCTIONS 26

6.1 LOADING VIA XXDP 26
6.2 DEDICATED MEDIA LOADING INSTRUCTIONS 27
6.3 TC11 DECTAPE INPUT 27
6.4 RK11 DECPACK DISK INPUT 28
6.5 TM11 MAGTAPE INPUT 30
6.6 TM02 MAGTAPE INPUT 31
6.7 RX11 FLOPPY DISK INPUT 32
6.8 RP04 DISK INPUT 33

7.0 INITIATING MPG 35

Table Of Contents (Cont'd)

Page

8.0 GENERATING MPG MEDIA 39

8.1 TC11 DECTAPE MEDIA 41
8.2 RK11 DECPACK MEDIA 41
8.3 TM11 MAGTAPE MEDIA 41
8.4 TM02 MAGTAPE MEDIA 42
8.5 RX11 FLOPPY DISK MEDIA 42
8.6 RP04 DISK MEDIA 42
8.7 USER PROGRAMS 42

9.0 MEMORY MANAGEMENT VERSION OF MPG 43

9.1 ADDITIONAL COMMANDS AND FEATURES 43
9.2 TRAP ROUTINE 44
9.3 RESTRICTIONS 45
9.4 MEMORY MANAGEMENT IMPLEMENTATION DETAILS 45
9.5 UNIBUS MAP IMPLEMENTATION DETAILS 46

APPENDIX A - USER COMMANDS 48

A.1 MPG COMMAND SUMMARY 48
A.2 MPG COMMAND ENTRY INFORMATION 49
A.3 MPG COMMAND OPERAND DEFINITIONS 50
A.4 MPG COMMANDS - DETAILED DESCRIPTIONS 51

APPENDIX B - USER PROGRAM LANGUAGE 58

B.1 MPG PROGRAM LANGUAGE SUMMARY 58
B.2 MPG PROGRAM LANGUAGE VALID PARAMETERS 59
B.3 MPG PROGRAM LANGUAGE DETAILED DESCRIPTION 60

APPENDIX C - PREDEFINED SYMBOLIC NAMES 66

C.1 USER PROGRAM EXCLUSIVE STORAGE 66
C.2 USER PROGRAM COMMON STORAGE 66
C.3 MPG PREDEFINED BIT PATTERNS 67

APPENDIX D - DEVICE ROUTINES 69

D.1 GENERAL INFORMATION 69
D.1.1 COMMON FEATURES IN DEVICE ROUTINES 70
D.1.2 COMMON ERROR INFORMATION FOR DEVICE ROUTINES 72
D.2 MINIMUM SUPPORT DEVICE ROUTINES 74
D.3 DJ11 16-LINE MULTIPLEXOR 76
D.4 DL11 SINGLE LINE INTERFACE 81
D.5 DQ11 NPR SYNC. LINE INTERFACE 87
D.6 RK11 DECPACK DISK 95
D.7 TC11 DECTAPE 105
D.8 TM11 MAGTAPE 114
D.9 LP11/LS11/LV11 PRINTERS 123
D.10 DH11 16-LINE PROGRAMMABLE MULTIPLEXER 130
D.11 PC11/PR11 PAPER TAPE READERS AND PUNCH 140
D.12 RP04/RP05/RP06 DISKS 144

Table Of Contents (Cont'd) Page

D.13 DU11 SYNCHRONOUS LINE INTERFACE 164
D.14 RK06 DISK 178
D.15 RP02/RP03 DISKS 196
D.16 RS03/RS04 DISKS 209

APPENDIX E - ERROR MESSAGES 222

E.1 GENERAL INFORMATION 222
E.2 EXECUTIVE 222
E.3 MPG COMMAND PROCESSOR AND CONTROL ROUTINE 224
E.4 MPG MEMORY MANAGEMENT VERSION 228
E.5 MPG LANGUAGE PROCESSOR AND COMPILER 230

APPENDIX F - USER PROGRAM SUBMITTAL FORM 232

APPENDIX G - MPG CODING FORM GENERATION 235

APPENDIX H - MPG REVISION HISTORY 236

H.1 INITIAL RELEASE OF MPG (VERSION # 1) 236
H.2 VERSION # 2 RELEASE 237
H.3 VERSION # 3 RELEASE 240

M.P.G. USERS' MANUAL

1.0 INTRODUCTION

This Manual contains all information necessary for use and operation of the Maintenance Program Generator (MPG) program. MPG provides hardware oriented technical personnel with the ability to easily generate and execute programs on the PDP-11 series of computers. While these programs may be written so as to perform any type of task, the major application expected are programs that aid in the maintenance and repair of peripheral devices.

Appendix H of this manual contains a summary of the capabilities and support provided on the initial release of MPG. Also contained in this Appendix is the changes incorporated in subsequent releases (versions) of MPG.

1.1 RELATED PROGRAMS

The XXDP Update Program # 2 (UPD2 - MAINDEC # 11-DZQUB-H) is used for certain MPG media maintenance and creation functions. In addition, MPG is supplied on XXDP media and therefore is preceded by execution of the applicable XXDP Monitor.

1.2 RELATED DOCUMENTS

For details concerning the operation of the XXDP Monitor and the UPD2 program, refer to the XXDP User's Manual (MAINDEC # 11-DZQXA-A-D).

There is now available from SDC an MPG Summary Manual (11-DTSMA-A-D) which consists of excerpts from this document. It is considerably smaller in size and contains summarized information concerning MPG loading, commands, instructions, buffer areas, patterns, and Device Routines. This manual assumes a knowledgable MPG user.

1.3 ORDERING INFORMATION

The files which comprise MPG are integral components of the XXDP Multi-Media system. When all items of certain XXDP packages are ordered, MPG's binary files will be included. However, MPG listings and/or MPG binary files can be ordered separately from the Software Distribution Center (SDC). When binary files are ordered, they will be supplied on the single XXDP medium on which they reside for that package. For example, if the DECTAPE version of MPG is ordered, you would get only TCDP DECTAPE # 18.

The listing kit supplied for MPG consists of the program listings for all Device Routines, this Users' Manual, and the XXDP Users' Manual. The SDC Checklist numbers that are applicable to MPG are as follows:

Checklist	Description
ZJ197-RZ	Listing Kit
ZJ197-RC	Listing Kit and TCDP DECTAPE # 18
ZJ197-RD	Listing Kit and TMDP 9 Track MAGTAPE # 3
ZJ197-RE	Listing Kit and RKDP DECPACK # 2
ZJ197-RF	Listing Kit and TMDP 7 Track MAGTAPE # 3
ZJ197-RY	Listing Kit and RXDP Diskette # 6

A checklist entry for the RP04 was not included in the above list since XXDP is not available on that type of medium. The user must build the system on his own pack from one of the available media. In the case of the RP04 for MPG, all that is needed is the Listing Kit.

2.0 GENERAL INFORMATION

Since the principal users of MPG will be non-programming personnel, a special language has been developed to simplify program definition and entry. This language consists of English language format statements for data processing and I/O functions. Characteristics of this language include flexability, machine independence, and easy learning.

In addition to the program language, the user is furnished a set of commands that allow him to control MPG's operations and utilize the support functions provided with MPG. These support functions include program modification, storage and retrieval of defined programs, device assignment, etc.

The combination of the program language and the user commands provide the foundation for MPG's usefulness as a diagnostic tool. With ease of use being a major consideration in MPG's design, these components were oriented toward relieving the user of as many tedious operations as possible. For example, I/O operations may be defined at a high level where the user specifies only the function to be performed and MPG handles the device at the interface level with default values for memory address and byte count. The user may still have MPG handle the device interface but define a specific memory address and byte count for the I/O function. Or, the user may program the device at the interface level himself with complete freedom of device usage.

Another consideration was the elimination of associated programs or libraries that would tend to complicate MPG's usage. Therefore, MPG contains all routines required to support the user and his programs. Even though it performs a compile function, MPG does not require additional assembly programs or macro libraries during its operations. MPG is completely self contained in the stand alone version. In the operating system version it will utilize only the components required, such as I/O handlers.

In keeping with the ease of use philosophy, all commands and program statements are entered to MPG via the keyboard. For instance, the user, after determining the task to be programmed, makes a series of keyboard entries which consist of brief statements that define the task to MPG. Upon completing these entries, the user can then use the MPG commands to execute these statements (program) with a variety of options without changing the statements he entered.

3.0 MPG FEATURES

3.1 USER PROGRAMS

The user is required to specify the tasks and/or data operations that are to be performed by his program. These are entered in the form of program language statements which are defined in Appendix B. Prefixed to each statement will be a line number which is used for branching and for statement modifications. The user may enter his own values for the line numbers or enter a space and have MPG calculate and display the next highest line number which will have an increment factor of ten.

3.2 MULTIPLE USER PROGRAMS

MPG will support simultaneous execution of up to sixteen user programs. This capability is useful in troubleshooting interaction problems or for allowing concurrent troubleshooting of individual device problems by two or more technicians. Each program controls one device and is treated as a separate entity by MPG. Programs are distinguished by having a number (1, 2, etc) assigned to each. The format of various commands utilize these numbers and allow the user to perform operations on any program without affecting the other programs.

3.3 USER PROGRAM DATA AREAS

In the course of writing a program, the user may have need of memory locations which will be used as counters, storage areas, data input, etc. While MPG allows the user to define specific memory addresses if wanted, it also provides the user with predefined areas which are referenced by symbolic names. These areas are of two types with the first being a series of sixteen contiguous 1 word locations that are unique to each program. These locations, whose names are TM00 thru TM15, are useful for data meaningful only to that program. The second type of data areas is common to all programs and is useful when passing information between programs or for supplying or altering data from the command level. This second type has two formats which may be used as the need dictates. The first format is ten contiguous 1 word locations with the symbolic names of COM0 thru COM9. The second format is a buffer that consists of sixteen contiguous 16 byte areas (256 bytes total) with the names of BF00 thru BF15. Any COM or BF areas may be accessed by the user programs and also altered from the command level with the FILL command.

3.4 I/O AREAS

Since many I/O operations will not be dependent upon usage of specific memory locations, MPG provides two types of I/O areas which have predefined symbolic names. For each program there is a varisble length read area (RDIO) followed immediately in memory by a variable length write area (WRIO). When Entering or Fetching each program, the user is asked to specify the size of these areas. The preset values, which are 256 bytes, may be changed to any value from 0 to 32766 or the largest allowed by memory size. RDIO and WRIO are unique to each program and even though two or more programs may be using the same symbolic names, each program will be referencing different areas of memory. The other I/O area, which is known as FREE, extends from the end of MPG to the end of memory. Its size will be determined by the size of memory and the number and length of user programs. Important aspects of FREE are its use as an area for large I/O operations and as a common area where I/O data is passed between programs. Caution should be exercised when multiple programs are arbitrarily using the FREE area since the data read by one program could destroy another program's data.

Note: In the Memory Management version of MPG, the combined sizes of RDIO and ARIO cannot exceed 24,250 bytes. FREE will have a maximum size of 24,576 byte (12K words) and will exist at the lowest 12K words of available memory. If 12K words are not available, FREE will be at the largest area available.

The high level I/O statements utilize the program's I/O areas as default values. READ and WRITE statements will default to RDIO and WRIO, respectively, with byte counts of 256. When the FREE area is referenced as the memory location for these commands, a byte count must be specified.

It should be noted that the user is not required to use these areas and has complete freedom in defining specific memory addresses and byte counts for his I/O operations. Of course, this freedom is tempered by the necessity of discretion in the user's selection of memory areas. They should not be within MPG or other user programs. If within his program area, they must be within his RDIO and WRIO areas. They may be anywhere within Free Memory.

If the user is doing only read operations or only write operations and their length exceeds the size of RDIO but are less than the combined size of RDIO and WRIO, he may specify RDIO as the I/O area. Since WRIO follows RDIO in memory and is contiguous, the user has in effect an I/O area the size of RDIO plus WRIO. Care must be taken to never perform reads, whose length exceeds the size of RDIO plus WRIO, into RDIO or reads, whose length exceeds the size of WRIO, into WRIO. If this occurs, the program's source statements and possibly its object code will be destroyed.

3.5 OPERATION SWITCHES

Since there are several functions common to the type of programs written with MPG (printing of error information, stopping after an error, etc), these functions are provided as integral components of MPG. However, in certain situations some or all of these functions may not be desired. Therefore, MPG has for each program a set of software switches (bits) that are interrogated throughout execution of the user's program. These switches are preset to default values but can be modified by the user in three different ways. The first two methods are through commands. With one format of the OPSW command, the user can change the switch settings for a particular program while the other OPSW format provides for setting all resident programs to the same value with one command entry. The third method is by statements in the user's program. By using the symbolic name OPSW, the user program can contain statements that reference its OPSW and modify the switch values at any time during execution of the program.

As will be seen, certain bits in the OPSW word are not used by MPG. If the user wishes, he may utilize these bits for his own purposes. Since each OPSW word is modifiable from the command level and the user program can interrogate any bits in it, these bits may be used to control the execution of a program in a manner to be determined by the user.

The switch (bit) meanings of each program's one word OPSW have been defined in accordance with the standard bit usage listed in the Diagnostic Engineering Standards Document # 175-003-009-0 and are as follows:

OPERATION SWITCHES

Preset value is: 100000

BIT	VALUE	FUNCTION
15 =	0	Continue program execution on error.
	1	Cease program execution on error. (Preset)
14 =	0	Go to next device at end of program.
	1	Cycle program on current device.
13 =	0	Print on error.
	1	Do not print on error.
12 =	0	Not used.
11 =	0	Not used.
10 =	0	Stop when the device list is exhausted.
	1	Cycle the device list.

MPG Features (Cont'd)

9	=	0	Continue on current device on error.
		1	Go to next device on error. (Bit 9 is effective only when bit 15 = 0.)
8	=	0	Perform error checking.
		1	Do not perform error checking.
7	=	0	Do not perform special operations.
		1	Perform special operations as indicated in the device routine's documentation.
6	=	0	Not used.
5	=	0	Check for I/O timeout.
		1	Do not check for I/O timeout.
4	=	0	Automatically display device counts based upon the value of Bit 3.
		1	Do not display device counts.
3	=	0	Display device counts at the end of each program pass.
		1	Display the counts only after the final program pass.
2	=	0	Clear the device counts when the RUN command is issued and after each program pass except the final pass.
		1	Clear the counts only when the RUN command is issued.
1	=	0	Print all data non-compares detected with the VERIFY statement.
		1	Print only the first data non-compare detected with the VERIFY statement.
0	=	0	Print 'PROGRAM COMPLETED' message after the final pass of the program.
		1	Do not print 'PROGRAM COMPLETED' message.

MPG Features (Cont'd)

OPSW BIT LAYOUT

	14 CYCLE CURRENT DEVICE	11 NOT USED	8 DON'T ERROR CHECK	5 DON'T CHECK FOR I/O TIMEOUT	2 CLEAR COUNTS ONLY AT RUN COMMAND
	13 DON'T PRINT ON ERROR	10 CYCLE DEVICE LIST	7 DO SPECIAL OPERATION	4 DON'T DISPLAY DEVICE COUNTS	1 PRINT ONLY FIRST MISCOMP ON VERIFY
15 STOP ON ERROR	12 NOT USED	9 GO TO NEXT DEVICE ON ERROR	6 NOT USED	3 DISPLAY DEVICE COUNTS ONLY ON FINAL PASS	0 DON'T PRINT PROG COMPL'D MSG

Preset Value = 100000

4.0 DESCRIPTION OF OPERATION

MPG is provided on XXDP media and may be loaded by the XXDP Monitor or UPD1/2 programs. However, inherent with MPG is the ability to operate on media which is dedicated solely to MPG. In this case, MPG provides its own boot and loading functions. Refer to Section 8 for instructions concerning creation of MPG dedicated media. Also, refer to Section 6 for instructions pertaining to the loading of MPG by XXDP or from its own dedicated media and to Section 7 for an example of initiating MPG.

After being loaded, the MPG Stand Alone Executive's housekeeping section will initialize vector locations 0 through 276, determine the memory size, clear higher memory, and request the date. After receiving the date, the Executive will attempt to access Memory Management's MMR0 register. If found, a message will be issued that asks if Memory Management is to be used. If the reply is Y, <CR>, or <LF>, the name of the MPG file to be loaded will be set to TMMAnm.MPG. If MMR0 was not found or if the reply was N, the file name will be set to TMGAnm.MPG. Next, new console terminal constants are requested followed by the loading of MPG.

> Note: The remainder of Section 4 (Description of Operation) pertains to the non-Memory Management version of MPG. The Memory Management version is nearly identical in operation except for some additional capabilities. Refer to Section 9 for information concerning these capabilities.

Once loaded, MPG will assume control of the processor. Housekeeping will be performed followed by the typing of MPG's title line which includes its version number and the addresses of the restart points. (The restart point RST1 provides for complete clearing of internal MPG flagwords, tables, and user programs while RST2 performs the same functions except that any user programs are left intact.) After the title message is displayed, MPG will load the Valid Devices Table, which is file TVDAnm.MPG, from the load device and place it in MPG's program space. Next, a Memory Map, which lists the Free Memory available, will be displayed. Upon completion of the Memory Map display, MPG types its ENTER CMND message followed by an asterisk and is now ready to receive its first command.

The MPG user commands provide a variety of services such as operation control, user program definition and execution, program modification, program storage and retrieval, etc. These commands, which are listed in Appendix A, result in immediate action by MPG. After each command is entered, it is checked for validity and then the task it specifies is performed. Following this, MPG types an asterisk and waits for the next command to be entered. This continues until the user has entered all desired commands except RUN. At this point, a RUN command or a null keyboard entry (carriage return only) will cause MPG to proceed to user program execution. When all programs have completed or if no programs were available for execution, MPG will return to request additional commands.

4.1 ENTERING A PROGRAM

Upon recognizing the ENTER command, MPG will issue the ?ASGN DEV: message which requests an entry that is either the keyword NONE or the model number and if applicable, the decimal unit numbers (from 1 to 16 unit numbers specified individually and/or as strings) of the supported device for which the program is to be written. Following this message and reply, two more messages, which display the preset sizes of RDIO and WRIO and request new values, are issued. For each message either a null keyboard entry, which indicates that the preset values are to be used, or a decimal number, which specifies the desired size of each area in bytes, may be entered.

If the reply to the ?ASGN DEV: message was NONE, a small dummy device routine, which occupies minimum memory space, will be generated followed by MPG proceeding to program statement processing. If a model number was entered, the device routine for the model specified will be loaded into memory with its filename being displayed on the console terminal. Acting upon information contained within the device routine, MPG will issue the first of three messages. These messages, each of which require a reply, will identify and display the preset values for the device's device register address, interrupt vector address, and bus request priority in that order. After each is displayed, a read will be issued to the console keyboard. If data is entered, it will be used to replace the existing data. If a null entry (carriage return or line feed only), the original data will be left intact. The first two messages process one word of data while the third has the capability of displaying and receiving two words (separated by a comma). This feature is required for communication devices which require different bus priorities for read and write. If the second bus priority word (write's) is zero, only the first word (read's or the only priority) will be displayed.

Note: If the reply to the ASGN DEV message is terminated by a line feed (instead of a carriage return), the subsequent messages and replies for RDIO size, WRIO size, device register address, interrupt vector address, and bus priority will be bypassed and results in their preset values being utilized. Also, if the reply to any one of the five messages just mentioned is terminated by a line feed, any remaining messages will be bypassed.

When device routine processing is completed, MPG allocates room for the RDIO and WRIO areas, issues a message indicating that it is ready to receive program statements and then types a "greater than' (>) sign. At this time the user begins entering these statements. (Refer to the sample programs included with each Device Routine description in Appendix D.) Since all program statements must have a line number as their first component, this must be the first entry for the statement. The user may enter a 1 to 4 character decimal number followed by a tab (or space) and MPG will use that as the statement's line number. Or, he may enter a space and MPG will generate a new line number which has the value of the current line number + 10. After the line number is defined (by either method), the program statement is entered on the same line and followed by a carriage return. As each entry is received, it is validated and then stored in memory in a condensed format. This continues without any other actions until the END and DONE statements have been processed. Following the DONE, MPG will scan the stored program statements, make further validity checks, and then compile a section of machine code based upon the functions indicated by these statements.

Next, MPG issues the "PGMS COMPILED / MEM REFORMATTED" message and then types an asterik to indicate that it is ready for the next command. If the user desires, he may enter the MM command and have displayed on the console terminal a memory map, which is a series of messages listing the starting and ending addresses for each user program and for Free Memory. Included for each program is its name if it was fetched or the default name of ORPHAN if it has been entered without a name being specified. Also included is the model code of the device assigned to the program. Whether the memory map is displayed or not, the user may now execute the program just defined, list the program statements and the octal machine code generated for the program, define another program in a different program slot, or initiate any other MPG command.

When the ENTER command or the FETCH command is issued, MPG makes no checks as to whether a program already exists for that slot. If one does exist, it will be destroyed by the new program.

4.2 EXECUTION OF A PROGRAM

Once the user has defined one or more programs, he may initiate their execution through use of the RUN command. The format of this command provides for activating a single program or multiple programs with each program being started at its beginning or, optionally, at a specific statement on the first unit. After entering all desired commands, the entry of the RUN command will initiate program execution.

Note: If desiring to continue execution of a previously stopped program without initiating any other programs, the issuing of the Continue command (CONT p) followed by either a null keyboard entry or the RUN command without a program number will initiate program execution.

Description of Operation (Cont'd)

Upon recognizing any of these entries, MPG will begin scanning the program slots utilizing a polling loop technique. Starting with slot # 1, MPG will check each slot and ascertain if a program is present. If not, it proceeds to the next slot. If a program is present, an internal program flagword will be tested to determine if the program has been specified for execution and if it has, whether it has been stopped by the user or by an error occurrence. If neither of the stop conditions are present and if the program is not waiting for an I/O termination, then control will be transferred to the program's object code at its current address. The program will retain control (executing) until one of the following conditions causes it to return control to MPG:

- The Device Routine has initiated a high level I/O command and is waiting for device termination.

- The Device Routine attempted a high level I/O command but the control unit was already busy.

- The Device Routine detected an error during a high level I/O operation.

- The user program contains a Control Release (LETGO) statement which is being performed.

- The user program has completed.

Upon receiving control again, MPG performs the necessary functions for suspending that program and then proceeds to test the next program slot. After processing the last program slot, the scanning will resume at program slot # 1. If all programs are waiting for I/O termination, MPG will continue looping through the program slots looking for a program that is ready to continue execution. When one of the I/O functions terminates, the next time that its associated program slot is scanned, program execution will be resumed. When a device is already busy or when a control release (LETGO) is issued, program execution will be resumed on the next program slot scan. This allows all other programs at least one chance at execution before these programs regain control.

Program slot scanning continues until all programs have either terminated or been stopped by the user or because of errors. If multiple units were specified for the program's device, the program will be run on each unit before terminating. When processing a program's termination, MPG will issue a "PROGRAM p COMPLETED" message for that program. Upon determining that all program slots are inactive, MPG will return to the command section and request new commands.

An automatic feature provided by MPG at program termination is the display of the applicable statistic counts accumulated by the the program's device routine. These counts, which include the commands issued, bytes read and/or written, and errors, are displayed at the end of each program pass or when continuing to the next unit upon detecting an error on the current unit. The automatic display of these counts and when they will be displayed are controlled by bits in the program's OPSW word. Another method to display these counts at anytime is with the REPORT command. The COUNTS operand will display the same information that is automatically displayed at the end of a program pass.

As mentioned previously, when a program returns control to MPG its execution will be suspended. This consists of storing the contents of registers R0 through R5 and the PC in a reserved area within the program's space. Also stored in this space are any words left on the stack by the user program. When MPG determines that a program is to be given control again, it will reload the stack with any saved words and restore registers R0 through R5 before resuming execution of the program. It should be noted that the maximum number of words that can be saved off of the stack is thirty. If a user program returns control with more than 30 words on the stack, it will be aborted with an error condition.

Whenever a program is specified for execution with the RUN command or when it is repeated for the next device in the device list, all storage areas for the registers, stack words, temporary work areas (TM00 through TM15), the device routine words (BLK, CYL, SECT, etc.), and the applicable device routine counters are cleared to zeros or set to their preset values before starting the program. Clearing of the device routine counters can be controlled by the user with bit 2 in the program's OPSW. The user may specify that these counts are to be reset only when the RUN command is issued.

4.3 INTERRUPTING USER PROGRAMS

While user programs are executing, the user may enter commands by first gaining MPG's attention with an CONTROL 'C' entry from the keyboard. Upon recognizing this entry, which will be honored only if user programs are executing, MPG will suspend execution of the user programs and abort any outstanding read or write issued by user programs to the console terminal. As an added debugging aid, MPG will display the values of the PC and SP at the time of interrupt and whether a user program (identified by its number) or MPG had control. Following this message, MPG types its ENTER CMND message and the asterisk which is preceded by an I to indicate that MPG is at the interrupt command level. At this time the user may enter any command with the exception of those that could change the position in memory of the currently executing programs or the current device number for a user program. The commands which will not be honored are:

ENTER	FM	
/FETCH	ASSIGN	(directed to a program)
MODIFY	SHIFT	(Memory Mgmnt only)
DELETE	UBMAP	(Memory Mgmnt only)

After all desired commands have been entered (changing of OPSW values, stopping a program, etc) and the RUN command or the null entry made, execution of the user programs will be resumed at their point of interruption.

If the user wishes to use one of the restricted commands, he must issue the STOP command for all executing programs and then let program execution resume. Return to the command entry section will occur immediately if a program did not have control at the time of interruption or if one did, when it releases control.

One command has been included specifically for use when interrupting user programs. The KILL command provides the user with the ability to either abort a program that has hung in a loop which does not allow return to MPG or to force an immediate termination of a program. Any program stopped with the KILL command can be restarted only with the RUN command.

4.4 USER PROGRAM MODIFICATIONS

After a user program has been entered, the need may arise for replacing or deleting existing statements or for the insertion of new ones. Rather than require the user to re-enter his entire program with updates applied, MPG contains a source correction facility which provides the functions necessary for this purpose.

Once the user has defined a program, he may alter its statements through use of the MODIFY command. When this command, which can specify only one program, is entered, MPG sets itself to accept program statements from the keyboard. Each statement entered must contain a line number (statements have a 1 to 4 digit line number as their first component) and it is this that identifies the type of update required. A line number followed by the word DELETE will cause the deletion of the matching numbered statement in the program. If a program statement is entered and its line number matches an existing statement, the original statement will be replaced with the new statement. If there is not a matching line number, the new statement will be inserted in correct numerical sequence. After all corrections have been entered, the user issues the DONE command which causes MPG to re-compile the modified program. Following this, MPG returns to request another command.

If the user desires a listing of all statements in his updated program, he may issue the DISPLAY command which will display all statements and optionally, the octal machine code generated by the compiler, on the console terminal or alternate print device.

Another form of program modification is accomplished with the ASSIGN command. With this command the user can change the unit numbers assigned to an existing program or even change the device constants for the program. If the user wishes to change only the device constants (device register address, interrupt vector address, or bus priority), all that is needed is the ASSIGN command issued with the original model and unit numbers. The existing values for the addresses and priority will be displayed and then changes, if any, can be entered.

4.5 SAVING AND FETCHING OF USER PROGRAMS

Since some user programs may be rather large and under various conditions may need to be keyed-in a number of times, MPG provides the ability to save and then later retrieve the user program's statements on either the system load device or paper tape. This capability is accomplished through use of the ASSIGN, /SAVE, and /FETCH commands.

The ASSIGN command has a format which allows the user to alter the devices used by the /SAVE and /FETCH commands. Normally, the system load device is the preset device for these commands. Refer to section 5.2 for more details.

The /SAVE command identifies the previously entered program whose statements are to be saved and the name that they will be saved under. After receiving this information, MPG will extract each program statement stored in the condensed format, expand it to full format, and then write it to the SAVE device in the correct blocked format. This continues until all statements have been processed. MPG then returns for the next command entry.

The /FETCH command identifies the stored program by name and the program slot into which it will be loaded. Processing of this command is very similar to the ENTER command described in section 3.1. The one major difference is that the FETCH device is searched for the named program and that the program statements are read from it instead of the keyboard. Other functions such as device assignment, statement validation, and program compilation are performed in the same manner.

Additional commands are provided to aid the user in the saving and fetching of his programs. The /LIST command provides a directory of the programs resident on the FETCH device while the /DELETE command allows the user to delete programs from the SAVE device. The /ZERO command will initialize a new medium on the SAVE device.

5.0 DEVICE SUPPORT INFORMATION

5.1 DEVICES REQUIRED FOR MPG

The devices required by the stand alone version of MPG are necessary for its loading and communication with the user. Any other devices required would be the units to be tested. The following are the minimum requirements:

- One PDP-11 series central processor (LSI-11 through 11/70) with a minimum of 16K words of memory.
- Any one of the following load media devices:
  - TU56 DECtape transport and associated TC11 control unit.
  - RK05 DECPACK disk drive and associated RK11 control unit.
  - TU10 MAGTAPE transport and associated TM11 control unit.
  - TU16 MAGTAPE transport and associated TM02 control unit.
  - RX01 Floppy Disk drive and associated RX11 control unit.
  - RP04 Disk drive and associated RH11 control unit.
- One console terminal (LA30, LA36, LT33, or VT05) and its associated control unit.
- Device to be tested

5.2 OPTIONAL DEVICES FOR MPG

The devices listed in section 5.1 are required for MPG and its Executive. There are, however, additional devices which may optionally be used for certain MPG functions. The support for these devices is completely independent of the high level I/O device routines.

The LIST device, which is initially assigned to the console terminal, can be assigned to a printer (LP11, LS11, or LV11) through use of the ASSIGN comand. When this option is used, all print information generated by the user programs, high level I/O device routines, program source displays (DISPLAY command), and the directory display (/LIST command) will be directed to the printer. If desiring to resume use of the console terminal as the LIST device, the assignment of KYBD, KBOO, or NONE as the LIST device will effect this change.

The SAVE and FETCH devices, which are initially assigned to the MPG load device, may be redirected to other devices through use of the ASSIGN command. Either or both the SAVE and the FETCH device may be assigned to another unit on the load device. This is accomplished by entering the load device's model number followed by the desired unit number with the ASSIGN command. The FETCH device can be assigned to the paper tape reader (PC11 or PR11) and the SAVE device to the paper tape punch (PC11). It should be noted that when using paper tape, the use of certain MPG commands (/LIST, /DELETE, /BOOT, and /ZERO) will result in errors being reported. Also, even though not used, the one to six character program name used in the /SAVE and /FETCH commands must still be supplied when using paper tape. After assigning an alternate SAVE or FETCH device, reassignment to the load device can be accomplished by assigning either LOAD, NONE, or its model name and unit number to the desired device.

5.3 MPG DEVICE USAGE DETAILS

Sections 5.1 and 5.2 list the devices that can be used by MPG for its own functions. This section provides the details about MPG usage and also general information about the devices.

Console Terminal

The console terminal driver is located in the Executive and operates in an interrupt driven mode. The preset values for its device constants are:

```
Read:   Device Register Address = 177560
        Interrupt Vector Address = 60
        Interrupt Proc. Status Word = 200  (BR 4)

Write:  Device Register Address = 177564
        Interrupt Vector Address = 64
        Interrupt Proc. Status Word = 200  (BR 4)
```

During its one time housekeeping, the Executive samples the console terminal by testing the write status register. If not there or if READY is not set, the Executive will halt. (This is the only halt in the Executive that is not preceded by an error message.) When stopping at this halt, register RO contains an address which points at the word containing the read device register address. The five words following this word are in the same order as listed above. At this time, the user may change any or all of the six words and thereby direct MPG to an operable console terminal. After the changes have been made, depressing CONTINUE will cause MPG to repeat the console terminal test with the new values.

During execution, the MPG console terminal driver recognizes certain control characters and acts upon them as follows:

RUBOUT The last character entered on the current read will be deleted. The character or characters deleted are typed and enclosed in backslash (\) characters. If all entered characters have already been deleted, a carriage return/line feed will be typed.

CONTROL/U Types the "↑U" characters, deletes all data entered on the current read, and then types a CR/LF. Effectively re-initializes the current read.

CONTROL/C This entry is honored only if user programs are running. When processed, it causes any read or write to the console terminal to be aborted, the "↑C" characters to be typed, and entry into MPG's User Interrupt Mode to be made.

CONTROL/O Causes all further writes to the console terminal to be suppressed. This action remains in effect until MPG returns to request a new command or until an error is detected by MPG components other than the Device Routines. If used while user programs are running, all messages issued to the console terminal are suppressed and will remain suppressed until all user programs have either been stopped or terminated. If desiring to resume typing during user program execution, a CONTROL/C entry will reset the internal software flag used for suppressing typeouts. During command entry there is no way to resume typing until the current command completes.

CONTROL/S This entry instructs MPG to suspend all further writes to the console terminal after the current one completes. The console terminal is put in a simulated busy condition and remains so until any character, other than a CONTROL/S, is entered on the keyboard. This character is discarded and typing will resume at the point of interruption with no loss of data. This feature can be used while user programs are running and will have the same effect on their output.

For console terminal writes, bufferring is provided for up to twenty bytes. There is no bufferring for read data.

PRINTER

Any of three printers (LP11, LS11, or LV11) can optionally be assigned as the MPG LIST device and will be used for certain display functions. The driver for the printer is interrupt driven and operates with the following preset constants:

Device Register Address = 177514
Interrupt Vector Address = 200
Bus Request Priority = 4

The above constants can be changed after using the ASSIGN command to assign the printer as the LIST device. The current values will be displayed after each ASSIGN command and can be changed if desired.

PAPER TAPE AND LOAD MEDIA HANDLERS

The handlers contained in MPG perform the services required for program and file maintenance. This includes the loading of MPG files, the saving and fetching of user programs, directory displays, etc. Currently, MPG supports seven handlers which are for the following devices:

PC11/PR11	Paper Tape
RK11	DECpack Disk
TC11	DECtape
TM02	MAGTAPE
TM11	MAGTAPE
RX11	Floppy Disk
RP04	Disk

Due to size requirements, the MPG Executive will contain the Paper Tape handler and only one of the other handlers which will be its load media. This results in six versions of the Executive with each supporting a different load medium. Refer to Section 7 for further details concerning the various Executives and instructions for generating MPG load media.

These handlers are modified versions of the XXDP Program handlers. They are not interrupt driven and require only device register addresses for their operation. Due to their original XXDP implementation, the device register addresses are fixed and cannot be changed. The following are the addresses for each device:

177550 = PC11/PR11 Paper Tape Reader
177554 = PC11 Paper Tape Punch
177400 = RK11 DECpack Disk
177340 = TC11 DECtape
172440 = TM02 MAGTAPE
172520 = TM11 MAGTAPE

Device Support Information (Cont'd)

177170 = RX11 Floppy Disk
176700 = RP04 Disk

5.4 DEVICES SUPPORTED FOR USER PROGRAMS

The title of this section is somewhat misleading. Devices considered as supported under MPG are those for which high level I/O functions (Read, Write, Seek, etc.) are available. Aowever, due to the flexability and power of the low level language, a knowledgeable user may program at the device interface level and therefore, in a lesser sense of the word, has support for practically any PDP-11 device. So, even though the device you wish to exercise may not be in the following list, do not lose heart. You will just have to work a little harder than if the high level I/O functions were available for the desired device. Even so, it is still easier to program the device interface with MPG instead of composing the necessary instructions in octal machine language and writing them to memory.

Supported Devices:

- DH11 16 Line Programmable Asynchronous Multiplexor
- DJ11 16 Line Asynchronous Multiplexer
- DL11 Single Line Interface
- DQ11 NPR Synchronous Line Control
- DU11 Single Line Synchronous Interface
- LP11/LS11/LV11 Printers
- PC11/PR11 Paper Tape
- RK06 Disk
- RK11 Cartridge Disk
- RP02/RP03 Disks
- RP04/RP05/RP06 Disks
- RS03/RS04 Disks
- TC11 DEC Tape
- TM11 Magnetic Tape

5.5 DEVICE SUPPORT SUMMARY TABLE

The following table reflects the type of support provided by MPG for each device. Note that some devices are supported for MPG functions but not for high level I/O operations. These devices and the other devices that are listed but which do not have high level I/O support indicated, all have small basic device routines which contain the symbolic names of the device registers. Also, devices that are supported as SAVE and FETCH devices, other than paper tape, are applicable only if they are the MPG load device.

DEVICE MODEL NAME	LOAD DEVICE	HIGH LEVEL I/O	LIST DEVICE	FETCH DEVICE	SAVE DEVICE
CD11					
CM11					
CR11					
DC11					
DH11		X			
DJ11		X			
DL11		X			
DN11					
DQ11		X			
DU11		X			
KW11					
LP11		X	X		
LS11		X	X		
LV11		X	X		
PC11		X		X	X
PR11		X		X	
RC11					
RF11					
RK05	X	X		X	X
RK06		X			
RK11	X	X		X	X
RP02		X			
RP03		X			
RP04	X	X		X	X
RP05	X	X		X	X
RP06	X	X		X	X
RS03		X			
RS04		X			
RX01	X			X	X
RX11	X			X	X
TA11					
TC11	X	X		X	X
TM02	X			X	X
TM11	X	X		X	X
TU10	X	X		X	X
TU16	X			X	X
TU56	X	X		X	X

6.0 MPG LOADING INSTRUCTIONS

As stated previously, MPG is supplied on XXDP media but has the ability to operate from media which is dedicated to it. Section 6.1 contains the information concerning loading of MPG through XXDP while sections 6.2 through 6.8 pertain to MPG dedicated media.

The boot procedures contained in sections 6.2 through 6.8 may be used to load either XXDP or dedicated MPG media. Conversely, the boot procedures listed in the XXDP User's Manual may also be used for loading both types of media.

Upon completing any of the following load procedures, MPG will perform its housekeeping functions, request the date, issue its intialization messages, and then type an asterisk to signify its readiness for the first user command.

6.1 LOADING VIA XXDP

After loading the XXDP Monitor by using the instructions in either the XXDP's User's Manual or in sections 6.2 through 6.8, the Monitor will type its title line and usage information and then wait for a keyboard entry. At this point the user may directly load MPG by typing one of the following commands followed by a carriage return:

```
R TCMPG       (TC11 version)
R RKMPG       (RK11 version)
R TMMPG       (TM11/TU10 version)
R THMPG       (TM02/TU16 version)
R RXMPG       (RX11 version)
R RBMPG       (RP04 version)
```

This entry results in the specified version of MPG being loaded and control being transferred directly to it.

The difference in the versions listed above consists of the type of device that will be used as the primary SAVE/FETCH device. Paper Tape is the secondary device in all versions.

An alternate method of loading MPG is through XXDP's UPD1 or UPD2 programs. Instead of loading MPG directly as just described, enter either "R UPD1" or "R UPD2" as the reply to the Monitor. The selected UPD program will be loaded and will request the date. After making this reply, an asterisk is typed and the program waits for a command entry. At this point the user must enter two commands in order to initiate MPG. The first command specifies that a program is to be loaded, the device from which it will be loaded, and the program's name. The valid program names are those listed previously but with .BIN extensions. The second command initiates execution of the loaded program. An example of loading and initiating the RK11 version of MPG from an XXDP RK11 pack is as follows:

```
LOAD DK0:RKMPG.BIN
START
```

The only difference in MPG's operation when loaded under XXDP is that after MPG requests the date, it will request the unit number of the drive that is to be assigned as the initial SAVE/FETCH/LOAD device. When loading a dedicated version of MPG, it is always set to the load device. In the prior example, a reply of 0 will set the SAVE/FETCH assignment to the same RK11 unit as which MPG happened to be loaded from.

It should be noted that when MPG assumes control, it will destroy the XXDP Monitor and any UPD program in core. In order to resume XXDP operation, the load medium must be rebooted or MPG's /BOOT command used.

6.2 DEDICATED MEDIA LOADING INSTRUCTIONS

The following sections contain the information and instructions neccessary for the loading of MPG. Each section will apply to a specific type of MPG input medium. As additional media are supported, their instructions will be added.

6.3 TC11 DECTAPE INPUT

The following are the initial steps for loading from DECtape:

1. Mount the MPG tape on unit 0.

2. Set the REMOTE/OFF/LOCAL switch to REMOTE.

3. If hardware bootstrap loaders BM792-YB, MR11-DB, or BM873-YA/YB are present, go to section 6.3.1.

4. If not present, go to section 6.3.2.

6.3.1 TC11 HARDWARE BOOTSTRAP

1. Set the Switch Register to:

   773100 for the BM792-YB or,
   773120 for the MR11-DB or,
   773030 for the BM873-YA or,
   773070 for the BM873-YB.

2. Press the LOAD ADDR switch.

MPG Loading Instructions (Cont'd)

3. If the BM792-YB, set the Switch Register to 777344 (word count register for DECtape).

4. Press the START switch.

6.3.2 TC11 SOFTWARE BOOTSTRAP

1. Set the Switch Register to 010000 and press the LOAD ADDR switch.

2. Set the Switch Register to the first of the following values and press the DEPosit switch. Continue depositing until all values are entered.

```
10000=   12700        MOV    #TCWC,R0
        177344
         12710        MOV    #-256.,TCWC
        177400
10010=   12740        MOV    #4002,TCCM
          4002
          5710    A:  TST    TCCM
        100376        BPL    A
10020=   12710        MOV    #3,TCCM
             3
        105710    B:  TSTB   TCCM
        100376        BPL    B
10030=   12710        MOV    #5,TCCM
             5
        105710    C:  TSTB   TCCM
        100376        BPL    C
10040=    5007        CLR    PC
```

3. Set the Switch Register to 010000 and press the LOAD ADDR switch.

4. Press the START switch.

6.4 RK11 DECPACK DISK INPUT

The following are the initial steps for loading from disk:

1) Mount the MPG DECpack on drive 0.

2) Load the drive, write lock it, and then wait until the drive is ready.

3) If hardware bootstrap loaders BM792-YB, MR11-DB, or BM873-YA/YB are present, go to section 6.4.1.

MPG Loading Instructions (Cont'd)

4) If not present, go to section 6.4.2.

6.4.1 RK11 HARDWARE BOOTSTRAP

1) Set the Switch Register to:

   773100 for the BM792-YB or,
   773110 for the MR11-DB or,
   773010 for the BM873-YA or,
   773030 for the BM873-YB.

2) Press the LOAD ADR switch.

3) If the BM792-YB, set the Switch Register to 777406.

4) Press the START switch.

6.4.2 RK11 SOFTWARE BOOTSTRAP

1) Set the Switch Register to 010000 and press the LOAD ADR switch.

2) Set the Switch Register to the first of the following values and press the DEPosit switch. Continue depositing until all values are entered:

```
10000=   12700        MOV     #RKWC,R0
        177406
         12710        MOV     #-256.,RKWC
        177400
10010=   12740        MOV     #5,RKCS
             5
        105710    A:  TSTB    RKCS
        100376        BPL     A
10020=    5007        CLR     PC
```

3) Set the Switch Register to 010000 and press the LOAD ADR switch.

4) Press the START switch.

6.5 TM11 MAGTAPE INPUT

The following are the initial steps for loading from magtape:

1) Mount the MPG tape on drive 0 and make it ready by rewinding it to BOT and placing it ON-LINE.

2) If hardware bootstrap loaders MR11-DB or BM873-YA/YB are present, go to section 6.5.1.

3) If not, go to section 6.5.2.

6.5.1 TM11 HARDWARE BOOTSTRAP

1) Set the Switch Register to:

773136 for the MR11-DB or,
773050 for the BM873-YA or,
773110 for the BM873-YB.

2) Press the LOAD ADR switch.

3) Press the START switch.

6.5.2 TM11 SOFTWARE BOOTSTRAP

1) Set the Switch Register to 010000 and press the LOAD ADR switch.

2) Set the Switch Register to the first of the following values and press the DEPosit switch. Continue depositing until all values are entered:

```
10000=   12700    A: MOV     #MTBRC,R0
        172524
          5310       DEC     MTBRC
         12740       MOV     #60011,MTC
10010=   60011
        105710    B: TSTB    MTC
        100376       BPL     B
          5710       TST     MTC
10020=  100767       BMI     A
         12710       MOV     #60003,MTC
         60003
        105710    C: TSTB    MTC
10030=  100376       BPL     C
          5710       TST     MTC
        100777    D: BMI     D
          5007       CLR     PC
```

MPG Loading Instructions (Cont'd)

3) Set the Switch Register to 010000 and press the LOAD ADR switch.

4) Press the START switch.

6.6 TM02 MAGTAPE INPUT

The following are the initial steps for loading from magtape:

1) Mount the MPG tape on drive 0 and make it ready by rewinding it to BOT and placing it ON-LINE.

2) If hardware bootstrap loader BM873-YB is present, go to section 6.6.1.

3) If not, go to section 6.6.2.

6.6.1 TM02 HARDWARE BOOTSTRAP

1) Set the Switch Register to 773150.

2) Press the LOAD ADR switch.

3) Press the START switch.

6.6.2 TM02 SOFTWARE BOOTSTRAP

1) Set the Switch Register to 010000 and press the LOAD ADR switch.

2) Set the Switch Register to the first of the following values and press the DEPosit switch. Continue depositing until all values are entered:

```
10000=    12700         MOV     #MTCS1,R0
         172440
          12760         MOV     #1300,MTTC
           1300
10010=       32
          12760         MOV     #-1,MTFC
         177777
              6
10020=    12710         MOV     #31,MTCS1
             31
           5760      A: TST     MTDS
             12
10030=   100375         BPL     A
          12760         MOV     #-256.,MTWC
         177400
```

MPG Loading Instructions (Cont'd)

```
                   2
10040=   12710         MOV       #71,MTCS1
                  71
        105710      B: TSTB      MTCS1
        100376         BPL       B
10050=    5710         TST       MTCS1
        100777      C: BMI       C
          5007         CLR       PC
```

3) Set the Switch Register to 010000 and press the LOAD ADR switch.

4) Press the START switch.

6.7 RX11 FLOPPY DISK INPUT

The following are the initial steps for loading from the Floppy Disk:

1) Insert the MPG Diskette in drive 0.

2) Insure that the drive is ready.

3) If hardware bootstrap loader BM792-YL is present, go to section 6.7.1.

4) If not present, go to section 6.7.2.

6.7.1 RX11 HARDWARE BOOTSTRAP

1) Set the Switch Register to 173400.

2) Press the LOAD ADR switch.

3) Press the START switch.

6.7.2 RX11 SOFTWARE BOOTSTRAP

1) Set the Switch Register to 010000 and press the LOAD ADR switch.

2) Set the Switch Register to the first of the following values and press the DEPosit switch. Continue depositing until all values are entered:

MPG Loading Instructions (Cont'd)

```
10000=    5000        CLR     R0
         12701        MOV     #RXCS,R1
        177170
        105711    A:  TSTB    RXCS
10010=    1776        BEQ     A
         12711        MOV     #3,RXCS
             3
          5711    B:  TST     RXCS
10020=    1776        BEQ     B
        100406        BMI     D
        105711        TSTB    RXCS
        100003        BPL     C
10030=  116120        MOVB    RXDB,(R0)+
             2
           770        BR      B
          5007    C:  CLR     PC
10040=       0    D:  HALT
           776        BR      D
```

3) Set the Switch Register to 010000 and press the LOAD ADR switch.

4) Press the START switch.

6.8 RP04 DISK INPUT

The following are the initial steps for loading from disk:

1) Mount the MPG pack on drive 0.

2) Load the drive, write lock it, and then wait until the drive is ready.

3) If hardware bootstrap loader BM873-YB is present go to section 6.8.1.

4) If not present, go to section 6.8.2.

6.8.1 RP04 HARDWARE BOOTSTRAP

1) Set the Switch Register to 773320.

2) Press the LOAD ADR switch.

3) Press the START switch.

6.8.2 RP04 SOFTWARE BOOTSTRAP

1) Set the Switch Register to 010000 and press the LOAD ADR switch.

MPG Loading Instructions (Cont'd)

2) Set the Switch Register to the first of the following values and press the DEPosit switch. Continue depositing until all values are entered:

```
10000=   12700          MOV     #RPCS1,RO
        176700
         12710          MOV     #23,RPCS1
            23
10010=    5060          CLR     RPDC
            34
          5060          CLR     RPDA
             6
10020=   12760          MOV     #-256.,RPWC
        177400
             2
         12710          MOV     #71,RPCS1
10030=      71
        105710     A:   TSTB    RPCS1
        100376          BPL     A
          5007          CLR     PC
```

3) Set the Switch Register to 010000 and press the LOAD ADR switch.

4) Press the START switch.

7.0 INITIATING MPG

When the Executive and MPG are initially loaded, they will display information pertinent to MPG's operation. They will also request information from the user and allow him to specify if MPG is to utilize certain optional hardware features if they are available on the system. The following example lists the messages that can occur during this initialization process. Included after each message is a brief explanation of the message and its optional responses, if any. Note that any response must be terminated by either a Carriage Return <CR> or a Line Feed <LF> and that default values are supported when either of these are the only entry made. For this example, the loading of MPG's RK11 Executive on an 11/70 system has been chosen and all user responses have been underlined. Also, even though they may be the same as the default values, responses are shown for all messages requiring them.

```
DTE1A-n  RKMPG - RK11 EXECUTIVE FOR MPG
```

This is the Executive's title message and lists its MAINDEC number, revision number, and type of load device. This message will be different for each version of the Executive.

```
?DATE  (DD-MMM-YY)

*10-FEB-76 <CR>
 --------------
```

The current date in standard DEC format is requested by this message. A valid date must be entered as the response.

```
?LOAD/SAVE/FETCH DEV UNIT # / 0 <CR>
                              ------
```

This is an optional message and will occur only if the Executive has been loaded by either an XXDP Monitor or XXDP's UPD1/2 programs. Since another program has loaded the Executive, it has no way of determining the unit number of the load device. Therefore, it requests that the unit number be entered. This number will always be retained for the LOAD device but the SAVE and FETCH devices may be reassigned later with the ASSIGN command. The default unit number for this response is 0. This message will not occur if the Executive is the program that is booted. In this case, the Executive retrieves the current unit number from the device registers and stores it as the unit number for all three devices.

```
?USE MEM MGMNT (Y/N) / Y <CR>
                       ------
```

This message will occur only if the Executive has detected the presence of the Memory Management feature. The reply entered will determine which version of the MPG program is to be loaded. If Memory Management is not present or if N is entered, the non-Memory Management version of MPG (DTMGAn) will be loaded. The entry of Y or a default entry will result in DTMMAn being loaded.

CONS TERMINAL CONSTANTS:

This is a header message which precedes a series of four messages that allow modification of the preset values for constants used with the console terminal. If a <LF> terminator was entered on the reply to any of the three preceding messages, this message and the following four messages will not be issued and their default values will be used.

```
?FILL AFTER:  012 / 012 <CR>
                    --------
```

Requests the octal value for the byte after which fill characters are to be transmitted. The default character is a Line Feed (012). A <LF> terminator on any one of these four messages will cause the Executive to bypass the remaining messages.

```
?FILL WITH:   000 / 0 <CR>
                    ------
```

Requests the octal value of the character to be issued when transmitting fill characters. The default character is a null (octal 000 byte).

```
?FILL COUNT:  002 / 2 <CR>
                    ------
```

Requests an octal count for the number of fill characters that are to be transmitted. Defaults to two characters.

Initiating MPG (Cont'd)

```
?CONVERT LOWER CASE TO UPPER (Y/N) / Y <CR>
                                     ------
```

This message asks the user if he wishes the Executive to automatically convert console keyboard data from lower case to upper case. This would be applicable if, for example, the console terminal is an LA36. If the reply is N, the data will not be converted and the user will have to use the Shift key when making all MPG entries. Default value is Y.

At this point the Executive will load the appropriate version of MPG and turn control over to it. The remaining messages listed are issued by the MPG program. For MPG, a <CR> or a <LF> may be used to terminate any responses entered. However, a <LF> terminator will not cause bypassing of any subsequent messages. Also, default values are supported for all responses.

```
DTMMA-n   M.P.G.   RST1:  xxxxxx;  RST2:  xxxxxx
```

This is MPG's title message and lists its MAINDEC number and revision number. Also included are the octal addresses of the two restart points available within MPG. Refer to Section 4.0 for details concerning the two restart points.

```
MPG BASE ADDRESS = xxxxxx
```

The starting memory address of the MPG program, which immediately follows the Executive in memory, is displayed in this message.

```
?USE UNIBUS MAP (Y/N) / Y <CR>
                        ------
```

This message will occur only if the Memory Management version of MPG is being executed and if it has detected the presence of the Unibus Map feature. The reponse to this message, which has a default value of Y, will determine if MPG is to utilize the Unibus Map.

```
?USE 22 BIT ADRS (Y/N) / Y <CR>
                         ------
```

This message will occur only if the Memory Management version of MPG is being executed and if it has determined that the processor has 22 bit addressing capabilities. The Y response or the default response will instruct MPG to use 22 bit addressing. Otherwise, it will use 18 bit addressing.

Initiating MPG (Cont'd)

```
* MEM MAP *

ABS ADRS:  FREE MEM = xxxxxxxx TO xxxxxxxx (MIDL) = xxxxxxxx)

END OF MEM = xxxxxxxx
```

The above messages are produced by the MM (Memory Map) command which is performed as the last step in initialization. The starting and ending addresses of the FREE memory area and the address of its mid point (MIDL) are displayed. The third message displays the end of actual memory. This third message does not occur in the non-Memory Management version. For that version, the end address of FREE is the end of actual memory.

```
*1st command
 ----------
```

The asterisk indicates that MPG has completed initialization and is now ready to receive its first command.

8.0 GENERATING MPG MEDIA

MPG supports operation when loaded from various media as specified in section 6. This support includes saving and fetching of user programs on the load device or other units on its control unit or on paper tape. Also included is the initial loading of MPG and the loading of the Device Routines for user programs.

The files required by MPG and any user programs are stored in absolute format which is compatible with the file structure utilized by the XXDP program. Since this compatibility exists and since XXDP provides file maintenance services, XXDP is used to create and maintain MPG media. This use of XXDP significantly reduces the size of MPG by not requiring additional commands and routines for functions used exclusively in MPG media maintenance. It should be noted that the user still has adequate maintenance capabilites for his programs through use of the MPG slash (/) commands.

MPG and its Device Routines reside on the XXDP/MPG media in absolute format with .MPG extensions while the Executives have .BIN extensions. User programs are also stored in the same format but with .USR extensions. This allows the user to easily distinguish between his stored programs and the files provided as components of MPG. It also eliminates the possibility of duplicate file names.

The format of the filenames assigned to MPG components adhere to XXDP multimedia standards with the exception of the Executives. Since they are the files specified in XXDP LOAD statements, their names more readily indicate their differences and the fact that they are part of MPG. Also, they do not contain revision numbers so that the same name may be used regardless of revision.

For MPG components, a new series of MAINDEC numbers have been allocated and are identified by the first two letters DT. These two letters indicate that MPG is a Diagnostic Tool. As a general rule, the MAINDEC numbers match the component's filenames except for the XXDP conventions of dropping the D and the adding of the MCO level. Once again, the only exceptions are the Executives. The following are the filenames and MAINDEC numbers for the Executives (where n is the revision number and m is the MCO level):

Filename	MAINDEC #	Version
TCMPG.BIN	11-DTE0A-n	TC11 DECtape Executive
RKMPG.BIN	11-DTE1A-n	RK11 DECpack Executive
TMMPG.BIN	11-DTE2A-n	TM11 Magtape Executive
THMPG.BIN	11-DTE3A-n	TM02 Magtape Executive
RXMPG.BIN	11-DTE4A-n	RX11 Floppy Disk Executive
RBMPG.BIN	11-DTE5A-n	RP04 Disk Executive

The actual MPG programs, which can operate only when loaded by one of the Executives, have the following filenames and MAINDEC numbers:

Filename	MAINDEC #	
TMGAnm.MPG	11-DTMGA-n	MPG (Non Memory Management)
TMMAnm.MPG	11-DTMMA-n	MPG (Memory Management)

The following are the filenames and MAINDEC numbers for the Device Routines currently provided with MPG:

Filename	MAINDEC #	Device(s)
TDHAnm.MPG	11-DTDHA-n	DH11
TDJAnm.MPG	11-DTDJA-n	DJ11
TDLAnm.MPG	11-DTDLA-n	DL11
TDQAnm.MPG	11-DTDQA-n	DQ11
TDUAnm.MPG	11-DTDUA-n	DU11
TLPAnm.MPG	11-DTLPA-n	LP11/LS11/LV11
TMSAnm.MPG	11-DTMSA-n	CD11,CR11,CM11,DC11,DN11,KW11 RC11,RF11,RX11,TA11,TM02
TPCAnm.MPG	11-DTPCA-n	PC11/PR11
TRKAnm.MPG	11-DTRKA-n	RK11/RK05
TRPAnm.MPG	11-DTRPA-n	RP04/RP05/RP06
TRSAnm.MPG	11-DTRSA-n	RS03/RS04
TR3Anm.MPG	11-DTR3A-n	RP02/RP03
TR6Anm.MPG	11-DTR6A-n	RK06
TTCAnm.MPG	11-DTTCA-n	TC11/TU56
TTMAnm.MPG	11-DTTMA-n	TM11/TU10

The following sections contain the instructions to generate the various MPG media using the UPD2 program of XXDP. These examples assume DECtape input but can be used as a basis for generating from any medium. Note that the last command in each example transfers the MPG Executives as files. This is not really necessary for MPG operation but is shown in case the user wishes to have these files on the output medium.

8.1 TC11 DECTAPE MEDIA

Assuming the MPG input medium is on drive 0 and the output will be created on drive 1, issue the following UPD2 commands:

COMMAND	FUNCTION
ZERO DT1:	Initializes output medium
LOAD DT0:TCMPG.BIN	Loads DECtape Exec into core
SAVM DT1:	Puts bootable Exec on tape
FILE DT1:<DT0:*.MPG	Puts all MPG files on tape
FILE DT1:<DT0:??MPG.BIN	Puts all MPG Exec's on tape

8.2 RK11 DECPACK MEDIA

Assuming the MPG input medium is DECtape drive 0 and that the output is DECpack drive 0, issue the following UPD2 commands:

COMMAND	FUNCTION
ZERO DK0:	Initializes output DECpack disk
LOAD DT0:RKMPG.BIN	Loads disk Exec into core
SAVM DK0:	Puts bootable Exec on disk
FILE DK0:<DT0:*.MPG	Puts all MPG files on disk
FILE DK0:<DT0:??MPG.BIN	Puts all MPG Exec's on disk

8.3 TM11 MAGTAPE MEDIA

Assuming the MPG input medium is DECtape drive 0 and that the output is MAGTAPE drive 0 (can be seven level or nine level), issue the following UPD2 commands:

COMMAND	FUNCTION
ZERO MT0:	Initializes output MAGTAPE
LOAD DT0:THMPG.BIN	Loads TM02 MAGTAPE Exec into core
SAVE MT0:THMPG.SAV	Puts bootable TM02 Exec on tape
LOAD DT0:TMMPG.BIN	Loads TM11 MAGTAPE Exec into core
SAVE MT0:TMMPG.SAV	Puts TM11 Exec on tape
FILEF MT0:<DT0:*.MPG	Puts all MPG files on tape
FILEF MT0:<DT0:??MPG.BIN	Puts all MPG Exec's on tape

8.4 TM02 MAGTAPE MEDIA

The UPD2 commands necessary to generate a TM02 version of MPG are identical to those needed for the TM11 version. The only difference is that instead of spefifying MT0:, MM0: should be used. The MAGTAPE media produced for the TM11 or the TM02 may be loaded on either the TM11 or the TM02. The bootstrap loader determines the load device type and then loads either the first or the second Executive on tape.

8.5 RX11 FLOPPY DISK MEDIA

Assuming the MPG input medium is DECtape drive 0 and that the output is Floppy Disk drive 0, issue the following UPD2 commands:

COMMAND	FUNCTION
ZERO DX0:	Initializes output Diskette
LOAD DT0:RXMPG.BIN	Loads Floppy Disk Exec into core
SAVM DX0:	Puts bootable Exec on Diskette
FILE DX0:<DT0:*.MPG	Puts all MPG files on Diskette
FILE DX0:<DT0:??MPG.BIN	Puts all MPG Exec's on Diskette

8.6 RP04 DISK MEDIA

Assuming the MPG input medium is DECtape drive 0 and that the output is RP04 drive 0, issue the following UPD2 commands:

COMMAND	FUNCTION
ZERO DB0:	Initializes output disk
LOAD DT0:RBMPG.BIN	Loads disk Exec into core
SAVM DB0:	Puts bootable Exec on disk
FILE DB0:<DT0:*.MPG	Puts all MPG files on disk
FILE DB0:<DT0:??MPG.BIN	Puts all MPG Exec's on disk

8.7 USER PROGRAMS

The UPD2 program can also be used to transfer stored user programs between different MPG media. When doing this, the LOAD and DUMP commands may be used for individual programs while the FILE or FILEF dev:*.USR commands may be used for transferring all user programs.

9.0 MEMORY MANAGEMENT VERSION OF MPG

The Memory Management version of MPG, which has a filename of TMMAnn.MPG, requires 18K words for loading and occupies 16K words or less after housekeeping. Not only does it have all of the features described for the non-Memory Management version, but it also has some additional features unique to its application and some restrictions.

9.1 ADDITIONAL COMMANDS AND FEATURES

The following are the more visable differences between the two versions of MPG:

- During its housekeeping, the Memory Management version of MPG will determine if the Unibus Map and 22 bit addressing are available on the system. If either or both are, MPG will ask if they are to be used. At this time the user may specify any combination of usage for these features.

- When Entering or Fetching a user program, the user is given the opportunity immediately after entering the command to specify the starting address (aligned to a 32 word boundary) of his program's memory area. If an address is entered, the program will begin at that location and will remain at that fixed address until deleted or moved to a new address through use of the SHIFT command. If an address is not entered, MPG will assign a memory address in the same manner as the non-Memory Management version. Note that fixed address programs will be identified by an F in their line of the Memory Map display.

- Two commands have been added and an existing one expanded for this version. Their full descriptions and optional formats are listed in Appendix A.4.

  The SHIFT command allows the user to move his program to a different area of memory and also change its program slot.

  The UBMAP command can be used to change the Unibus Map registers assigned to a user program.

  The MM command with a program number specified, will list the following detailed information about the program:

  -- The standard information line which consists of the absolute program area addresses, the program's name, and the device assigned to the program.

  -- The values that will be loaded into the User Mode PAR and PDR registers when the program is running.

  -- The absolute addresses for RDIO and WRIO.

-- The absolute addresses of FREE, MIDL, and the end of memory.

If the Unibus Map is being used by the program, the following will also be displayed:

-- The Unibus addresses for RDIO and WRIO.

-- The numbers of the Unibus Map registers, which are assigned to the program, and their contents.

-- The Unibus addresses for FREE and MIDL.

-- FREE's Unibus Map register numbers and their contents.

- The "DISPLAY p CODE" command will display the absolute addresses and the virtual addresses of the program's object code.
- The RDM, WRM, BOC, ADD, and SUB commands will support up to 22 bits for their addresses and/or operands. The CBD and CDB commands still support only 16 bits.

9.2 TRAP ROUTINE

A common trap catcher has been added for traps that occur at vectors 4, 10, 114, 250, and non MPG traps at 34. If the trap occurred in a user program while it is running, an attempt will be made to kill that program and continue running any remaining programs. If the trap occurred in MPG, a halt will occur after the display. Depressing the 'CONTINUE' switch after the halt will cause a jump to the restart (RST2) point. When the trap catcher is entered, it will display the following information:

- The vector address of the trap.
- The contents of registers 0 thru 7 and the PSW at the time of the trap.
- The contents of the Memory Management registers (MMR0-MMR3).
- The contents of the failing mode's PAR and PDR registers.
- Whether a user's interrupt was being processed.
- Whether the trap occurred within MPG or a user program. If a user program, its number will be displayed.

9.3 RESTRICTIONS

The following restrictions apply only to the Memory Management version:

- The size of each program's area (device routine + RDIO + WRIO + source + object code) cannot exceed 12K words.
- The size of FREE will be a maximum of 12K words.
- The RESET instruction is a no-no and is not supported.
- The VECTOR instruction is not supported. Therefore, the user will not be able to write and include an interrupt service routine in his program. MPG's generation of absolute addresses for the user program's object code prohibits this feature in a virtual environment.
- Extreme care must be taken if the user is using absolute octal addresses in his program. Octal addresses, which reference the device registers or other registers in the I/O page, are supported. However, if outside of these areas, traps may occur if the area they reference is not mapped.
- The maximum size of an absolute octal address entered on user program statements is 16 bits. This includes high level I/O statements.

9.4 MEMORY MANAGEMENT IMPLEMENTATION DETAILS

MPG utilizes the Memory Management feature in a manner that is compatible with all PDP-11 CPU's. This results in the Data Space and Supervisor Mode capabilities not being used if present.

MPG operates in Kernel Mode and then transfers the mode to User Mode when it gives control to a user program. Prior to giving control, the User Mode PAR/PDR registers are mapped to that user's areas. When the user program gives control back, Kernel Mode will be entered again and MPG will remain in that mode until it gives control back to the same program or to another program. Interrupts generated by user program device routines are serviced in Kernel Mode. Also, certain MPG provided functions utilized by the device routines and program statements (PRINT for example), will result in Kernel Mode being entered for a very brief period to initiate the function.

To aid in the understanding of MPG's operation, the following tables list the usage of the PAR/PDR registers for each mode:

Kernel PAR/PDR	Usage
0 - 3	Always mapped to MPG's program space.
4 - 6	General work and also used during interrupt servicing.
7	I/O Page

User PAR/PDR	Usage
0 - 2	User's program area
3 - 5	The FREE I/O area
6	Shared code area within MPG
7	I/O Page

The values loaded into the User Mode PAR/PDR registers are initialized only when the RUN command is issued or when starting the next pass of the program. When the user program is running and gives up control, the current contents of these registers are stored and then reloaded when it gains control again. This allows the user to set up his own values in certain registers and operate without being concerned that MPG will change them back. However, certain registers must never be changed or the results will be unpredictable. So, keep your hands off of PAR/PDR registers 0, 1, 2, 6, and 7 since they are holy. If you must use a PAR/PDR pair, use one or more of the 3, 4, and 5 group. These are normally for FREE, but if the program never references FREE or MIDL, they can be used for other purposes.

If the user has a program which must know whether it is running under Memory Management or not, the following two statements may be used to determine this fact. The second statement will perform the GOTO if Memory Management is being used:

```
LOAD TMOO WITH OPSW
IF TMOO = 2 GOTO nnnn
```

9.5 UNIBUS MAP IMPLEMENTATION DETAILS

When a device routine is loaded for a user's program, a flag is interrogated to determine if the device requires the Unibus Map. If it does and if the Map is being used, MPG will determine how many contiguous registers are needed (1 to 3) and then ask for a first register number. The valid register numbers are in the range of decimal 7 thru 30. If three registers are needed, the highest valid register number would be 28. If a number is not entered, MPG will select the lowest available registers and then display the number of the first register chosen. If the user later wishes to use different registers with his program, he may issue the UBMAP command and change the registers.

The following table lists the utilization of the Unibus Map registers:

Registers	Usage
0 - 1	Lower area of MPG's program space.
2	Never used within MPG. Available as a work register.
3 - 5	The FREE I/O area.
6	Shared code area within MPG.
7 - 30	Assignable to user programs.

Registers 0, 1, and 6 are initialized by MPG upon its initial loading and whenever MPG is restarted. Registers 3 thru 5 and 7 thru 30 are re-initialized whenever the UBMAP command is issued and when memory's format is changed. The latter occurs after a program has been Entered, Fetched, Deleted, Modified, or Shifted.

APPENDIX A

USER COMMANDS

A.1 MPG COMMAND SUMMARY

For detailed descriptions of the following commands, refer to section A.4.

```
        ENTER p
        ENTER p AS name
        RUN
        RUN p
        RUN p AT ln
        STOP p
        CONT p
        KILL p
        ASSIGN mdl TO p
        ASSIGN mdl,u,u,u TO p
        ASSIGN mdl,u TO LIST            (also SAVE and FETCH)
        DELETE p
        REPORT p
        REPORT p COUNTS                 (also STATUS)
        OPSW oooo
        OPSWp oooo
        SHIFT p TO bbbb                 (Memory Mgmnt only)
        UBMAP p                         (Memory Mgmnt only)
        FILL BFnn WITH *aaa...a
        FILL BFnn WITH oooo,oooo,etc
        FILL BFnn WITH $aaa...a
        FILL BUF WITH PATn
        FILL COMn WITH oooo
        FILL COM WITH *aaa...a
        MODIFY p
        DISPLAY p
        DISPLAY p CODE
        /SAVE p
        /SAVE p AS name
        /FETCH name AS p
        /DELETE name
        /LIST
        /LIST ALL,FAST
        /ZERO
        /BOOT
        MM
        MM p,p,p                        (Memory Mgmnt only)
        FM
        RDM bbbb,cccc
        WRM bbbb,oooo
        BOC bbbb,cccc
        ADD oooo,oooo
        SUB oooo,oooo
        CBD oooo
        CDB dddd
```

A.2 MPG COMMAND ENTRY INFORMATION

The following is information concerning the details and options for entry of MPG commands:

-- All command entries are terminated by a carriage return or a line feed.

-- All command names (ASSIGN,DISPLAY,ENTER,RUN,/LIST,etc) may be entered either in full or in their shortened form which is their first two characters (AS,DI,EN,RU,/L).

-- Separator words (AT,TO,WITH,AS) and certain keywords (BUF,BFnn,COM,COMn,LIST,SAVE,FETCH,CODE,STATUS,COUNTS) may be entered in full or in their shortened form which consists of only their first character (A,T,W,A) and (B,Bn,C,Cn,L,S,F,C,S,C).

For example, "FILL BF02 WITH 1,2,3" could be entered as "FI B2 W 1,2,3".

-- Device model numbers and the keywords NONE, KYBD, KB00, and LOAD must be entered as four characters.

-- In the command summary and definitions, commas are shown as operand separator characters; spaces may be used instead.

-- Leading and trailing commas and/or spaces will be ignored.

-- Multiple commas and/or spaces will be treated as a single comma or space.

-- Leading zeros in program numbers, octal data, decimal data, unit numbers, line numbers, and buffer numbers do not need to be entered. If they are, they will be ignored, regardless of the number of them entered.

-- Several commands accept multiple program numbers and in some cases, intermixed formats. These commands are:

RUN
STOP
CONT
DELETE
REPORT
DISPLAY
KILL
MM (Memory Mgmnt only)

-- The following commands require a program number entry but will accept only one number:

ENTER
ASSIGN
OPSWp
MODIFY
/SAVE
/FETCH
SHIFT (Memory Mgmnt only)
UBMAP (Memory Mgmnt only)

-- All commands that perform I/O operations on either the SAVE device or the FETCH device are prefixed with the slash (/) character.

A.3 MPG COMMAND OPERAND DEFINITIONS

p = The decimal program number in the range of 1 to 16 that is assigned to a user program in MPG.

line number
ln = The one to four digit decimal line number prefixed to each statement in a user program.

mdl = The four character model number of a device that will be used within MPG. This may be the control electronics' model number or the unit's model number.

u = The one or two digit decimal unit number in the range of 0 thru 15 for a specific unit on a multiple unit control electronics.

oooo = A word of data expressed in one to six octal digits.

aaa...a = One or more ASCII characters used to define ASCII data.

name = One to six alpha-numeric characters. Used to identify saved user programs.

bbbb
cccc = An absolute memory address specified in one to six octal digits. May be one to eight digits (22 bits) on the Memory Management version.

dddd = A decimal number in the range of 0 thru 65,535.

n = One digit number in the range of 0 thru 9.

nn = A two digit decimal number in the range of 00 thru 15.

A.4 MPG COMMANDS - DETAILED DESCRIPTIONS

CLASS 1 -- PROGRAM ASSOCIATED COMMANDS

ENTER p

ENTER p AS name

Indicates that the user wishes to enter program statements that will be identified as program number p (1-16). If a program name is not supplied, the default name of ORPHAN will be assigned to the program.

Note: Only one program number will be processed with this command.

RUN p

RUN p AT line number

RUN p,p
RUN p,p AT ln, p
RUN

This command initiates program execution at the first statement of the program indicated by p. A second format, p AT line number, provides for starting a program at a specific statement which is identified by its line number. A single command may be used to start one or more programs with any mix of formats. If no program number is supplied, MPG will scan the program slots and start execution of any user programs that are active. For example, a CONT command has just been issued for a program.

STOP p

STOP p,p,p

Used to stop the execution of one or more programs. Each program will remain intact and may be resumed from the point it was stopped.

CONT p

CONT p,p,p

Used to resume execution of one or more programs that have been stopped by a user command or have ceased execution due to the occurrence of an error. Actual program execution does not resume until a RUN command is entered or a null keyboard entry is made.

KILL p

KILL p,p,p

This command is used primiarily to force termination of user programs that are hung in an internal loop and therefore do not release control to MPG which would allow the STOP command to cease their execution. KILL can also be used to stop a program that is not in a loop. However, any program stopped with the KILL command will be stopped immediately and can be restarted only with the RUN command.

```
DELETE p
DELETE p,p,p
```

Instructs MPG to delete the indicated programs and deallocate the memory they occupied.

```
ASSIGN mdl TO p

ASSIGN mdl,u,u TO p
ASSIGN mdl,u,u-u TO p
ASSIGN mdl,u TO SAVE
ASSIGN mdl,u TO FETCH
ASSIGN NONE TO LIST
```

Allows the user to change the unit numbers assigred to an existing program and/or to change the device's constants. If a single unit device, no unit number is needed. If a multiple unit device, such as magnetic tape, either a single unit number, a series of single unit numbers, and/or a string of consecutive unit numbers (maximum of sixteen) may be specified. When using this command to change unit numbers, a line feed terminator on the reply will cause MPG to bypass requesting new device constants.

The NONE entry is used to deassign the current device for the LIST, SAVE, and FETCH devices. Other keywords which may be used for deassignment are KYBD and KBOO for LIST and LOAD for SAVE and FETCH. It should be noted that for the LIST, SAVE, and FETCH operands, entry of only their first character (L,S,F) is acceptable.

```
REPORT p

REPORT p,p STATUS,p
REPORT p COUNTS
```

Forces the display of all device registers, error information, and statistics for the program or programs specified. This command applies only to those programs that utilize high level I/O language statements and device routines which provide reports. A variation of this command allows the user to display specific report information. The COUNTS operand will display only the counts while the STATUS operand will display only the device registers. If neither of these operands are specified, both reports will be displayed. Entry of only the first letter for COUNTS and STATUS is acceptable.

OPSW oooo

OPSWp oooo

Used to set the MPG Operation Switches to the values indicated by oooo which is an octal word (Refer to Section 3.5). If the program number p is not entered, the Operation Switches for all currently resident programs will be set to the indicated value. When p is entered, only that program's switches will be modified. When entering a program number, only one may be entered.

SHIFT p TO bbbb
SHIFT p TO p
SHIFT p TO MPG

This command is supported only on the Memory Management version of MPG. The first format is used to move the specified program to a new area in memory whose starting address is defined by the octal address bbbb. The move will be performed as long as the area is not already occupied and there is enough room. The second format allows the user to change his program's number and consequentially its polling loop slot. If this command is directed at a program whose memory address is controlled by MPG, the program may be relocated in memory in order to retain the sequential memory addressing common to MPG controlled programs. The third format allows the user to return a program, which is located at a fixed address that was defined earlier, to MPG control of its address.

UBMAP p

This command is supported only on the Memory Management version of MPG. With this command the user may change the Unibus Map registers assigned to program p. The first action performed is to de-assign the registers currently assigned to the program. Next, it will ask for new register assignments in a manner which is identical to the way it requested the register assignments when the program was Entered or Fetched. After the assignments are received, all programs will be recompiled.

CLASS 2 -- DATA PROCESSING COMMANDS

```
FILL BFnn WITH oooo,oooo,etc

FILL BFnn WITH *aaa...aa
FILL BUF WITH oooo,oooo,etc
FILL BFnn WITH #aaa...a
FILL BUF WITH PATn
```

Allows the user to load one of the sixteen common data buffers with either octal words or ASCII data. The value of nn may be 0 thru 15 and specifies which 16 Byte buffer is to be filled. If less than 16 bytes of octal words or ASCII data is specified, the entered data will be propagated throughout the 16 bytes. If more than 16 bytes are entered, the data will be truncated after the first 16 bytes. In the third format, where n is not specified, all sixteen buffers will be treated as one 256 byte buffer. Octal or ASCII data will be propagated throughout its entire length with no restriction (other than the size of the 71 byte keyboard read-in area) on the number of bytes entered.

The prefix codes * and # for the ASCII data formats provide the user with control over the inclusion of carriage return and line feed characters at the end of his data. The * format will automatically include the CR/LF while the # format will result in only the data entered by the user. When propagating data, the CR/LF will be treated the same as user data.

The PATn format will fill the indicated buffer with the bit pattern specified by PATn. The various patterns available are listed in Appendix C.3. For this format n can have a value of 0 - 9, A, or B.

User Commands (Cont'd)

FILL COMn WITH oooo

FILL COM WITH *aa..a
FILL COMn WITH #aa
FILL COM WITH PATn

This is an alternate format of the FILL command and allows the user to access the ten common work areas which are two bytes each. The number n represents the work area number and may be in the range of 0 thru 9. The data may be entered as a pattern or in either octal or ASCII format. If a number is not supplied, COM0 through COM9 will be treated as a twenty byte area with data propagation taking place if less than twenty bytes of data is entered.

The codes *, #, and PATn are treated the same as in the FILL BFnn command. However, when filling a specific COM word (COMn) and using the *aa format, the resulting contents of the COMn word will be a CR/LF, regardless of the number of data characters entered.

CLASS 3 -- USER PROGRAM SOURCE MAINTENANCE COMMANDS

MODIFY p

Instructs MPG to interpret the subsequent keyboard entries as program statements which are to be applied as corrections to the program specified by p. The program p must already exist before using this command.

DISPLAY p

DISPLAY p CODE
DISPLAY p,p CODE,p

Causes the display of the program p's statements on the MPG print device. The second format causes the display of program statements and the octal machine code generated for each statement.

/SAVE p

/SAVE p AS name

Instructs MPG to write the program statements of program p to the current SAVE device. They will be saved under the one to six character I.D. code furnished by name. If a name is not specified, the statements will be saved under the name assigned to the program in memory. Program p will still be retained in memory.

/FETCH name AS p

Used to read programs previously saved by MPG. The name entry furnishes the I.D. of the saved program on the current FETCH device while p indicates the program slot into which it will be loaded.

User Commands (Cont'd)

/DELETE name

/DELETE name,name,name

This command is used to delete user programs stored on the SAVE device. One or more names may be supplied.

/LIST

/LIST ALL
/LIST FAST
/LIST ALL,FAST

The directory of the FETCH device may be listed through use of this command. If the keyword ALL is not supplied, only the saved user programs will be listed. If ALL is entered, all files, including those required by MPG, will be listed. If FAST is entered, only the filenames of the files will be listed. ALL and FAST are independent of each other. The information produced by this command will be displayed on the current LIST device (either the console terminal or the printer).

/ZERO

Use of this command will initialize the directory of the SAVE device. It is useful when creating alternate program library media. Extreme care must be taken when using this command. If directed at the MPG load device, it will destroy the data on that device. It is for this reason that a warning message, which requires a reply, is displayed before performing the Zero function. This message displays the SAVE device's model and unit numbers and requires a yes reply before performing the Zero function.

/BOOT

Causes MPG to perform a simulated bootstrap function on the current FETCH device. The first data block read by a normal bootstrap will be read by MPG into memory starting at location zero with a branch to location zero after the read completes. This destroys the memory resident MPG.

CLASS 4 -- SERVICE COMMANDS

MM

Display the limits of the memory locations occupied by each user program and the addresses of Free Memory. (MM stands for Memory Map)

MM p MM p,p,p	This command is supported only on the Memory Management version of MPG. With this command the user may display more detailed information about a single or series of user programs. Refer to Section 9.1 for details as to what information is displayed.
FM	This command formats memory by eliminating any unused memory space between MPG controlled user programs and then recompiling all user programs. This situation shouldn't occur but under some unique error or abort conditions it might.
RDM bbbb RDM bbbb,cccc	Display the contents of the word at the octal memory address specified by bbbb. If the optional second address is supplied, display up to and including the second address.
WRM bbbb,oooo WRM bbbb,oooo,oooo,etc	Write the single or string of octal words (oooo) beginning at the octal memory address specified by bbbb.
BOC bbbb,cccc	Used to calculate the branch offset value between two even memory addresses. The first address (bbbb) is the memory address of a branch offset type of branch instruction with the second address (cccc) being the destination address of the branch. The answer produced must be visually checked for an out of range condition by the user.
ADD oooo,oooo,etc	Add the two or more octal numbers and display their result.
SUB oooo,oooo	Subtract the first octal number from the second and display the result.
CBD oooo	Convert the octal binary number to decimal and display the decimal result.
CDB dddd	Convert the decimal number to binary and display it in octal.

APPENDIX B

USER PROGRAM LANGUAGE

B.1 MPG PROGRAM LANGUAGE SUMMARY

See the following pages for detailed description and definition of v, n, and ln. ( ) indicates other valid symbolic operators.

```
LOAD    v  WITH v
INCR    v
INCR    v  BY v
DECR    v
DECR    v  BY v
MOVE                                (NBR,SRC,DST)
MOVE    v  TO v
MOVE    v  AT v TO v
ADD     v  TO v
ADD     v  AT v TO v
SUB     v  FROM v
SUB     v  AT v FROM v
NEGATE  v
SET     v  BIT n
SET     v  BIT n THRU n AND n        (-,&)
CLEAR   v  BIT n
CLEAR   v  BIT n AND n THRU n        (&,-)
IF      v  BIT n SET GO TO ln        (SET,CLEAR)
IF      v  BIT n&n-n CLEAR GO TO ln  (CLEAR,SET)
IF      v  = v GO TO ln              (=,>,<,=>,<=,<>)
IF      v  AT v => v GO TO ln        (=,<>)
GO TO   ln
LINK    ln
RETURN
PRINT   *  text-text-text-text
PRINT   $  text-text-text-text
PRINT   v  IN BINARY                 (BINARY,OCTAL,DECIMAL,ASCII)
PRINT   v  AT v IN ASCII             (ASCII,OCTAL,DECIMAL,BINARY)
FILL    v  WITH RANDOM               (RANDOM,ASCII,[variable],PATn)
FILL    v  AT v WITH ASCII           (ASCII,RANDOM,[variable],PATn)
ROTATE  v
ROTATE  v  AT v
PAUSE
DELAY   v
VERIFY  v  WITH v
VERIFY  v  AT v WITH v
VECTOR  v  TO ln                     (N/A for Memory Mgmnt)
LETGO
ENTRY
EXIT
RESET                                (N/A for Memory Mgmnt)
WORD v
END
```

For specific peripheral instructions such as READ, WRITE, REWIND, SEEK etc., see writeup listed under appropriate device name, in Appendix D.

B.2 MPG PROGRAM LANGUAGE - VALID PARAMETERS

VARIABLES (v) - Any of the following are equally interchangeable:

Symbols	- Any predefined symbolic name listed in Appendix C or Appendix D. Includes those for temporary storage, common storage and device register names.
Octal Numbers	- Any octal number from 0 thru 177777.
Decimal Numbers	- Any number preceded by a 'D' will be considered a decimal number. The acceptable range is from: D0 thru D65535.

BIT NUMBERS (n) - Any decimal number from 0 thru 15.

LINE NUMBERS (ln) - Any decimal number from 0 thru 9999.

B.3 PROGRAM LANGUAGE DETAILED DESCRIPTION

LOAD [variable] WITH [variable]

Load the value of the second variable into the memory address specified by the first.

INCR [variable]

INCR [variable] BY [variable]

Increment the contents of the memory word location indicated by the first variable by the amount specified by the second. Increment by 1 if the second variable is not specified.

DECR [variable]

DECR [variable] BY [variable]

Decrement the contents of the memory word location indicated by the first variable by the amount specified by the second. Decrement by 1 if the second variable is not specified.

MOVE

The MOVE instruction with no parameters implies an indirect move. The number of bytes to be moved is contained in location NBR. Location SRC contains the starting address of the data, while location DST points to where it is to be stored.

MOVE [variable] TO [variable]

Move one word from the memory location indicated by the first variable to the location indicated by the second.

MOVE [variable] AT [variable] TO [variable]

Same as basic MOVE except the first variable specifies the number of bytes to be moved.

ADD [variable] TO [variable]

Add one word located at the memory location indicated by the first variable to the word residing at the location indicated by the second. Replace the second with the result.

ADD [variable] AT [variable] TO [variable]

Same as basic ADD except the first variable specifies the number of bytes to be added.

SUB [variable] FROM [variable]

Subtract one word located at the memory location indicated by the first variable from the word residing at the location indicated by the second. Replace the second with the result.

SUB [variable] AT [variable] FROM [variable]

Same as basic SUB except the first variable specifies the number of bytes to be subtracted.

NEGATE [variable]

Two's complement the contents of the memory location referenced by variable.

SET [variable] BIT [n] - [n]&[n]

Set the bit identified by the number n in the location specified by variable.

Bit combinations (AND,&) or bit groups (THRU,-) can also be set.

CLEAR [variable] BIT [n] THRU [n] AND [n]

Reset the bit identified by the number n in the location specified by variable.

Bit combinations (AND,&) or bit groups (THRU,-) can also be reset.

```
IF [variable] BIT [n] SET GO TO [line number]
                      CLEAR
```

If the bit identified by the number n in the location specified by variable meets the condition called for (Set or Clear) then perform a program branch to the statement indicated by line number.

```
IF [variable] BIT [n] AND [n]&[n] THRU [n] SET GOTO [line number]
                       &    AND    -       CLEAR
```

Bit combinations (AND,&) or bit groups (THRU,-) can be identified for test.

User Program Language (Cont'd)

```
IF [variable] = [variable] GOTO [line number]
              >
              <
              =>
              <=
              <>
```

If the relationship (=,> etc), between the contents of memory specified by the variables, is true, perform a program branch to the statement identified by line number. Unsigned branch instructions will be used.

NOTE- There is one difference between this instruction and all others. If the second variable is a decimal or octal number, it is used as a value, not as an address. If it is a symbolic name (tag), it is used as an address that contains the value to be compared.

```
IF [variable] AT [variable] = [variable] GO TO [line number]
                            <>
```

Same as preceding IF operation except that the first variable specifies the number of bytes to be compared. Note that either GOTO or GO TO is acceptable, and only a compare for = or <> is valid.

GOTO [line number]

Causes an unconditional program branch to the statement indicated by line number. Either GOTO or GO TO are valid.

LINK [line number]

Used as a jump to subroutine to a subroutine beginning at the statement indicated by line number.

RETURN

Used as a return from a subroutine.

PRINT *[text-text-text]

When this statement is encountered, print the text message on the display device assigned to this program. Follow the text with a Carriage Return and Line Feed.

PRINT #[text-text-text]

Same as PRINT *, except that the text is printed without a carriage return and line feed following.

PRINT [variable] IN BINARY
OCTAL
DECIMAL
ASCII

Convert one word of data located in memory at the address specified by variable to the format requested and print it on the assigned display device. First character only (B,O,D,A) is acceptable.

PRINT [variable] AT [variable] IN BINARY
OCTAL
DECIMAL
ASCII

The first variable specifies the number of bytes to be printed (starting at the location specified by the second variable). A decimal or octal number is valid.

FILL [variable] WITH RANDOM
ASCII
[variable]
PATn

Fill the area of memory starting with the location specified by the first variable with 256 bytes of random data, the ASCII character set, the bit combination specified by the second variable, or the specified pattern (see Appendix C.3).

FILL [variable] AT [variable] WITH RANDOM
ASCII
[variable]
PATn

Specify the number of bytes to be filled by the first variable.

ROTATE [variable]

Rotate the number of bytes specified by variable, in this program's write buffer (WRIO), one place to the right.

ROTATE [variable] AT [variable]

Rotate the number of bytes specified by the first variable starting at the location specified by the second variable, one place to the right.

VECTOR [variable] TO [line number]

Store the address of the line number in the location specified by the variable. This location is normally an interrupt vector address. (Note: Not supported on the Memory Management version of MPG.)

User Program Language (Cont'd)

VERIFY [variable] WITH [variable]

Compare two bytes at the address specified by the first variable, with two bytes at the address specified by the second. Print any pair in octal along with the byte number if they do not compare.

VERIFY [variable] AT [variable] WITH [variable]

Same as previous except first variable specifies the number of bytes to be compared.

PAUSE

Compiles as a Wait for Interrupt instruction.

DELAY [variable]

Delay the number of milliseconds indicated by variable. This should be a decimal value such as D1000 for 1 second. This instruction uses a simple time-out loop, and the actual delay will vary with memory and processor speed. If accuracy is critical, the User should time a delay of 60 Seconds (D60000), and adjust the value he desires by the percentage deviation from the one minute delay.

ENTRY

Save registers R0 thru R5 on the stack. This instruction is for use at the start of an interrupt service routine written in MPG language.

EXIT

Restore registers R0 thru R5 from the stack, then do a return from interrupt (RTI) instruction. This instruction is for use at the end of an MPG interrupt routine.

LETGO

Causes a control release as described in Section 4.2. Execution resumes inline when control is returned to this program.

RESET

Issue a Bus Initialize to the hardware. (Note: Not supported on the Memory Management version of MPG.)

WORD v

Store the binary value of the variable specified as the next word in the user program. Octal and decimal values are valid for v. Also, if a symbolic name is entered for v, the address for the symbolic name will be generated. Note: USE AT YOUR OWN RISK.

END

End of program. Must have the highest line number. Causes return to the Control Routine when its object code is executed during program execution.

APPENDIX C

PREDEFINED SYMBOLIC NAMES

C.1 USER PROGRAM EXCLUSIVE STORAGE

Each MPG user program has access to 16 locations that are part of the program and that have predefined symbolic names. Each program uses the same symbolic names, but when two or more programs are co-resident, these common names reference different physical locations internal to each program. The valid predefined symbols are as follows:

TM00
TM01
TM02
TM03
TM04
TM05
TM06
TM07
TM08
TM09
TM10
TM11
TM12
TM13
TM14
TM15

Likewise, the symbolic names RDIO and WRIO refer to each of the two I/O areas included as part of each program area. The locations NBR, SRC, and DST, which are used by the indirect move instruction (MOVE), and the operation switch word (OPSW) also exist separately within each program area.

RDIO
WRIO
OPSW
NBR
SRC
DST

C.2 USER PROGRAM COMMON STORAGE

Each MPG user program also has access to 10 locations that are physically common to all programs resident at any given time. They are also accessable by the FILL command. These may be used for communication between programs, or variation of parameters by command. The valid predefined symbols are as follows:

COM0
COM1
COM2
COM3
COM4
COM5
COM6
COM7
COM8
COM9

The common or shared I/O area is referenced by the symbolic name FREE. The middle of this area is referenced by the symbolic name MIDL.

FREE
MIDL
CRLF

A common location containing the ASCII code for a Carriage Return and Line Feed can be referenced as CRLF, and used for formatting messages.

Symbolic Names (Cont'd)

The common data buffer is accessable both by MPG command and by each program, and allows the user to generate and change data without modifying the program itself. The buffer is 256 bytes long and can be referenced in subdivisions of 16 bytes using the following symbols:

```
BUF/BF00    BF04    BF08    BF12
    BF01    BF05    BF09    BF13
    BF02    BF06    BF10    BF14
    BF03    BF07    BF11    BF15
```

For the FILL command, this buffer can be referenced as a single 256 byte buffer by using the name BUF or just B. If BUF is used in a program statement, it is equivalent to BF00.

C.3 MPG PREDEFINED BIT PATTERNS

Inherent within MPG is the ability to easily generate variable length data fields with any of a series of predefined bit patterns. These fields may be generated with either the FILL program statement or the FILL command and are useful when checking for failures due to data sensitivity. These patterns are identified by symbolic names that are listed below along with the bit patterns that each produces:

```
PAT0      100000        WALKING ONES
          040000
          020000
          010000
           etc.

PAT1      077777        WALKING ZEROES
          137777
          157777
          167777
           etc.

PAT2      100000        EXPANDING ONES
          140000
          160000
          170000
           etc.

PAT3      077777        EXPANDING ZEROES
          037777
          017777
          007777
           etc.
```

Symbolic Names (Cont'd)

PAT4	125252 125252 125252 125252 etc.	ALTERNATE ONES (HORIZ.)
PAT5	052525 052525 052525 052525 etc.	ALTERNATE ZEROES (HORIZ.)
PAT6	177777 000000 177777 000000 etc.	ALTERNATE ONES (VERT.)
PAT7	000000 177777 000000 177777 etc.	ALTERNATE ZEROES (VERT)
PAT8	070707 107070 070707 107070 etc.	OCTAL CHECKERBOARD
PAT9	125252 052525 125252 052525 etc.	BINARY CHECKERBOARD
PATA	000000 000001 000002 000003 etc.	COUNT WORDS
PATB	165555 133333 165555 133333 etc.	RP04 DISK SERIAL DATA

APPENDIX D

DEVICE ROUTINES

D.1 DEVICE ROUTINES - GENERAL INFORMATION

The information unique to a particular device and the routines necessary to perform the operations indicated by its high level language statements (READ, WRITE, SEEK, etc.) are contained in an MPG program segment referred to as a Device Routine. The Device Routines are resident on the MPG load device and are automatically loaded by MPG when the device is specified by model name for a user program.

While device routines are primarily drivers for the various PDP-11 devices, they also provide other services to the user and his program and also to MPG. Services for the user include:

- High level language functions
- Statistic accumulation and display
- Error detection and display
- Device registers display
- Symbolic names for device registers

Services for MPG are actually extensions of MPG and are activated by MPG when necessary. These services include housekeeping functions, report displays, and functions required for the KILL command.

The device routines serve as add on components to the basic MPG program. Without them it is still possible to write programs for any device by using the low-level language. This basic feature allows programming at the device interface level. With them, higher level functions such as Read, Write, Seek, Rewind, etc., can be invoked with one MPG statement.

A feature provided by the device routines is the specifying of device registers by symbolic name. In order to reference a device's registers by name, the MPG user must assign that device to the program he is defining. As an extension of this support, abbreviated device routines exist for all devices mentioned in section D.2 even though high level functions have not yet been programmed.

The remaining sections of Appendix D contains documentation pertaining to the devices for which high level functions have been programmed.

D.1.1 COMMON FEATURES IN DEVICE ROUTINES

Although each device routine is tailored to the device it supports, several features are common among all full support device routines unless otherwise noted. These features, which will be documented only once to avoid redundancy, are described in this section. When a device routine description lists one of these features, it will direct the user to this section for detailed information.

OPSW Special Operation Bit

One bit in the Operation Switches (OPSW) word associated with each user program has been reserved for use as a flag to the device routines. This bit, which is called the Special Operations (SPOPER) flag, is bit 7 of the OPSW and may or may not be supported by a device routine. A typical application of this bit's usage would be its control of the Maintenance bit in the device's registers. When SPOPER is set by the user, it would indicate to the device routine that all I/O operations are to be issued with the device's Maintenance bit set. For other devices it may have a completely different implementation and meaning. Therefore, refer to the applicable device routine section to determine if this bit is utilized and in what manner.

Remember that the OPSW can be modified either at the command level with the OPSW command or in the user's programs with the LOAD, SET, or CLEAR instructions.

Information Words

A total of eight words are available to each device routine for use in passing information between itself and the user's program. Six of these words may be implemented as needed for the device routine's own requirements. Examples of these words are CYL, HEAD, and SECT for a disk. The other two words, which have been assigned specific symbolic names and functions for all device routines, are as follows:

Name	Definition
SIZE	A one word location which is loaded by the device routine with the number of bytes actually transferred on an I/O operation that performed data transfers.
ERR	A one word location which is cleared to 0's before each I/O operation. Will be set to a value of 1 if an error was detected on the I/O operation.

Statements

The following statements will be accepted by all full support device routines and will produce similar results:

NOWAIT This statement instructs the device routine to go into a mode where it returns immediately to the next statement in the user's program after initiating any I/O operations. This allows the user to initiate an I/O operation and then sample the device registers while it is in progress or to perform concurrent data operations.

For communication devices, this mode also allows a simultaneous READ and WRITE in order to test the data loop back provided by the hardware. Multiple I/O operations can be started in parallel as long as different units (lines) are selected. When any subsequent statement attempts to select a line that is already in use, the program will wait at this point until the initial I/O on any common line is completed.

If NOWAIT is issued on a non-communications type of device and another I/O statement is issued before a previously issued one has terminated, the device routine will automatically wait until the operation in progress is completed and then initiate the next operation.

When using this statement, care must be taken to ensure that a WAIT is issued after the last I/O operation in the program. Otherwise, an error display may be missed and the final counts will not be updated for the last I/O operation.

WAIT Resets the NOWAIT mode and instructs the device routine to return to its preset mode of operation. In its preset mode, the device routine will wait until the current I/O operation has terminated before returning to the next statement in the user's program.

COUNTS This statement produces a display of the statistical counts accumulated thus far in the operation of the user's program. This display will be directed to the MPG LIST device (console terminal or printer) with the information produced being dependent upon the device routine.

STATUS With this statement the user can display the contents of the device registers associated with the device assigned to his program. The display will be directed to the MPG LIST device with each register's contents being identified by the register's symbolic name. Normally, two sets of contents will be displayed. The first set is the registers' contents when the last interrupt occurred. If there has not been a previous interrupt, these will not be displayed. The second set consists of their current contents and will always be displayed. For the second display, the contents of the registers are retrieved upon processing this statement.

D.1.2 COMMON ERROR INFORMATION FOR DEVICE ROUTINES

When encountering an error and if permitted to do so by bits 8 and 13 in the program's OPSW, the device routine will display information pertaining to the error. This information will be directed to the MPG LIST device and as a minimum will consist of an error I.D. and the line number of the current statement in the user's program. Depending upon the device routine and the type of error, additional information may also be displayed. This usually consists of the current unit number, the contents of the device registers at the time the error was detected, and the error bit mnemonics. For details as to what is produced by a specific device routine, refer to that routine's section in this appendix.

There exists within MPG three error messages which may be issued during execution of a user program. While these errors will be displayed with the program's number as an identifier, they are issued by MPG and not by the device routine. The first message may occur when the VERIFY statement is being used within the user's program. The other two messages may occur when a user program or device routine is giving control back to MPG with its stack having improper attributes. These messages and their explanations are as follows:

DATA ERROR (STMNT # nnnn)

A data miscompare has been detected by the VERIFY statement. The data that did not compare will be displayed following this message. Whether to list only the first miscompare or all miscompares is controlled by bit 1 in the program's OPSW. The data error count within the device routine will be incremented when this error occurs.

ER INV STK ADR

The value of the stack pointer was greater than the maximum allowable value when a user program or device routine returned control to MPG. The user program will be terminated immediately and cannot be continued through use of the CONT command.

ER STK TOO BIG

This error occurs when a user program or device routine returns control to MPG and has left more than thirty words on the stack. Since storage space reserved for stack information is thirty words long, stack data would be lost for the program. Therefore, this is reported as an error condition and the program is terminated. The CONT command cannot be used after its occurrence.

After any error is displayed, other than the stack control errors, execution of the user program will cease, unless directed to continue on error by bit 15 being reset in the program's OPSW. If the program stops after an error and the CONT command is then issued, execution will resume at the next statement in the user's program.

D.2 MINIMUM SUPPORT DEVICE ROUTINES

For certain devices MPG provides a minimum level of support. This support consists of symbolic names for device registers when programming a device at the interface level. This capability eliminates the need for octal addresses when accessing these registers. Also, by altering the device registers' base address, the same program may be run without modification on a similar device but at a different Unibus address.

In order to reduce file usage, the information for all minimum support devices has been included in one file which has the filename of TMSAnm.MPG. When one of these devices is specified for a program, this file will be loaded as its device routine. Immediately after loading the file, code will be executed that deletes the information for all devices other than the one specified. The length of the routine's area in memory is then adjusted to the size needed for only the specified device's information.

The device model names and the symbolic names supported for each device are listed in the following table. Also listed is the displacement that will be added to the device registers' base address for each symbolic name.

DEVICE	NAME	DISPL	DESCRIPTION
CD11	CDST	+0	Card Status
	CDCC	+2	Column Count
	CDBA	+4	Current Address
	CDDB	+6	Card Data
CR11	CRS	+0	Card Status
CM11	CRB1	+2	Card Data
	CRB2	+4	Card Data Compressed
DC11	RCSR	+0	Receiver Status
	RBUF	+2	Receiver Data Buffer
	TSCR	+4	Transmitter Status
	TBUF	+6	Transmitter Data Buffer
DN11	ACU	+0	Auto Call Unit
KW11	KWSC	+0	R.T. Clock Status
	KWBR	+2	R.T. Clock Buffer
	KWCR	+4	R.T. Clock Counter
	LKS	+5006	Line Time Clock Status

D.2 - Minimum Support Device Routines (Cont'd)

RC11	RCLA	+0	Look Ahead
	RCDA	+2	Disk Address
	RCER	+4	Error Status
	RCCS	+6	Control and Status
	RCWC	+10	Word Count
	RCCA	+12	Current Address
	RCMN	+14	Maintenance
	RCDB	+16	Data Buffer
RF11	DCS	+0	Control and Status
	WC	+2	Word Count
	CMA	+4	Current Memory Address
	DAR	+6	Disk Address
	DAE	+10	Address Extension Error
	DBR	+12	Disk Data
	MA	+14	Maintenance
	ADS	+16	Address of Disk Segment
RX11	RXCS	+0	Command and Status
RX01	RXDB	+2	Data Buffer
	RXTA	+2	Track Address
	RXSA	+2	Sector Address
	RXES	+2	Error and Status
TA11	TACS	+0	Control and Status
	TADB	+2	Data Buffer
TM02	MTC1	+0	Control and Status 1
TU16	MTWC	+2	Word Count
	MTBA	+4	Unibus Address
	MTFC	+6	Frame Count
	MTC2	+10	Control and Status 2
	MTDS	+12	Drive Status
	MTER	+14	Error
	MTAS	+16	Attention Summary
	MTCC	+20	Character Check
	MTDB	+22	Data Buffer
	MTMR	+24	Maintenance
	MTDT	+26	Drive Type
	MTSN	+30	Serial Number
	MTTC	+32	Tape Control

D.3 DJ11 16-LINE ASYNCHRONOUS SERIAL LINE MULTIPLEXER

PRESET ADDRESS - 160010
PRESET INTERRUPT VECTOR - 300
PRESET BUS REQUEST - 5,5 (READ/WRITE)

A. REGISTER NAMES RECOGNIZED

CSR
RBUF
TCR
BCR
TBUF

B. MPG INSTRUCTIONS SUPPORTED

READ
WRITE
BREAK
CRESET
NOWAIT
WAIT
FDUPLX
HDUPLX
STATUS
COUNTS

C. STATISTICAL INFORMATION

BYTES:	Read, Written
CMNDS:	Read, Write, Break, Misc.
INTERRUPTS:	Read, Write
ERRORS:	Overrun(OVR), Framing(FRM), Parity(PAR), Data

D. OPSW SPECIAL OPERATION BIT SUPPORT

When Bit 7 (SPOPER) of the OPSW is set, the Maintenance Mode Bit (Bit 2 CSR) will be set prior to issuing each I/O data transfer command.

E. I/O TIMEOUT

If a terminating interrupt is not received 3 minutes after initiating an I/O operation, the program will be aborted and the user informed. This time is approximate, calibrated on a PDP-11/45 with no other user programs running.

F. INSTRUCTION DESCRIPTIONS

```
READ  (D256 INTO RDIO)      () = default values
READ  v (INTO RDIO)
READ  v INTO v
```

This instruction will cause a data transfer to occur between the unit previously assigned to this Device Routine and the memory area specified by the second variable. The first variable contains the byte count for the transfer.

```
READ  v INTO v FROM u
```

Same as above read except U specifies the units or line numbers. Whatever unit numbers were assigned have no effect. This allows simultaneous data transfers on up to 16 lines. A typical form of U is 4-7&11 or 4 THRU 7 AND 11.

```
WRITE  (D256 FROM WRIO)
WRITE  v (FROM WRIO)
WRITE  v FROM v
WRITE  v FROM v TO u
```

The description for the Read Instruction applies to the Write except for the direction of data flow.

```
BREAK  v
BREAK  v ON u
```

This instruction causes a break (spacing condition) on the assigned unit or on those specified by u as in the Read Instruction. The duration at the break is determined by v which specifies the number of character times to hold the spacing condition.

```
CRESET
```

This instruction initiates an MOS clear sequence which clears the silo and all 16 UART's. It then clears all of the bits in the Control Register.

NOWAIT

Standard implementation - see section D.1.1.

WAIT

Standard implementation - see section D.1.1.

FDUPLX

Places the DJ11 in a Full Duplex Mode. This is the initial mode of a DJ11 after a RESET or CRESET.

HDUPLX

Places the DJ11 in a Half Duplex Mode.

STATUS

Standard implementation - see section D.1.1.

COUNTS

Standard implementation - see section D.1.1.

D.3.1 DJ11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2.

The unique error messages which may occur in the use of this device routine are as follows:

DJ11 TIMEOUT ON I/O

An I/O operation was begun, but the terminating interrupt was not received. The time allowed for this interrupt to occur is approximately three minutes when the program is run on the PDP-11/45 and no other programs are running.

DJ11 ERROR: ddd,ddd,etc

The codes ddd represent mnemonics which are printed to indicate which error bits were set. The message may contain from one to three codes. The following mnemonics may be printed in this message:

PAR = Received Data Parity Error
FRM = Framing Error
OVR = Overrun Error

D.3.2 DJ11 SAMPLE PROGRAMS

1. Write same data sequentially to units 5,6,7 and 14

```
*ENTER 1
?ASGN  DEV: DJ11,5,6,7,14
>0010  FILL D64 AT WRIO WITH ASCII
>0020  WRITE D64 FROM WRIO
>0030  WRITE 2 FROM CRLF
>0040  END
>DONE

*RUN 1
```

2. Write same data simultaneously to units 5,6,7, and 14

```
*ENTER 1
?ASGN DEV: DJ11

>0010  FILL D64 AT WRIO WITH ASCII
>0020  WRITE D64 FROM WRIO TO 5-7&14
>0030  WRITE 2 FROM CRLF TO 5 THRU 7 AND 14
>0040  END
>DONE

*RUN 1
```

3. Write different data sequentially to units 1,4 and 11

```
*ENTER 1
?ASGN DEV: DJ11

>0010  WRITE D16 FROM BF00 TO 1
>0020  WRITE D16 FROM BF01 TO 4
>0030  WRITE D16 FROM BF02 TO 11
>0040  END
>DONE

*FILL BF00 WITH *TEST MSG NBR 1
```

```
*FILL BF01 WITH *TEST MSG NBR 2
*FILL BF02 WITH *TEST MSG NBR 3
*RUN 1
```

4. Write different data simultaneously to units 1,4 and 11

```
*ENTER 1
?ASGN DEV: DJ11

>0010  NOWAIT
>0020  WRITE D16 FROM BF00 TO 1
>0030  WRITE D16 FROM BF01 TO 4
>0040  WRITE D16 FROM BF02 TO 11
>0050  WAIT
>0060  END
>DONE
```

5. Read 1 character from unit 12 and Write it to units 12 and 14. Use the operation switch to loop on this program.

```
*ENTER 1
?ASGN  DEV: DJ11

>0010  READ 1 INTO TM00 FROM 12
>0020  WRITE 1 FROM TM00 TO 12 AND 14
>0030  END
>DONE

*OPSW1 140020
*RUN   1
```

6. Test the DJ11 using the Special Operation Bit to cause data wrap-around. Set the OPSW to loop the program and turn on the Maintenance Bit. Use 256 bytes of random data each time, and sequentially check all 16 lines.

```
*ENTER 1
?ASGN DEV: DJ11,0-15

>0010  FILL D256 AT WRIO WITH RANDOM
>0020  NOWAIT
>0030  READ
>0040  WRITE
>0050  WAIT
>0060  VERIFY D256 AT RDIO WITH WRIO
>0070  END
>DONE

*OPSW1 100214
*RUN 1
```

D.4 DL11 SINGLE ASYNCHRONOUS SERIAL LINE INTERFACE

PRESET ADDRESS - 175610
PRESET INTERRUPT VECTOR - 300
PRESET BUS REQUEST - 4,4 (READ/WRITE)

A. REGISTER NAMES RECOGNIZED

RCSR
RBUF
XCSR
XBUF

B. MPG INSTRUCTIONS SUPPORTED

READ
WRITE
BREAK
CRESET
WAIT
CALL
LISTEN
ANSWER
SEND
RECV
STATUS
COUNTS

C. STATISTICAL INFORMATION

BYTES:	Read, Written
CMNDS:	Read, Write, Break, Misc.
INTERRUPTS:	Read, Write
ERRORS:	Overrun(OVR), Framing(FRM), Parity(PAR), Data

D. OPSW SPECIAL OPERATION BIT SUPPORT

When Bit 7 (SPOPER) of the OPSW is set, the Maintenance Mode Bit (Bit 2 XCSR) will be set prior to issuing each I/O data transfer command.

E. I/O TIMEOUT

If a terminating interrupt is not received 3 minutes after initiating an I/O operation, the program will be aborted and the user informed. This time is approximate, calibrated on a PDP-11/45 with no other user programs running.

F. INSTRUCTION DESCRIPTIONS

```
READ  (D256 INTO RDIO) () = default values
READ  v (INTO RDIO)
READ  v INTO v
```

This instruction will cause a data transfer to occur between the device connected to this DL11, and the memory area specified by the second variable. The first variable contains the byte count for the transfer.

```
WRITE  (D256 FROM WRIO)
WRITE  v (FROM WRIO)
WRITE  v FROM v
```

The description for the Read Instruction applies to the Write except for the direction of data flow.

BREAK v

This instruction causes a break (spacing condition) on the line connected to this DL11. The duration of the break is determined by V which specifies the number of character times to hold the spacing condition.

CRESET

This instruction clears all the bits in the receive control register RCSR and transmit control register XCSR.

NOWAIT

Standard implementation - see section D.1.1.

WAIT

Standard implementation - see section D.1.1.

CALL

Used to initiate a call thru a MODEM from a DL11E. Sets Data Terminal Ready, waits for Carrier Detected.

D.4 - DL11 Device Routine (Cont'd)

LISTEN

Used to wait for an incoming call. Test the Ring Indicator and does not proceed until it is detected.

ANSWER

Used to answer an incoming call to a DL11E by way of a MODEM. Sets Data Terminal Ready, waits for Carrier Detected.

HANGUP

Used to terminate a call. Lowers Request To Send, delays 15 milliseconds then lowers Data Terminal Ready.

SEND

Sets up line for transmit. Does not cause any data transfer. Raises Request To Send, then waits for Clear To Send.

RECV

Sets up line for data reception. Does not cause any data transfer. Lowers Request To Send.

RDRON

Sets Bit 0 of RCSR which turns on the paper tape run relay when so attached thru a DL11-A and DL11-C.

RDROFF

Clears Bit 0 of RCSR which turns off the paper tape run relay where applicable.

STATUS

Standard implementation - see section D.1.1.

COUNTS

Standard implementation - see section D.1.1.

D.4.1 DL11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2.

The unique error messages which may occur in the use of this device routine are as follows:

DL11 TIMEOUT ON I/O

> An I/O operation was begun, but the terminating interrupt was not received. The time allowed for this interrupt to occur is approximately three minutes when the program is run on the PDP-11/45 and no other programs are running.

DL11 ERROR: ddd,ddd,etc.

> The codes ddd represent mnemonics which are printed to indicate which error bits were set. The message may contain from one to three codes. The following mnemonics may be printed in this message:
>
> PAR = Received Data Parity Error
> FRM = Framing Error
> OVR = Overrun Error

D.4.2 DL11 SAMPLE PROGRAMS

1. Wait for the phone to ring, answer it, send a message to the calling station, then echo back each character received until a 'rubout' is detected. After a 'rubout' is detected disconnect and terminate the program.

```
*FILL BUF WITH *READY FOR DL11 ECHO TEST
*ENTER 1
?ASGN DEV: DL11

>0010  LISTEN
>0020  ANSWER
>0030  SEND
>0040  WRITE D26 FROM BUF
>0050  RECV
>0060  READ 1 INTO TM00
>0070  IF TM00 = 377 GOTO 110
>0080  SEND
>0090  WRITE 1 FROM TM00
>0100  GOTO 50
>0110  HANGUP
```

```
>0120  END
DONE

*RUN 1
```

2. Provide a program compatible with sample number 1, that will initiate a call, receive the 26 byte message and print it. It should then transmit, the ASCII character set one character at a time and print the echoed character string. This should be followed by a 'rubout' and a disconnect.

```
*ENTER 1
?ASGN DEV: DL11

>0010  CALL
>0020  RECV
>0030  READ D26
>0040  PRINT D26 AT RDIO IN ASCII
>0050  LOAD TM00 WITH 40
>0060  LOAD SRC WITH TM01
>0070  LOAD DST WITH RDIO
>0080  LOAD NBR WITH 1
>0090  SEND
>0100  WRITE 1 FROM TM00
>0110  RECV
>0120  READ 1 INTO TM01
>0130  MOVE
>0140  INCR DST
>0150  INCR TM00
>0160  IF TM00 < 140 GOTO 90
>0170  PRINT D64 AT RDIO IN ASCII
>0180  LOAD WRIO WITH 377
>0190  SEND
>0200  WRITE 1
>0210  HANGUP
>0220  END
>DONE

*RUN 1
```

3. By using the Special Operation bit in the OPSW, use the Maintenance Mode for data loopback, do a simultaneous Read and Write, and then verify the data.

```
*ENTER 1
?ASGN DEV: DL11

>0010  FILL D256 AT WRIO WITH RANDOM
>0020  NOWAIT
>0030  READ
>0040  WRITE
```

D.4 - DL11 Device Routine (Cont'd)

```
>0050  WAIT
>0060  VERIFY D256 AT RDIO WITH WRIO
>0070  END
>DONE

*OPSW1 100200
*RUN   1
```

D.5 DQ11 NPR SYNCHRONOUS LINE INTERFACE

PRESET ADDRESS - 160030
PRESET INTERRUPT VECTOR - 300
PRESET BUS REQUEST - 5,5 (READ/WRITE)

A. REGISTER NAMES RECOGNIZED BY ALL INSTRUCTIONS

RCSR
TCSR
RERR
REG

B. REGISTER NAMES RECOGNIZED BY SELREG INSTRUCTION

RBAP
RCCP
TBAP
TCCP
RBAS
RCCS
TBAS
TCCS
CDET
SYNC
MISC
TBUF
SEQ
RBCC
TBCC
POLY

C. MPG INSTRUCTIONS SUPPORTED

SELREG
SELSEQ
READ
WRITE
CRESET
NOWAIT
WAIT
NOIDLE
IDLE
CALL
LISTEN
ANSWER
HANGUP
SEND
RECV
FDUPLX
HDUPLX
STATUS
COUNTS

D. STATISTICAL INFORMATION

BYTES:	Read, Written
CMNDS:	Read, Write, Misc.
INTERRUPTS:	Read, Write
ERRORS:	Vertical Redundancy Check (VRC), Block Check Character (BCC), Rcvr Non-Existent Mem (RNEM) Xmtr Non-Existent Mem (TNEM) Rcvr and Xmtr Latency (RLAT, TLAT), Rcvr and Xmtr Clock Loss (RCLK, TCLK), Data

E. OPSW SPECIAL OPERATION BIT SUPPORT

When Bit 7 (SPOPER) of the OPSW is set, the Test Loop Bit (Bit 3 MISC) will be set prior to issuing each I/O data transfer command.

F. I/O TIMEOUT

If a terminating interrupt is not received 3 minutes after initiating an I/O operation, the program will be aborted and the user informed. This time is approximate, calibrated on a PDP-11/45 with no other user programs running.

G. INSTRUCTION DESCRIPTIONS

SELREG v

This instruction enhances the low level language of MPG, by allowing the reference of DQ11 Secondary Registers by name. The names recognized by this instruction are listed in section C, and cause the appropriate 4 bit code to be loaded in the REG/ERR Register.

SELSEQ v

This instruction simplifies program access to the two 16 by 16 registers that are available when the Protocol option is added. The variable v should be the register number in decimal or octal. If it is a tag such as TM00, the contents will be used as the register number. This instruction will place that number in the proper bits in RCSR.

READ (D256 INTO RDIO) () = default values
READ v (INTO RDIO)
READ v INTO v

This instruction will cause a data transfer to occur between the device connected to this DQ11, and the memory area specified by the second variable. The first variable contains the byte count for the transfer.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

The description for the Read instruction applies to the Write, except for the direction of data flow.

CRESET

Sets Master Clear, clears all Secondary and Sequence Control Registers, then sets bits per character equal to 8.

NOWAIT

Standard implementation - see section D.1.1.

D.5 - DQ11 Device Routine (Cont'd)

WAIT

Standard implementation - see section D.1.1.

NOIDLE

Resets the Idle Mode bit in TCSR.

IDLE

Sets the Idle Mode bit in TCSR allowing the transmission of IDLE characters whenever Tx GO is zero.

CALL

Used to initiate a call thru a MODEM from a DQ11. Sets Data Terminal Ready, waits for Data Set Ready.

LISTEN

Used to wait for an incoming call. Tests the Ring Indicator and does not proceed until it is detected.

ANSWER

Used to answer an incoming call to a DQ11 by way of a MODEM. Sets Data Terminal Ready, waits for Data Set Ready.

HANGUP

Used to terminate a call. Lowers Request to Send, delays 15 milliseconds, then lowers Data Terminal Ready.

SEND

Sets up line for transmit. Does not cause any data transfer. Raises Request to Send, then waits for Clear to Send.

RECV

Sets up line for data reception. Does not cause any data transfer. Lowers Request to Send.

FDUPLX

Places the DQ11 in a Full Duplex Mode. This is the initial mode of a DQ11 after a CRESET or a RESET.

HDUPLX

Places the DQ11 in a Half Duplex Mode.

STATUS

Standard implementation - see section D.1.1.

COUNTS

Standard implementation - see section D.1.1.

D.5.1 DQ11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2.

The unique error messages which may occur in the use of this device routine are as follows:

DQ11 TIMEOUT ON I/O

An I/O operation was begun, but the terminating interrupt was not received. The time allowed for this interrupt to occur is approximately three minutes when the program is run on the PDP-11/45 and no other programs are running.

DQ11 ERROR: ddd,ddd,etc.

The codes ddd represent mnemonics which are printed to indicate which error bits were set. The message may contain from one to eight codes. The following mnemonics may be printed in this message:

TCLK = Transmitter Clock Loss Error
RCLK = Receiver Clock Loss Error
TLAT = Transmitter Latency Error
RLAT = Receiver Latency Error
TNEM = Transmitter Non-existent Memory Error
RNEM = Receiver Non-existent Memory Error
RBCC = Receiver Block Check Error
RVRC = Receiver Vertical Redundancy Error

D.5.2 DQ11 SAMPLE PROGRAMS

1. Test the DQ11 using the Test Loop bit to cause data wraparound. Set the OPSW to loop the program and turn on the Test Loop bit. Use a printable character as a sync character ([).

```
*ENTER 1
?ASGN DEV: DQ11

>0010  CRESET
>0020  SELREG SYNC
>0030  MOVE COMO TO REG
>0040  FILL D16 AT BF02 WITH 20040
>0050  IDLE
>0060  NOWAIT
>0070  READ D16 INTO BF02
>0080  WRITE D16 FROM BF00
>0090  WAIT
>0100  NOIDLE
>0110  PRINT D16 AT BF02 IN ASCII
>0120  END
>DONE

*FILL COMO WITH *[[
*FILL BF00 WITH *[[DQ TST MSG 1
*OPSW 40214
*RUN 1
```

2. Modify sample program number 1 to have the DQ11 strip off all sync characters as they are received. Note- Read byte count must be decreased.

```
*MODIFY 1

ENTER STMNT'S

>45    SET RCSR BIT 1
>70    READ D14 INTO BF02
>DONE
*RUN 1
```

3. Test the automatic switchover from primary to secondary write registers and back again. Use the Test Loop bit as in program 1.

```
*ENTER 1
?ASGN DEV: DQ11

>0010  CRESET
>0020  SELREG SYNC
>0030  MOVE COMO TO REG
>0040  FILL D64 AT BF04 WITH 20040
>0050  IDLE
>0060  NOWAIT
>0070  READ D64 INTO BF04
>0080  WRITE D16 FROM BF00
>0090  WRITE D16 FROM BF01
>0100  WRITE D16 FROM BF02
>0110  WRITE D16 FROM BF03
>0120  WAIT
>0130  NOIDLE
>0140  PRINT D64 AT BF04 IN ASCII
>0150  END
>DONE

*FILL COMO WITH *[[
*FILL BF00 WITH *[[DQ TST MSG 1
*FILL BF01 WITH *[[DQ TST MSG 2
*FILL BF02 WITH *[[DQ TST MSG 3
*FILL BF03 WITH *[[DQ TST MSG 4
*OPSW 200
*RUN 1
```

4. Modify Program 3 to test the switchover of the read registers while using only the primary write registers.

```
*MODIFY 1

ENTER STMNT'S

>70    READ D16 INTO BF04
>80    READ D16 INTO BF05
>90    WRITE D64 FROM BF00
>100   READ D16 INTO BF06
>110   READ D16 INTO BF07
>DONE

*RUN 1
```

5. For systems with the DQ11-BB (PROTOCOL) Option, display the contents of the Character Detection and Sequence Registers.

```
*ENTER 1
?ASGN DEV: DQ11

>0010  PRINT *CHARACTER DETECTION REGISTER CONTENTS
>0020  LOAD TM00 WITH 0
>0030  SELREG CDET
>0040  SELSEQ TM00
>0050  PRINT REG IN OCTAL
>0060  INCR TM00
>0070  IF TM00 < 20 GOTO 40
>0080  PRINT *
>0090  PRINT *SEQUENCE REGISTER CONTENTS
>0100  LOAD TM00 WITH 0
>0110  SELREG SEQ
>0120  SELSEQ TM00
>0130  PRINT REG IN OCTAL
>0140  INCR TM00
>0150  IF TM00 < 20 GOTO 120
>0160  END
>DONE

*RUN 1
```

D.6 RK11 DISK DEVICE ROUTINE

The RK11 Device Routine, whose filename is TRKAnm.MPG, supports the operation of RK05 and RK03 disk drives. Other types of disk drives may be utilized as long as they do not require special considerations in the manipulation of the device registers.

D.6.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

In reply to the "ASGN DEV:" message, two model names (RK11 and RK05) are acceptable and may be accompanied by a maximum of 16 unit numbers. For this device the unit numbers must be in the range of 0 thru 7.

- Interface Addresses

The following are the values preset for the RK11 and may be altered from the console terminal following the "ASGN DEV:" message:

Device Register Base Address = 177400
Interrupt Vector Address = 220
Bus Request Priority = 5

- Symbolic Register Names

The symbolic names listed below may be used to reference the RK11 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register.

Name	Displ	Description
RKDS	+0	Drive Status
RKER	+2	Error
RKCS	+4	Control and Status
RKWC	+6	Word Count
RKBA	+10	Bus Address
RKDA	+12	Disk Address
RKDB	+16	Data Buffer

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the RK11. Detailed explanations are listed in section D.6.2.

READ
WRITE
SEEK
WRLOCK
RDFMT
WRFMT
RDCK
WRCK
CRESET
DRESET
NOWAIT
WAIT
STEPUP
STEPDN
STATUS
COUNTS

- Information Words

The five words listed below are used to pass information between the user program and the device routine. The first three are related and are used to specify the location on the disk where the I/O operation is to begin. The last two words are used to pass information back to the user program. All words are initially preset to 0's.

CYL A one word location that contains the cylinder number in bits 0-7 that will be used in subsequent I/O operations. The contents of CYL and HEAD will be shifted to the appropriate bit positions, merged into one word with SECT, and then loaded into RKDA. This occurs for all I/O operations that require a disk address.

HEAD The word location that follows CYL and contains the value of the head number to be used (0 or 1).

SECT This word contains the sector number in bits 0-3. Located immediately following HEAD in memory.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, RDFMT, WRFMT, and WRCK.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement or the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ, RDFMT
BYTES WR:	WRITE, WRFMT
READ CMNDS:	READ, RDFMT
WRITE CMNDS:	WRITE, WRFMT
SEEK CMNDS:	SEEK
MISC CMNDS:	RDCK, WRCK, WRLOCK, DRESET, CRESET
DEV ERRORS:	All hardware errors
DATA ERRORS:	Invalid unit number errors and errors detected by the VERIFY statement.
INTERRUPTS:	Number of entries into the interrupt routine.

- OPSW Special Operation Bit Support

When bit 7 (SPOPER) of the OPSW is set, the Inhibit Increment (INH BA) bit in RKCS will be set prior to issuing each I/O command.

D.6.2 DESCRIPTION OF RK11 INSTRUCTIONS

The RK11 Device Routine supports execution of sixteen MPG language statements. For certain functions (READ, WRITE, RDFMT, WRFMT, SEEK, RDCK, and WRCK), the desired disk address must be loaded into CYL, HEAD, and SECT before performing those functions. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2. Note that if an odd byte count is supplied in any of the following instructions, it will effectively be decremented by 1 before being used. All I/O operations are performed with the Stop on Soft Error (SSE) bit set in the RKCS register.

```
READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v
```

This instruction transfers the number of bytes, indicated by the first v, from the disk to memory beginning at the memory location specified by the second v.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Transfer the number of bytes from memory to disk beginning at the memory location indicated by the second v.

SEEK

Performs a seek to the disk location specified by the current values of CYL, HEAD and SECT. This instruction does not terminate until the Search Complete Interrupt is received.

WRLOCK

Causes the current disk drive to be write protected.

RDFMT v INTO v

This is a read instruction but with the RKCS Format (FMT) bit set. Same as READ except that only two bytes per sector (Header Word) are transferred to memory and default operands are not supported.

WRFMT v FROM v

Same as the WRITE instruction but has the Format (FMT) bit set.

RDCK

This instruction issues the Read Check command on the sector addressed by CYL, HEAD, and SECT.

WRCK v AT v

Performs the Write Check command at the current sector with the number of data bytes specified by the first operand at the location specified by the second operand.

CRESET

A Control Reset command is performed by this instruction. No interrupts are involved.

DRESET

The Drive Reset command is issued to the current unit with this instruction. This instruction does not terminate until Search Complete Interrupt is received.

The following instructions do not perform any I/O functions on the RK11.

NOWAIT

Standard implementation - see section D.1.1.

WAIT

Standard implementation - see section D.1.1.

STEPUP v

Easy incrementing of the values of CYL, HEAD, and SECT is provided with this statement. The variable v is a count of the number of sectors that these values are to be incremented. Incrementing of the HEAD value will occur when the value of SECT exceeds decimal 11 and incrementing of CYL will occur when HEAD exceeds 1. When the values of the three words exceed 202,1,11, wrap around to 0,0,0 will occur. For example, if v = 1, the result will be the address of the next sector on disk; if v = decimal 12 (D12), the new address will be the same sector but on the next head; if v = decimal 24, the resulting address will be the same sector and head but on the next cylinder.

Note that this instruction (and STEPDN) will operate with invalid values in CYL, HEAD, and SECT upon execution. Regardless of their initial contents, the contents of all three words will be valid when this instruction completes. This allows the use of "FILL 6 AT CYL WITH RANDOM" statements followed by "STEPUP 0" to generate random disk addresses.

STEPDN v

This statement is essentially the same as STEPUP but provides a decrementing facility for the three values. When wrap around occurs, it will go from 0,0,0 to 202,1,11.

Since the values of CYL, HEAD, and SECT are preset to zeroes, STEPUP or STEPDN may be used to set these words to their desired initial values with one statement. For example, STEPUP D496 would result in the decimal values CYL = 20, HEAD = 1, and SECT = 4.

STATUS

Standard implementation - see section D.1.1.

Included with the standard display is the display of the current contents of the words CYL, HEAD, and SECT. These values do not necessarily reflect the current position of the disk, but merely the contents of the three words.

COUNTS

Standard implementation - see section D.1.1.

D.6.3 RK11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. Note that the contents of the device registers and the CYL, HEAD, and SECT words at the time of the error will be displayed for all errors except the "INV UNIT #" error.

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

INV DEV I.D.

When processing a search complete type of interrupt, the drive identity in bits 13 thru 15 of RKDS did not match the current unit number.

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erroneous unit number. Increments the data error count.

STATUS ERROR: ddd,ddd,etc.

This message indicates that an error status bit has been detected in the RK11 device registers. The codes ddd identify which bits were found with a maximum of five being listed. The following are the codes possible for the ddd fields:

DRY = Drive Not Ready
DPL = Drive Power Low
DRU = Drive Unsafe
SIN = Seek Incomplete
DRE = Drive Error
OVR = Overrun
WLO = Write Lock Out
SKE = Seek Error
PGE = Programming Error
NXM = Non-Existent Memory
DLT = Data Late
TME = Timing Error
NXD = Non-Existent Disk
NXC = Non-Existent Cylinder
NXS = Non-Existent Sector
CSE = Checksum Error
WCE = Write Check Error

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately 22 seconds when on the PDP-11/45 and with no other user programs executing.

T/O ON CRESET

This error, which indicates that Ready (RDY) did not set within a few milliseconds after issuing the Control Reset command, may occur under either of two conditions. The first is when the CRESET statement is issued and the other is for the initiation of any other I/O function. When initiating any function, the device registers are tested for error conditions left over from previous functions. If present, the Control Reset command is issued to clear these conditions and timeout may occur in this situation.

UNEXP SRCH COMP INT

An unexpected search complete interrupt has occurred. This error may occur from either of two conditions. The first is when this interrupt occurs on an I/O operation that should not result in this type of interrupt (READ, WRITE, etc). The second condition is when the search complete interrupt occurs before the normal interrupt on the SEEK and DRESET functions.

UNIT NOT RDY

Immediately prior to initiating an I/O operation, the Drive Ready bit (DRY) in the RKDS register was found to be reset.

D.6.4 RK11 SAMPLE PROGRAMS

The following are sample RK11 programs that perform the functions indicated in their descriptions:

1. This program will start at cylinder 0, head 0, sector 0 of unit 2 and write a sector with random data, read it back, and then verify the read data with the write data. It will then increment to the next sector and repeat the operation with different random data. This entire sequence will be repeated 20 times.

```
*ENTER 1
?ASGN DEV: RK11,2
?RDIO = 256 / 512
?WRIO = 256 / 512
?DEV REG = 177400 /
?INT VEC = 000220 /
?BUS REQ = 5   /
ENTER STMNT'S

>0010   LOAD TM00 WITH D20
>0020   FILL D512 AT WRIO WITH RANDOM
>0030   WRITE D512
>0040   READ D512
>0050   VERIFY D512 AT RDIO WITH WRIO
>0060   STEPUP 1
>0070   DECR TM00
>0080   IF TM00 > 0 GOTO 20
>0090   END
>DONE

*RUN 1
```

D.6 - RK11 Device Routine (Cont'd)

2. The following program is an example of using the low level language and programming at the interface level without the use of high level I/O statements. An equivalent program could be written using the SEEK instruction and it would automatically include statistic gathering and error reporting.

   The purpose of this program is to initially issue a Control Reset command to unit 0 followed by a seek to cylinder 8, head 0, sector 0. When this seek completes, then issue a seek to cylinder 128, head 0, sector 0 and wait for it to complete. The dual seeks will be repeated ten times. Only the pertinent messages are shown in the example.

```
*ENTER 1
?ASGN DEV: RK05
?RDIO = 256 / 0
?WRIO = 256 / 0

>0010    LOAD RKCS WITH 1
>0020    LOAD TM10 WITH 12
>0030    LOAD RKDA WITH 400
>0040    LOAD RKCS WITH 11
>0050    IF RKCS BIT 7 CLEAR GOTO 50
>0060    IF RKDS BIT 6 CLEAR GOTO 60
>0070    LOAD RKDA WITH 10000
>0080    LOAD RKCS WITH 11
>0090    IF RKCS BIT 7 CLEAR GOTO 90
>0100    IF RKDS BIT 6 CLEAR GOTO 100
>0110    DECR TM10
>0120    IF TM10 > 0 GOTO 30
>0130    END
>DONE

*RUN 1
```

3. The following example consists of two programs which will copy the data from one disk to another. Data will be copied in four sector blocks from unit 0 to unit 1. Since each program can process only one device at a time, two programs that synchronize themselves are needed.

```
*ENTER 1 AS RKCPY1
?ASGN DEV: RK11,0
?RDIO = 256 / 0
?WRIO = 256 / 0

>0010    LOAD COMO WITH 0
>0020    READ D2048 INTO FREE
>0030    LOAD COMO WITH 1
>0040    LETGO
>0050    IF COMO = 1 GOTO 40
>0060    STEPUP 4
>0070    IF CYL > 0 GOTO 20
>0080    IF HEAD > 0 GOTO 20
>0090    IF SECT > 0 GOTO 20
>0100    END
>DONE

*ENTER 2 AS RKCPY2
?ASGN DEV: RK11,1
?RDIO = 256 / 0
?WRIO = 256 / 0

>0010    IF COMO = 1 GOTO 40
>0020    LETGO
>0030    GOTO 10
>0040    WRITE D2048 FROM FREE
>0050    STEPUP 4
>0060    LOAD COMO WITH 0
>0070    IF CYL > 0 GOTO 20
>0080    IF HEAD > 0 GOTO 20
>0090    IF SECT > 0 GOTO 20
>0100    END
>DONE

*RUN 1,2
```

D.7 TC11 DECTAPE DEVICE ROUTINE

The TC11 Device Routine, whose filename is TTCAnm.MPG, supports the execution of I/O operations on the TU56 DECtape.

D.7.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

In reply to the "ASGN DEV:" message, two model names (TC11 and TU56) are acceptable and may be accompanied by a maximum of 16 unit numbers. For this device the unit numbers must be in the range of 0 thru 7.

- Interface Addresses

The following are the values preset for the TC11 and may be altered from the console terminal following the "ASGN DEV:" message:

Device Register Base Address = 177340
Interrupt Vector Address = 214
Bus Request Priority = 6

- Symbolic Register Names

The symbolic names listed below may be used to reference the TC11 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register.

Name	Displ	Description
TCST	+0	Control and Status
TCCM	+2	Command
TCWC	+4	Word Count
TCBA	+6	Bus Address
TCDT	+10	Data Buffer

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the TC11. Detailed explanations are listed in section D.7.2.

FWD
REV
READ
WRITE
RDNUM
RDALL
WRALL
WRTM
STOP
STPALL
NOWAIT
WAIT
STATUS
COUNTS

- Information Words

The three words listed below are used to pass information between the user program and the device routine and are initially preset to 0's.

BLK A one word location used by the user program to specify to the device routine the number of the DECtape block on which the I/O operation is to begin. The acceptable range of values for this word is decimal 0 - 577. The commands that utilize the contents of BLK are:

READ
WRITE
RDALL
WRALL

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, RDALL, WRALL, RDNUM, and WRTM.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement or the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ, RDALL, RDNUM
BYTES WR:	WRITE, WRALL, WRTM
READ CMNDS:	READ, RDALL, RDNUM
WRITE CMNDS:	WRITE, WRALL, WRTM
MISC CMNDS:	STOP, STPALL
DEV ERRORS:	All hardware errors
DATA ERRORS:	Invalid unit and BLK number errors and errors detected by the VERIFY statement.
INTERRUPTS:	Number of entries into the interrupt routine.

- OPSW Special Operation Bit Support

  Bit 7 of the OPSW (SPOPER) is not supported by the TC11 device routine and its setting has no effect upon device routine operation. Support for the TC11 maintenance bit was not included due to the complexity of providing meaningful functions with the variety of operation needed. MPG's low level language lends itself to this type of application and does not require seldom used code in the device routine.

D.7.2 DESCRIPTION OF TC11 INSTRUCTIONS

The TC11 Device Routine supports execution of twelve MPG language statements. For certain functions (READ, WRITE, RDALL, and WRALL), the desired block number must be loaded into BLK before performing those functions. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2. Note that if an odd byte count is supplied in any of the following instructions, it will effectively be decremented by 1 before being used.

FWD

The tape direction of all subsequent commands is controlled with this statement. FWD, which is the preset direction, indicates that all subsequent READ, WRITE, RDALL, WRALL, RDNUM, and WRTM commands are to be performed with the tape moving in a forward direction during data transfers. Block searching will still be done in whichever direction is needed to arrive at the specified block. This command does not cause any tape movement.

REV

Similar to the FWD instruction but indicates that all data transfers are to be done while moving tape in the reverse direction. FWD/REV also controls the initial tape direction when searching for a specified block.

READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v

This instruction transfers the number of bytes, specified by the first v, from DECtape to memory beginning at the memory location indicated by the second v. Data will be transferred starting at the block whose number is stored at BLK with the tape direction controlled by FWD/REV. Searching for the block will occur, if needed.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Same as the READ except that the number of bytes will be written to tape from memory.

RDNUM v INTO v

Reads the block number of the next block into memory at the location indicated by the second v. Tape direction is controlled by FWD/REV and block searching does not occur. The first v is a byte count and in effect controls how many block numbers will be read. If 2, the block number of only the first block is read. If 6 for example, the numbers of the next 3 blocks will be stored in memory.

RDALL v INTO v

This instruction is very similar to the READ statement. However, since the TC11 Read All command includes two checksum words and transfers 18 bits of data at a non-NPR level, special considerations had to be taken. Block searching and data transfers take place without interrupts. The entire processor is dedicated to this instruction during the search and during data transfers. I/O operations on other devices should not be in progress when initiating this command. This is due to the tight timing requirements imposed upon data servicing. The time used to service another device's interrupt could possibly result in errors on this command. Also, since the data consists of eighteen bits, difficulty was encountered in trying to stuff them into a PDP-11 sixteen bit word. The method chosen was to place data bits 16 and 17 into bits 0 and 1 of the first word of a pair of memory words and data bits 0 thru 15 in the second word. This results in two memory words for every eighteen bit word. Therefore, the byte count for the RDALL (and WRALL) command must be doubled. The reading of one block with the RDALL command results in the transfer of 258 eighteen bit words. With a sixteen bit word this would result in a byte count of 516. However, due to the double word storage format already mentioned, the RDALL command must specify a byte count of 1032 in order to read all data in one block. Byte count totals will reflect four bytes read for every eighteen bit word.

WRALL v FROM v

Essentially the same as the RDALL instruction except for the direction of data transfer. The mode of operation and all special considerations pertaining to the RDALL statement apply to WRALL. Data is expected to have bits 16 and 17 in bits 0 and 1 of the first word of a pair of words with bits 0 thru 15 in the second word.

WRTM v FROM v

This instruction is used to write the timing track on DECtapes. FWD/REV controls the tape direction and block searching is not performed. Since this is a non-NPR function and is performed without interrupts, the restrictions for RDALL/WRALL concerning processor control and the operation of other devices apply to WRTM. Since the data words are sixteen bits, the byte count is handled in the same manner as for WRITE. Data transfers begin occurring as soon as READY sets.

STOP

This command stops tape movement on the current unit.

STPALL

Used to stop tape movement on all units.

NOWAIT

Standard implementation - see section D.1.1.

The NOWAIT mode has no effect upon the RDALL, WRALL, and WRTM instructions since they always operate in the WAIT mode.

WAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1. The first display is the values when the last interrupt occurred or at the termination of the last non-NPR function (RDALL, WRALL, or WRTM).

Included with the standard display is the display of the current contents of the word BLK. This value does not necessarily reflect the current position on tape, but merely the contents of this word.

COUNTS

Standard implementation - see section D.1.1.

D.7.3 TC11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. Note that the contents of the device registers and the BLK word at the time of the error will be displayed for all errors except the "INV UNIT #" error.

The unique error messages, which may occur in the use of this device routine, are as follows:

BLK SEARCH ERROR

While searching for the specified block prior to performing an I/O operation, tape direction has been reversed six times without finding the block. Possibly due to an invalid block number on tape.

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

INV BLK #

Immediately prior to initiating an I/O operation that requires block searching, the contents of the word BLK was not in the range of decimal 0 - 577. Since block searching is a programmed function, it would be futile to proceed. Increments the data error count.

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erroneous unit number. Increments the data error count.

STATUS ERROR: dddd,dddd,etc.

This message indicates that an error status bit has been detected in the TC11 device registers. The codes dddd identify which bits were found with a maximum of four being listed. The following are the codes possible for the dddd fields:

ENDZ = End Zone
PARE = Parity Error
MKTE = Mark Track Error

ILOP = Illegal Operation
SELE = Selection Error
BLKM = Block Missed
DATM = Data Missed
NEXM = Non-Existent Memory

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately two minutes and 34 seconds when on the PDP-11/45 and with no other user programs executing.

D.7.4 TC11 SAMPLE PROGRAMS

The following are sample TC11 programs that perform the functions indicated in their descriptions:

1. Read block 0 of unit 1 in a forward direction and display its contents in octal on the line printer.

```
*ENTER 1
?ASGN DEV: TC11 1
?RDIO = 256 / 512
?WRIO = 256 / 0
?DEV REG = 177340 /
?INT VEC = 000214 /
?BUS REQ = 6   /
ENTER STMNT'S

>0010  READ D512
>0020  PRINT D512 AT RDIO IN OCTAL
>0030  END
>DONE

*ASSIGN LP11 TO LIST
*RUN 1
```

2. Using the above program, modify it so that the read is performed in the reverse direction on block number octal 20.

```
*MODIFY 1
ENTER STMNT'S

>02    LOAD BLK WITH 20
>05    REV
>DONE

*RUN 1
```

3. Starting at octal block number 100 on unit 2, write 512 bytes of random data, read it back, and compare it to the data that was written. Then, point to the next block and repeat the operation. Do this test for 15 blocks and repeat the entire test on units 3, 1, and 4 in that order. Only pertinent messages are shown in this example.

```
*ENTER 16 AS TCRDWR
?ASGN DEV: TU56,2,3,1,4
?RDIO = 256 / 0
?WRIO = 256 / 512

>0010  LOAD BLK WITH 100
>0020  LOAD TM08 WITH D15
>0030  FILL D512 AT WRIO WITH RANDOM
>0040  WRITE D512
>0050  READ D512 INTO FREE
>0060  VERIFY D512 AT FREE WITH WRIO
>0070  INCR BLK
>0080  DECR TM08
>0090  IF TM08 > 0 GOTO 30
>0100  END
>DONE

*RUN 16
```

4. Read the two checksum words and all data words in block number 1 on unit 7 as eighteen bit words.

```
*ENTER 9
?ASGN DEV: TC11,7
?RDIO = 256 / 0
?WRIO = 256 / 0

>0010  INCR BLK
>0020  RDALL D1032 INTO FREE
>0030  END
>DONE

*RUN 9
```

D.8 TM11 MAGTAPE DEVICE ROUTINE

The TM11 Device Routine, whose filename is TTMAnm.MPG, supports the execution of I/O operations on the TU10 magtape.

D.8.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

In reply to the "ASGN DEV:" message, two model names (TM11 and TU10) are acceptable and may be accompanied by a maximum of 16 unit numbers. For this device the unit numbers must be in the range of 0 thru 7.

- Interface Addresses

The following are the values preset for the TM11 and may be altered from the console terminal following the "ASGN DEV:" message:

Device Register Base Address = 172520
Interrupt Vector Address = 224
Bus Request Priority = 5

- Symbolic Register Names

The symbolic names listed below may be used to reference the TM11 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register.

Name	Displ	Description
MTS	+0	Status
MTC	+2	Command
MBRC	+4	Byte/Record Counter
MCMA	+6	Current Memory Address
MTD	+10	Data Buffer
MTRD	+12	Read Lines

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the TM11. Detailed explanations are listed in section D.8.2.

READ
WRITE
WREIRG
WREOF
SPFWD
SPREV
REWIND
OFFLIN
CRESET
EVEN
ODD
BPI
NOWAIT
WAIT
STATUS
COUNTS

- Information Words

The six words listed below are used to pass information between the user program and the device routine.

RDRB The contents of this word specifies the number of times that MPG will attempt to re-read a block on which a CRE or PAE error was detected. Preset value is 1.

WRRB Essentially the same as RDRB but specifies the number of retries for WRITE, WREIRG, and WREOF. Preset value is 3.

EOF This word is cleared to zeroes before each I/O operation and then set to a value of 1 if the End Of File bit was set in the MTS register when the terminating interrupt was processed. This allows user detection of EOF's since they are not treated as error conditions. Preset value is 0.

EOT Utilized in the same manner as EOF but reflects the status of the End Of Tape bit in the MTS register. Preset value is 0.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, and WREIRG.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement or the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ
BYTES WR:	WRITE, WREIRG
READ CMNDS:	READ
WRITE CMNDS:	WRITE, WREIRG, WREOF
MISC CMNDS:	SPFWD, SPREV, REWIND, OFFLIN, CRESET
RD ROLLBACKS:	READ
WR ROLLBACKS:	WRITE, WREIRG, WREOF
# OF EOF'S:	Number of EOF's encountered during program execution.
# OF EOT'S	Number of EOT's encountered during program execution.
DEV ERRORS:	All hard errors and unrecoverable rollbacks
DATA ERRORS:	Invalid unit number and invalid BPI errors and errors detected by the VERIFY statement.
INTERRUPTS:	Number of entries into the interrupt routine.

- OPSW Special Operation Bit Support

Bit 7 of the OPSW (SPOPER) is not supported by the TM11 device routine and its setting has no effect upon device routine operation.

D.8.2 DESCRIPTION OF TM11 INSTRUCTIONS

The TM11 Device Routine supports execution of sixteen MPG language statements. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2.

```
READ  (D256 INTO RDIO)
READ  v (INTO RDIO)
READ  v INTO v
```

This instruction transfers the number of bytes, specified by the first v, from magtape to memory beginning at the memory location indicated by the second v. Note that when reading an EOF block with this command, the BYTES RD count will not be incremented.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Same as the READ except that the number of bytes will be written to tape from memory. If necessary to perform write rollback on this command, the Device Routine will utilize the Write with Extended Interrecord Gap command for the writes performed during rollback.

WREIRG v FROM v

This instruction is similar to the WRITE statement except that only one statement format is supported and it uses the TM11 Write with Extended Interrecord Gap command.

WREOF

This command is used to write an End of File record on tape at its current position.

SPFWD v

The number of records specified by the operand v will be spaced over in a forward direction.

SPREV v

The number of records specified by the operand v will be spaced over in a reverse direction.

REWIND

Use of this instruction causes the tape to be rewound to the BOT marker. This instruction does not terminate until BOT is reached and the second interrupt is processed.

OFFLIN

Causes the tape to be rewound and then unloaded.

CRESET

This instruction is used to perform a control reset to the controller and tape units. Sets the Power Clear bit in the MTC register.

BPI v

Used to set the proper Bits Per Inch values into the DEN 8 and DEN 5 bits in the MTC register. Four values are valid for v and are as follows: (Preset to 11)

```
   00 = 200 BPI (7 Level)
   01 = 556 BPI (7 Level)
   10 = 800 BPI (7 Level)
   11 = 800 BPI (9 Level or core dump
                  mode for 7 Level)
```

EVEN

Sets the parity mode to even for reading and writing. Affects all subsequent I/O operations.

ODD

Similar to EVEN but sets the parity mode to odd. ODD is the preset mode.

NOWAIT

Standard implementation - see section D.1.1.

WAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

Included with the standard display is the display of the current contents of the EOF and EOT words.

COUNTS

Standard implementation - see section D.1.1.

D.8.3 TM11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. Note that the contents of the device registers and the EOF and EOT words will be displayed for all errors except the "INV BPI" and "INV UNIT #" errors.

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

INV BPI VALUE

The operand supplied with the BPI statement was not one of the values which are valid for the TM11. The valid octal values for the TM11 are 00, 01, 10, and 11. Increments the data error count.

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erroneous unit number. Increments the data error count.

ROLLBACK EXH.

This error indicates that the TM11 device routine has failed to sucessfully perform a read or write type of operation. The number of retries attempted is controlled by the contents of RDRB and WRRB. This error will not occur if a rollback type of error occurs and the number of retries specified by the applicable word is zero. In this case only the STATUS ERROR message will be displayed.

STATUS ERROR: ddd,ddd,etc.

This message indicates that an error status bit has been detected in the TM11 device registers. The codes ddd identify which bits were found with a maximum of five being listed. The following are the codes possible for the ddd fields:

```
ILC = Illegal Command
CRE = Cyclic Redundancy Error
PAE = Parity Error
BGL = Bus Grant Late
RLE = Record Length Error
BTE = Bad Tape Error
NXM = Non Existent Memory
SLR = Select Remote
WRL = Write Lock
```

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur when on the PDP-11/45 and with no other user programs executing is as follows:

```
WREOF, OFFLIN          = 22 Seconds
READ, WRITE, WREIRG    = 1 Minute 6 Seconds
SPFWD, SPREV           = 2 Minutes 12 Seconds
REWIND                 = 7 Minutes
```

D.8.4 TM11 SAMPLE PROGRAMS

The following are sample TM11 programs that perform the functions indicated in their descriptions:

1. Beginning on unit 2, write 100 blocks of 1000 bytes of random data, rewind the tape, read all blocks, and then rewind the tape again. Repeat the program on units 4, 5, and 6.

```
*ENTER 6
?ASGN DEV: TM11,2,4-6
?RDIO = 256 / 1000
?WRIO = 256 / 1000
?DEV REG = 172520 /
?INT VEC = 000224 /
?BUS REQ = 5    /

ENTER STMNT'S

>0010   LOAD TM04 WITH D100
>0020   FILL D1000 AT WRIO WITH RANDOM
>0030   WRITE D1000
>0040   DECR TM04
>0050   IF TM04 > 0 GOTO 20
>0060   REWIND
>0070   LOAD TM05 WITH D100
>0080   READ D1000
>0090   DECR TM05
>0100   IF TM05 > 0 GOTO 80
>0110   REWIND
>0120   END
>DONE

*RUN 6
```

2. Write 4096 byte blocks of all one bits to end of tape on unit 1. Terminate the tape by writing two EOF's and doing a rewind. Read the tape forward until detecting the two EOF's and then rewind the tape.

```
*ENTER 9
?ASGN DEV: TU10,1
?RDIO = 256 / 0
?WRIO = 256 / 0

>0010   FILL D4096 AT FREE WITH 177777
>0020   WRITE D4096 FROM FREE
>0030   IF EOT = 0 GOTO 20
>0040   WREOF
>0050   WREOF
>0060   REWIND
>0070   READ D4096 INTO FREE
```

```
>0080   IF EOF = 0 GOTO 70
>0090   INCR TM03
>0100   IF TM03 <> 2 GOTO 70
>0110   REWIND
>0120   END
>DONE

*RUN 9
```

3. Test the spacing capabilities of all eight TU10 tape units by initially rewinding the tape and writing a 2000 byte block of random data whose data is saved. Next, write another block of different random data, space reverse two blocks, read the data in the first block and compare it to the data that was saved. Space forward over the block last written and save its write data. Repeat this procedure until End Of Tape is detected at which time, read and compare the final block followed by rewinding the tape.

```
ENTER 8 AS GRINDR
?ASGN DEV: TU10,0-7
?RDIO = 256 / 2000
?WRIO = 256 / 2000

>0010   REWIND
>0020   FILL D2000 AT WRIO WITH RANDOM
>0030   WRITE D2000
>0040   MOVE D2000 AT WRIO TO FREE
>0050   FILL D2000 AT WRIO WITH RANDOM
>0060   WRITE D2000
>0070   SPREV 2
>0080   READ D2000
>0090   VERIFY D2000 AT RDIO WITH FREE
>0100   IF EOT = 1 GOTO 130
>0110   SPFWD 1
>0120   GOTO 40
>0130   READ D2000
>0140   VERIFY D2000 AT RDIO WITH WRIO
>0150   REWIND
>0160   END
>DONE

*RUN 8
```

D.9 LP11/LS11/LV11 PRINTERS' DEVICE ROUTINE

The LP11/LS11/LV11 Device Routine, whose filename is TLPAnm.MPG, supports the execution of I/O operations on the LP11, LS11, and LV11 printers.

D.9.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

In reply to the ASGN DEV: message, three model names (LP11, LS11, and LV11) are acceptable. Unit numbers are not needed and are not interrogated by the Device Routine. However, up to sixteen may be entered and will result in the program being repeated on the current printer for each unit number entered. Effectively acts as a pass count.

- Interface Addresses

The following are the values preset for all three printers and may be altered from the console terminal following the ASGN DEV: message:

Device Register Base Address = 177514
Interrupt Vector Address = 200
Bus Request Priority = 4

- Symbolic Register Names

The symbolic names listed below may be used to reference the printers' device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register. Note that symbolic names for all three printers are always supported, even though they may reference the same location.

Name	Displ	Description
LPS	+0	LP11 Control and Status
LPB	+2	LP11 Data Buffer
LPCS	+0	LP11 Control and Status
LPDB	+2	LP11 Data Buffer
LSCS	+0	LS11 Control and Status
LSDB	+2	LS11 Data Buffer
LVCS	+0	LV11 Control and Status
LVDB	+2	LV11 Data Buffer

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the LP11, LS11, and LV11 printers. Detailed explanations are listed in section D.9.2.

WRITE
SPACE
TOF
EOT
BUFCLR
PLOT
NOPLOT
NOWAIT
WAIT
STATUS
COUNTS

- Information Words

The two words listed below are used to pass information from the device routine to the user program.

SIZE Standard implementation - see section D.1.1. Contents updated by WRITE, SPACE, TOF, and EOT.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement or the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the MPG functions from which data will be included under each are:

BYTES WR:	WRITE, SPACE, (TOF & EOT in NOPLOT mode)
WRITE CMNDS:	WRITE
MISC CMNDS:	SPACE, TOF, EOT, BUFCLR
DEV ERRORS:	All printer errors
DATA ERRORS:	Errors detected by the VERIFY statement and PLOT commands to the LP11 or LS11.
INTERRUPTS:	Number of entries into the interrupt routine.
ADDITIONAL XFERS DURING INT:	The number of additional bytes transferred to the printer during one interrupt servicing. Varies according to printer buffer size and the data being printed.

- OPSW Special Operation Bit Support

Bit 7 of the OPSW (SPOPER) is interrogated only while performing WRITE commands. Normally, the Device Routine will automatically issue a carriage return and line feed (CR/LF) following each WRITE while in NOPLOT (printer) mode and a Line Terminate in PLOT mode when the data length is not a multiple of decimal 128. If the SPOPER bit is set, neither of these automatic features will be performed.

D.9.2 DESCRIPTION OF LP11/LS11/LV11 INSTRUCTIONS

The LP11/LS11/LV11 Device Routine supports execution of eleven MPG language statements. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2.

```
WRITE  (D256 FROM WRIO)
WRITE  v (FROM WRIO)
WRITE  v FROM v
```

This instruction transfers the number of bytes, specified by the first v, from memory, beginning at the memory location defined by the second v, to the printer. Unless inhibited by the SPOPER bit in the program's OPSW, a CR/LF will automatically be issued following the data when in printer mode or Line Terminate will be set when in PLOT mode with the data length not a multiple of decimal 128.

```
SPACE v
```

The number of lines specified by v will be advanced in either print or plot mode. In print mode, the two CR and LF bytes (015 and 012) will be sent to the printer for each line specified. In plot mode, two bytes of 0's are sent to the plotter followed by Line Terminate being set. This occurs for each plotter line to be advanced.

```
TOF
```

Performs a Top Of Form function in either print or plot mode. In print mode, the Form Feed byte (014) is sent to the printer. In plot mode, the Remote Form Feed bit is set in the C/S device register.

```
EOT
```

Performs an End Of Transmission function in either print or plot mode. In print mode, the End Of Transmission byte (004) is sent to the printer. In plot mode, the Remote End Of Transmission bit is set in the C/S device register.

BUFCLR

This instruction allows the user to clear the appropriate buffer in the LV11. The printer buffer will be cleared if in print mode while the plotter buffer will be cleared if in plot mode. This instruction does not use interrupts and has no effect if used on the LP11 or LS11.

PLOT

Specifies to the Device Routine that all following I/O functions are to be performed on the plotter section of the LV11. When in plot mode, the execution of all previously described I/O functions will be altered to accommodate the differences in the device's interface and operation. This statement results in an error if used with an LP11 or LS11 assigned to the program.

NOPLOT

Resets the PLOT mode and causes all subsequent I/O operations to be executed in print mode. This is the preset mode of operation when starting a prog-am.

NOWAIT

Standard implementation - see section D.1.1.

WAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

COUNTS

Standard implementation - see section D.1.1.

D.9.3 LP11/LS11/LV11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. Note that the contents of the device registers at the time of the error will be displayed for all errors except the "PLOT INV" error.

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

"PLOT" INV FOR LP11/LS11

When executing the program, a PLOT statement has been encountered. This command is valid only when an LV11 has been assigned as the device type. Increments the data error count.

RDY NOT SET

Before initiating an I/O transfer, the READY bit was not set in the printer's device register.

STATUS ERROR

This message indicates that the ERROR bit has been detected in the LP11/LS11/LV11 Control/Status device register.

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur when on the PDP-11/45 and with no other user programs executing is approximately 22 seconds.

D.9.4 LP11/LS11/LV11 SAMPLE PROGRAMS

The following are sample printer programs that perform the functions indicated in their descriptions:

1. Starting with spaces, print a 132 byte line of each character on a 96 character LP11 printer. After the last line, issue an End Of Transmission character.

```
*ENTER 3 AS ALPHAB
?ASGN DEV: LP11
?RDIO = 256 / 132
?WRIO = 256 / 132
?DEV REG = 177514 /
?INT VEC = 000200 /
?BUS REQ = 4   /

ENTER STMNT'S

>0010   LOAD TM00 WITH D96
>0020   FILL D132 AT RDIO WITH 401
>0030   FILL D132 AT WRIO WITH 20040
>0040   WRITE D132 FROM WRIO
>0050   ADD D132 AT RDIO TO WRIO
>0060   DECR TM00
>0070   IF TM00 > 0 GOTO 40
>0080   EOT
>0090   END
>DONE

*RUN 3
```

2. Utilize the plot and print modes of an LV11 by first plotting a 16 bit by 16 bit checkerboard pattern for one page. Then, advance to the next page, shift to printer mode and print a page of 132 byte lines consisting of the standard 64 character ASCII set with each line precessed one position. Follow this with an EOT character.

```
*ENTER 5 AS PRPLOT
?ASGN DEV: LV11
?RDIO = 256 / 0
?WRIO = 256 / 132

>0010   PLOT
>0020   LOAD TM00 WITH D25
>0030   FILL D128 AT WRIO WITH PAT6
>0040   LOAD TM01 WITH D16
>0050   WRITE D128
>0060   DECR TM01
>0070   IF TM01 > 0 GOTO 50
>0080   FILL D128 AT WRIO WITH PAT7
>0090   LOAD TM02 WITH D16
```

```
>0100   WRITE D128
>0110   DECR TM02
>0120   IF TM02 > 0 GOTO 100
>0130   DECR TM00
>0140   IF TM00 > 0 GOTO 30
>0150   TOF
>0160   NOPLOT
>0170   FILL D132 AT WRIO WITH ASCII
>0180   LOAD TM03 WITH D60
>0190   WRITE D132
>0200   ROTATE D132 AT WRIO
>0210   DECR TM03
>0220   IF TM03 > 0 GOTO 190
>0230   EOT
>0240   END
>DONE

*RUN 5
```

3. Test the line spacing capabilities of an LS11 by printing lines of E's with various spacing between them. Then check for correct Form Feed operation by issuing a TOF. Repeat the test three times on the assigned printer by using dummy unit numbers. Note that this test can be used on an LV11 in plot mode merely by inserting a PLOT statement at line number 0005. If this is done, E's will not be printed but instead, the plotter's binary bit representation of E's.

```
ENTER 8
?ASGN DEV: LS11,1,2,3
?RDIO = 256 / 0
?WRIO = 256 / 126

>0010   FILL D126 AT WRIO WITH 42505
>0020   WRITE D126
>0030   SPACE 1
>0040   WRITE D126
>0050   SPACE 2
>0060   WRITE D126
>0070   SPACE 3
>0080   WRITE D126
>0090   SPACE D10
>0100   WRITE D126
>0110   TOF
>0120   END
>DONE

*RUN 8
```

D.10 DH11 PROGRAMMABLE ASYNCHRONOUS 16-LINE MULTIPLEXER

PRESET ADDRESS - 160020
PRESET INTERRUPT VECTOR - 300
PRESET BUS REQUEST - 5,5 (READ/WRITE)

A. REGISTER NAMES RECOGNIZED

SCR
NRC
LPR
CAR
BYCR
BAR
BCR
SSR

B. MPG INSTRUCTIONS SUPPORTED

READ
WRITE
BREAK
CRESET
NOWAIT
WAIT
STATUS
COUNTS
FDUPLX
HDUPLX
EVEN
ODD
NOPAR
ONESTP
TWOSTP
BITS
ALARM
SETUP
RBAUD
TBAUD
BAUD
ECHO
NOECHO
PRESET

C. STATISTICAL INFORMATION

BYTES:	Read, Written
CMNDS:	Read, Write, Break, Misc.
INTERRUPTS:	Read, Write
ERRORS:	Non-existent Mem(NEM), Overrun(OVR), Framing(FRM), Parity(PAR), Silo Overflow(SOF), Data

D. OPSW SPECIAL OPERATION BIT SUPPORT

When Bit 7 (SPOPER) of the OPSW is set, the Maintenance Mode Bit (Bit 9 SCR) will be set prior to issuing each I/O data transfer command.

E. I/O TIMEOUT

If a terminating interrupt is not received 3 minutes after initiating an I/O operation, the program will be aborted and the user informed. This time is approximate, calibrated on a PDP-11/45 with no other user programs running.

F. INSTRUCTION DESCRIPTIONS

```
READ  (D256 INTO RDIO)       () = default values
READ  v (INTO RDIO)
READ  v INTO v
```

This instruction will cause a data transfer to occur between the unit previously assigned to this Device Routine and the memory area specified by the second variable. The first variable contains the byte count for the transfer.

```
READ  v INTO v FROM u
```

Same as above read except u specifies the units or line numbers. Whatever unit numbers were assigned have no effect. This allows simultaneous data transfers on up to 16 lines. A typical form of u is 4-7&11 or 4 THRU 7 AND 11.

```
WRITE  (D256 FROM WRIO)
WRITE  v (FROM WRIO)
WRITE  v FROM v
WRITE  v FROM v TO u
```

The description for the READ instruction applies to the WRITE except for the direction of data flow.

```
BREAK  v
BREAK  v ON u
```

This instruction causes a break (spacing condition) on the assigned unit or on those specified by u as in the READ instruction. The duration at the break is determined by v which specifies the number of character times to hold the spacing condition.

```
CRESET
```

This instruction sets the Master Clear bit in the System Control Register, then clears all bits in the SCR.

NOWAIT

Standard implementation - see section D.1.1.

WAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

COUNTS

Standard implementation - see section D.1.1.

ALARM v

This instruction loads the silo alarm level in the Silo Status Register. The only acceptable values of v are 0,1,2,4,10 or D8, 20 or D16 and 40 or D32. All other values will be intercepted when the program is run. After an error message is printed, the program will take the normal error action as determined by the OPSW.

SETUP u

All of the following instructions listed, set or reset bits in the Line Parameter Register. The SETUP instruction selects which line parameter registers will be updated by these instructions. The unit u can be any line or combination of lines between 0 and 15, such as 3 THRU 7 AND 15.

RBAUD v

Sets the Receiver speed as indicated in the table.

TBAUD v

Sets the Transmitter speed as indicated in the table.

BAUD v

Sets both Receiver and Transmitter speed as indicated in the table.

Speed Table for Receiver and Transmitter

v(Octal)	v(Decimal)	Baud Rate
0	D0	0
1	D1	50
2	D2	75
3	D3	110
4	D4	134.5
5	D5	150
6	D6	200
7	D7	300
10	D8	600
11	D9	1200
12	D10	1800
13	D11	2400
14	D12	4800
15	D13	9600
16	D14	EXT INP A
17	D15	EXT INP B

EVEN

Sets Even Parity.

ODD

Sets Odd Parity.

NOPAR

Disables Parity.

ONESTP

Sets number of stop bits = 1

TWOSTP

Sets number of stop bits = 2

BITS v

Sets bits per character according to v. Valid entries are 5,6,7 and 10 or D8

ECHO

Enables Auto-Echo

NOECHO

Disables Auto-Echo

PRESET

Sets 8 Bits per Character, disables Parity, sets 2 Stop Bits, 110 Baud, and Full Duplex with No Echo.

D.10.1 DH11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2.

The unique error messages which may occur in the use of this device routine are as follcws:

DH11 TIMEOUT ON I/O

An I/O operation was begun, but the terminating interrupt was not received. The time allowed for this interrupt to occur is approximately three minutes when the program is run on the PDP-11/45 and no other programs are running.

DH11 ERROR: ddd,ddd,etc.

The codes ddd represent mnemonics which are printed to indicate which error bits were set. The message may contain from one to five codes. The following mnemonics may be printed in this message:

PAR = Received Data Parity Error
FRM = Framing Error
OVR = Data Overrun Error
NEM = Non-existent Memory Error
SOF = Silo Overflow Error (storage interrupt)

BAUD SELECTION CODE > 15

The user program entered a value with the RBAUD or TBAUD statement to select a baud rate. This value must be from 0 to 15. A value greater than 15 was entered.

OF CHAR BITS NOT 5,6,7 OR 8

The user program entered a value with the BITS statement to select the length of a character. This value must be 5, 6, 7, or 8. A value not in that range was entered.

SILO ALARM LEVEL OTHER THAN 0,1,2,4,8,16 OR 32

The user program entered a value with the ALARM statement to set the silo alarm level. This value must be an integral power of 2 between 0 and 32. A value not in that range was entered.

D.10.2 DH11 SAMPLE PROGRAMS

1. Write same data sequentially to units 3,4.5 and 11.

```
*ENTER 1
?ASGN DEV: DH11,3,4,5,11

>0010  FILL D64 AT WRIO WITH ASCII
>0020  CRESET
>0030  SETUP 3-5&11
>0040  PRESET
>0050  WRITE D64
>0060  WRITE 2 FROM CRLF
>0070  END
>DONE

*RUN 1
```

2. Write same data simultaneously to units 3,4.5 and 11.

```
*ENTER 1
?ASGN DEV: DH11

>0010  FILL D64 AT WRIO WITH ASCII
>0020  CRESET
>0030  SETUP 3-5&11
>0040  PRESET
```

```
>0050  WRITE D64 FROM WRIO TO 3-5&11
>0060  WRITE 2 FROM CRLF TO 3-5&11
>0070  END
>DONE

*RUN 1
```

3. Write different data sequentially to units 5,9 and 15 at 300 Baud.

```
*ENTER 1
?ASGN DEV: DH11

>0010  CRESET
>0020  SETUP 5 AND 9 AND 15
>0030  PRESET
>0040  BAUD 7
>0050  WRITE D16 FROM BF00 TO 5
>0060  WRITE D16 FROM BF01 TO 9
>0070  WRITE D16 FROM BF02 TO 15
>0080  END
>DONE

*RUN 1
```

4. Write different data simultaneously to units 5,9 and 15. Setup for 7 bits per character, even parity, 2 stop bits at 110 baud to line 5, 150 baud to line 9 and 300 baud to line 15.

```
*ENTER 1
?ASGN DEV: DH11

>0010  CRESET
>0020  SETUP  5&9&15
>0030  BITS 7
>0040  EVEN
>0050  TWOSTP
>0060  SETUP 5
>0070  BAUD  3
>0080  SETUP 9
>0090  BAUD 5
>0100  SETUP 15
>0110  BAUD 7
>0120  NOWAIT
>0130  WRITE D16 FROM BF00 TO 5
>0140  WRITE D16 FROM BF01 TO 9
>0150  WRITE D16 FROM BF02 TO 15
>0160  WAIT
>0170  END
>DONE
```

```
*FILL BF00 WITH *TEST MSG NBR 1
*FILL BF01 WITH *TEST MSG NBR 2
*FILL BF02 WITH *TEST MSG NBR 3
*RUN 1
```

5. Read 1 character from unit 3 and write it to units 3,7,8 and 9. Use the operation switch to loop on this program. Note- The PRESET instruction sets the baud rate at 110. Add the necessary BAUD instruction to match the available terminals.

```
*ENTER 1
?ASGN DEV: DH11

>0010  CRESET
>0020  SETUP 3 AND 7 THRU 9
>0030  PRESET
>0040  READ 1 INTO TM00 FROM 3
>0050  WRITE 1 FROM TM00 TO 3&7-9
>0060  END
>DONE

*OPSW1 140020
*RUN 1
```

6. Test the DH11 using the maintenance bit to cause data wraparound. Write the same data simultaneously to lines 0,1,2 and 3 and read back the data into separate memory areas. Print the data received.

```
*ENTER 1
?ASGN DEV: DH11

>0010  FILL D64 AT BF00 WITH 5015
>0020  CRESET
>0030  SETUP 0 THRU 3
>0040  PRESET
>0050  NOWAIT
>0060  READ D16 INTO BF00 FROM 0
>0070  READ D16 INTO BF01 FROM 1
>0080  READ D16 INTO BF02 FROM 2
>0090  READ D16 INTO BF03 FROM 3
>0100  WRITE D16 FROM COM0 TO 0-3
>0110  WAIT
>0120  PRINT D16 AT BF00 IN ASCII
>0130  PRINT D16 AT BF01 IN ASCII
>0140  PRINT D16 AT BF02 IN ASCII
>0150  PRINT D16 AT BF03 IN ASCII
>0160  END
>DONE

*FILL COM WITH *DH11 TST MSG 1
```

```
*OPSW 200
*RUN 1
```

7. Modify Program 6 to run at 9600 Baud.

```
*MODIFY 1

ENTER STMNT'S

>45     BAUD 15
>DONE

*RUN 1
```

8. Modify Programs 6 or 7 to receive and transmit at different baud rates, causing a framing error.

```
*MODIFY 1

ENTER STMNT'S

>45     RBAUD 5
>47     TBAUD 4
>DONE

*RUN 1
```

9. Using the Special Operation bit in the OPSW to set the maintenance bit, loop back all !ines and run at 9600 Baud to test silo overflow error detection.

```
*ENTER 1
?ASGN DEV: DH11

>0010  FILL D64 AT WRIO WITH ASCII
>0020  CRESET
>0030  SETUP 0-15
>0040  PRESET
>0050  BAUD 15
>0060  NOWAIT
>0070  READ D64 INTO RDIO FROM 0-15
>0080  WRITE D64 FROM WRIO TO 0-15
>0090  WAIT
>0100  PRINT D64 AT  RDIO IN ASCII
>0110  END
>DONE

*OPSW 200
*RUN 1
```

D.10 - DH11 Device Routine (Cont'd)

10. Modify Program 9 to run at 4800 Baud and therefore run without error.

```
*MODIFY 1

ENTER STMNT'S

>50    BAUD 14
>DONE

*RUN 1
```

11. Test the silo alarm for proper operation. Use the Special Operation bit in the OPSW for data loopback.

```
*ENTER 1
?ASGN DEV: DH11,0

>0010  FILL D16 AT WRIO WITH ASCII
>0020  FILL D16 AT RDIO WITH 20040
>0030  CRESET
>0040  PRESET
>0050  BAUD 15
>0060  ALARM D16
>0070  NOWAIT
>0080  READ D16
>0090  WRITE D15
>0100  DELAY D1000
>0110  PRINT D16 AT RDIO IN ASCII
>0120  PRINT 2 AT CRLF IN ASCII
>0130  WRITE 3
>0140  WAIT
>0150  PRINT D16 AT RDIO IN ASCII
>0160  PRINT 2 AT CRLF IN ASCII
>0170  END
>DONE

*OPSW 200
*RUN 1
```

Comments: The first printed line should be blank because not enough characters were received to cause an interrupt by exceeding the silo alarm level. The second line should contain 16 ASCII characters. This test can be repeated for all valid alarm levels.

D.11 PC11/PR11 HIGH SPEED PAPER TAPE READER PUNCH

PRESET ADDRESS - 177550
PRESET INTERRUPT VECTOR - 70
PRESET BUS REQUEST - 4,4 (READ/WRITE)

A. REGISTER NAMES RECOGNIZED

PRS
PRB
PPS
PPB

B. MPG INSTRUCTIONS SUPPORTED

READ
WRITE
LEADER
NOWAIT
WAIT
STATUS
COUNTS

C. STATISTICAL INFORMATION

BYTES:	Read, Write
CMNDS:	Read, Write, Misc.
INTERRUPTS:	Read, Write
ERRORS:	Reader(RDR), Punch(PUN), Data

D. OPSW SPECIAL OPERATION BIT SUPPORT

The setting of Bit 7 (SPOPER) of the OPSW, has no effect on this Device Routine.

E. I/O TIMEOUT

If a terminating interrupt is not received 3 minutes after initiating an I/O operation, the program will be aborted and the user informed. This time is approximate, calibrated on a PDP-11/45 with no other user programs running.

F. INSTRUCTION DESCRIPTIONS

```
READ  (D256 INTO RDIO)
READ  v (INTO RDIO)
READ  v INTO v
```

This instruction will cause a data transfer to occur between the Paper Tape Reader and the memory area specified by the second variable. The first variable contains the byte count for the transfer.

```
WRITE  (D256 FROM WRIO)
WRITE  v (FROM WRIO)
WRITE  v FROM v
```

This instruction will cause a data transfer to occur between the memory area specified by the second variable and the Paper Tape Punch. The first variable contains the byte count for the transfer.

```
LEADER  v
```

This instruction causes a blank strip of tape to be punched, whose length in inches is specified by the variable.

```
NOWAIT
```

Standard implementation - see section D.1.1.

```
WAIT
```

Standard implementation - see section D.1.1.

```
STATUS
```

Standard implementation - see section D.1.1.

```
COUNTS
```

Standard implementation - see section D.1.1.

D.11.1 PC11/PR11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2.

The unique error messages which may occur in the use of this device routine are as follows:

PC11/PR11 TIMEOUT ON I/O

An I/O operation was begun, but the terminating interrupt was not received. The time allowed for this interrupt to occur is approximately three minutes when the program is run on the PDP-11/45 and no other programs are running.

PC11/PR11 ERROR: ddd,etc

The codes ddd represent mnemonics which are printed to indicate which error bits were set. The message may contain one or two codes. The following mnemonics may be printed in this message:

PUN = Punch Error (PPS)
RDR = Reader Error (PRS)

D.11.2 PC11/PR11 SAMPLE PROGRAMS

1. Punch a 64 byte ASCII character string, preceeded by a 10 inch leader and STX code.

```
*ENTER 1
?ASGN DEV:  PC11

>0010  LOAD TMOO WITH 2
>0020  FILL D64 AT WRIO WITH ASCII
>0030  LEADER D10
>0040  WRITE 1 FROM TMOO
>0050  WRITE D64
>0060  LEADER D10
>0070  END
DONE

*RUN 1
```

2. Read and print the tape produced in (1), ignoring the leader and STX character.

```
*ENTER 1
?ASGN DEV:  PC11

>0010  READ 1
>0020  IF 1 AT RDIO <> 2 GOTO 10
>0030  READ D64
>0040  PRINT D64 AT RDIO IN ASCII
>0050  END
DONE

*RUN 1
```

3. Read and punch simultaneously.

```
*ENTER 1
?ASGN DEV:  PC11

>0010  FILL D256 AT WRIO WITH ASCII
>0020  NOWAIT
>0030  READ
>0040  WRITE
>0050  WAIT
>0060  END
DONE

*RUN 1
```

D.12 RP04/RP05/RP06 DISKS' DEVICE ROUTINE

The RP04/RP05/RP06 Device Routine, whose filename is TRPAnm.MPG, supports the operation of RP04, RP05, and RP06 disk drives on the standard RH11 MASSBUS controller and on the 11/70 version (RH70) of the RH11.

This Device Routine will automatically adjust its operation based upon the device assigned to it and the CPU that it is operating on. For the RP06 it will adjust to a maximum of 815 cylinders. Error bits displayed in the ERROR BITS message will be applicable to the device and the controller. Error bits will be included as necessary for common errors, RP04 errors, RP05/RP06 errors, and RH70 errors. Also, two additional device registers are supported for the RH70.

D.12.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

  In reply to the "ASGN DEV:" message, three model names (RP04, RP05 and RP06) are acceptable and may be accompanied by a maximum of 16 unit numbers. For this device the unit numbers must be in the range of 0 thru 7.

- Interface Addresses

  The following are the values preset for the RP04/RP05/RP06 and may be altered from the console terminal following the "ASGN DEV:" message:

  Device Register Base Address = 176700
  Interrupt Vector Address = 254
  Bus Request Priority = 5

- Symbolic Register Names

  The symbolic names listed below may be used to reference the RH11/RH70 - RP04/RP05/RP06 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register. Note that the last two (RPAE and RPC3) will be supported only when running on an 11/70.

Name	Displ	Register
RPC1	+0	Control and Status 1
RPWC	+2	Word Count

RPBA	+4	Unibus Address
RPDA	+6	Desired Sector/Track Address
RPC2	+10	Control and Status 2
RPDS	+12	Drive Status
RPE1	+14	Error Register 1
RPAS	+16	Attention Summary
RPLA	+20	Look Ahead
RPDB	+22	Data Buffer
RPMR	+24	Maintenance
RPDT	+26	Drive Type
RPSN	+30	Serial Number
RPOF	+32	Offset
RPDC	+34	Desired Cylinder
RPCC	+36	Current Cylinder
RPE2	+40	Error Register 2
RPE3	+42	Error Register 3
RPPO	+44	ECC Position
RPPA	+46	ECC Pattern
RPAE	+50	Bus Address Extension
RPC3	+52	Control and Status 3

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the RP04/RP05/RP06. Detailed explanations are listed in section D.12.2.

READ
WRITE
RDHD
WRHD
WRCK
WRCKHD
SEEK
SEARCH
OFFSET
RETCTR
UNLOAD
RECAL
CRESET
DRESET
PAKACK
RDPSET
REL
STEPUP
STEPDN
WAIT
NOWAIT
STATUS
COUNTS
APORT
BPORT
FMT22
FMT20
ODD
EVEN
ECION
ECIOFF
HCION
HCIOFF
BAION
BAIOFF
CORON
COROFF
VVON
VVOFF

Certain of the above instructions control the setting of internal flag bits. The preset states of these bits are the same as though the following instructions were issued:

WAIT
APORT
FMT22
ODD
ECIOFF
HCIOFF
BAIOFF
CORON
VVON

- Information Words

The six words listed below are used to pass information between the user program and the device routine. All words are preset to 0's except RTRY which is preset to 3.

CYL A one word location that contains the cylinder number in bits 0-9 that will be used in subsequent I/O operations. The contents of CYL will be loaded into RPDC for all I/O operations that require a cylinder address.

HEAD The word location that follows CYL and contains the value of the head (track) number to be used in bits 0-4. The value in this word and in SECT will be merged and loaded into RPDA for all I/O operations that require a head and sector address.

SECT This word contains the sector number in bits 0-4. Located immediately following HEAD in memory.

RTRY This word specifies the number of additional attempts that the Device Routine will try on an I/O operation before deciding that it is unrecoverable. Applicable only to those types of errors that normally have a possibility of recovery.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, RDHD, WRHD, WRCK and WRCKHD.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement and the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ, RDHD
BYTES WR:	WRITE, WRHD
BYTES CK:	WRCK, WRCKHD
READ CMNDS:	READ, RDHD
WRITE CMNDS:	WRITE, WRHD
CHECK CMNDS:	WRCK, WRCKHD
SEEK CMNDS:	SEEK, SEARCH, OFFSET, RETCTR, RECAL
MISC CMNDS:	DRESET, CRESET, PAKACK, RDPSET, REL, UNLOAD
DEV ERRORS:	All hardware errors that resulted in an error report. Correctable ECC errors and errors that were recoverable when retried will not increment this count.
CORR ECC ERRORS:	Number of ECC errors corrected by software.
DATA ERRORS:	Invalid unit number errors and errors detected by the VERIFY statement.

RETRYS: The number of occurrences for each of the seven retryable type of errors. Counts are maintained for DLT, DTE, HCRC, FER, HCE, DCK, and WCE.

TOTAL RETRYS: Total number of retry attempts on all failing I/O operations.

INTERRUPTS: Number of entries into the interrupt routine.

- Correctable ECC Errors

When encountering a correctable ECC error (DCK), this device routine will automatically correct the data. The contents of the ECC Position register (RPEC1) and the ECC Pattern register (RPEC2) will be used to determine the bits to be modified and their placement in the data field. After the correction has been applied, the CORR ECC count will be incremented and the data transfer will be resumed at its point of interruption.

This automatic data correction will not be performed if the ECI bit is set, if the word length is 18 bits (FMT22 reset), or if the user has entered the COROFF statement. In these cases, the DCK will be treated the same as the other retryable errors.

- Retryable Errors

Seven errors are classified as retryable by this device routine. When one occurs (DLT, DTE, HCRC, FER, HCE, DCK, or WCE), the retry count contained in the interface word RTRY is checked for a zero value. If zero, an I/O Termination error is reported. If non-zero, the retry count for the error and the total retry error count are incremented by one and the I/O operation is re-issued. If the retry is successful, the program will proceed to the next statement in the user's program. If it was not successful, the retry count from RTRY is decremented, the total retry count is incremented, and the command is re-issued. This continues until either the retry is successful or the RTRY count goes to zero. When the latter occurs, the "EXHAUSTED RETRIES" error message will be issued and the DEV error count will be incremented.

When processing retries on failing I/O operations, the original function will just be re-issued. The use of Offsets and Recalibrate commands are not employed during error recovery in this device routine.

- Disk Types

This device routine does not interrogate the Drive Type register to determine which type of disk it is operating on. It relies solely on the model number assigned to the program. If an RP04 has been assigned but it is actually an RP06, the device routine will process it as an RP04.

- Dual Controller Operation

This device routine supports operation of disks connected to two systems through the dual controller option. If the disk is busy on the other controller when the device routine is ready to issue an I/O operation, it will wait up to 22 seconds for the disk to become available. If it does not acquire the disk within that time, a timeout error will be reported. In order to make efficient use of the disk on a dual controller system, both programs operating the disk should issue Release commands through use of the REL statement provided by this device routine.

- Instructions And Interrupts

Certain of the I/O functions for the RP04/RP05/RP06 disks do not normally terminate with interrupts and will not have Interrupt Enable set when they are issued. These functions are termed Non-Interrupt functions and are:

DRESET
PAKACK
RDPSET
REL
UNLOAD

After issuing the above functions, the device routine will delay a few microseconds and then check the status of the error bits. Any found to be set will be reported as a Non-Int Termination error.

The remaining I/O functions, which are listed below, utilize interrupts in their operations:

READ
WRITE
RDHD
WRHD
WRCK
WRCKHD
SEEK
SEARCH
OFFSET
RETCTR
RECAL

For the above instructions, tests for error conditions will be made when the terminating interrupt is received.

When operating with a dual controller disk and the disk is busy on the other controller, interrupts will be enabled to allow ATA to signal when the disk has been acquired. This feature is utilized for both groups of functions listed previously. Therefore, the interrupt count may be incremented on a PAKACK instruction, for example, even though it is not a result of the command.

- Accessing The Data Buffer Register

  The contents of the Data Buffer (RPDB) register will be stored only when processing an interrupt with the Output Ready (OR) bit set in the RPCS2 register. At all other times, this register will not be read and its contents will be displayed as 0's.

- OPSW Special Operation Bit Support

  Bit 7 of the OPSW (SPOPER) is not supported by this device routine and its setting has no effect on disk operation.

D.12.2 DESCRIPTION OF RP04/RP05/RP06 INSTRUCTIONS

The RP04/RP05/RP06 Device Routine supports execution of thirty nine MPG language statements. For certain functions (READ, WRITE, RDHD, WRHD, WRCK, WRCKHD, SEEK and SEARCH), the desired disk address must be loaded into CYL, HEAD, and SECT before performing those functions. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2. Note that if an odd byte count is supplied in any of the following instructions, it will effectively be decremented by 1 before being used.

READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v

This instruction transfers the number of bytes, indicated by the first v, from the disk to memory beginning at the memory location specified by the second v.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Transfer the number of bytes from memory to disk beginning at the memory location indicated by the second v.

The following statements do not have default values for their operands.

RDHD v INTO v

Reads the four words of sector header data and the data field into memory using the Read Header and Data command. In order to read a complete sector, the byte count must reflect the 4 header words.

WRHD v FROM v

Performs a format type of operation by using the Write Header and Data command to write the header words and data field on disk.

WRCK v AT v

Compares the number of data field bytes in memory to those at the specified disk location. Uses the Write Check Data command.

WRCKHD v AT v

Similar to the WRCK command but also compares the header data through use of the Write Check Header and Data command.

SEEK

Performs a seek function to the disk location specified by CYL, HEAD, and SECT. This instruction does not terminate until the seek is finished.

SEARCH

Similar to SEEK but uses the Search command.

The following I/O commands do not utilize the contents of CYL, HEAD, and SECT in their operations.

OFFSET v

The least significant eight bits of the quantity specified by v will be loaded into the Offset register followed by the Offset command being issued. Examples of the offset obtained with different octal values of v are:

v	Microinches
010	+200
210	-200
020	+400
220	-400
030	+600
230	-600
040	+800
240	-800
060	+1200
260	-1200

RETCTR

This statement is normally issued after OFFSET statements and results in the Return to Center Line command being issued.

UNLOAD

Places the drive in the Standby state by issuing the Unload command. Note that this command is not considered to be terminated until the drive is brought back on-line. While the drive is off-line, the device routine will check on its status approximately once every second. Timeout is not enabled for this command and the device routine will wait an indefinite amount of time for the drive to become operable. Upon determining that the drive is operable again, the device routine will proceed to the next statement in the user's program.

RECAL

Results in the Recalibrate command being issued.

CRESET

This statement performs a controller clear function by setting the CLR bit in the RPCS2 register.

DRESET

Housekeeping of the drive is accomplished by this statement's issuing of the Drive Clear command.

PAKACK

Causes the Pack Acknowledge command to be issued.

RDPSET

Causes the Read-in Preset command to be issued.

REL

Performs a drive clear and releases the drive for another port by issuing the Release command.

The following instructions do not perform I/O operations:

STEPUP v

This command does not perform any I/O functions. It provides easy incrementing of the contents of CYL, HEAD, and SECT. The operand v is the number of sectors that these values are to be incremented. When the SECT word exceeds its maximum valid value, it will be set to 0 and the contents of HEAD will be incremented. When HEAD overflows, it will be set to 0 and CYL will be incremented. When incrementing past the last sector on disk, all three values will be set to 0's. Automatically adjusts to the number of sectors on a track (20 or 22) based upon the current setting defined by the FMT20/FMT22 statements and to the number of cylinders (411 for the RP04/RP05 vs. 815 for the RP06).

Note that this instruction and STEPDN will operate with invalid values in CYL, HEAD, and SECT upon execution. Regardless of their initial contents, the contents of all three words will be valid when this instruction completes. This allows the use of the "FILL 6 AT CYL WITH RANDOM" statement followed by "STEPUP 0" to generate random disk addresses.

STEPDN v

Similar to STEPUP but provides a decrementing capability. When all three values are 0, the next decrement will result in the decimal values for the three words of 410/814, 18, 21/19.

WAIT

Standard implementation - see section D.1.1.

NOWAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

Included with the standard display is the display of the current contents of the CYL, HEAD and SECT words. These values do not reflect the current position of the disk, but merely the contents of the three words.

COUNTS

Standard implementation - see section D.1.1.

The following statements define to the Device Routine the desired settings of certain bits in the device registers. These statements do not immediately alter the register bits but all subsequent I/O functions and other applicable statements will be performed with the bits set as defined.

APORT

Indicates that the value of the PSEL bit in RPCS1 is to be set to 0 (UNIBUS Port A). (Preset mode)

BPORT

PSEL will be set to a 1 (UNIBUS Port B) on subsequent functions.

FMT22

Sets the number of sectors on a track to 22 and the number of bits in the data field words to 16. FMT22 is the preset mode and results in the FMT22 bit in the RPOF register being set to a 1.

FMT20

Sets the number of sectors on a track to 20 and the number of bits for the data field words to 18. Resets the FMT22 bit in RPOF to a 0.

ODD

Causes the Parity Select (PAT) bit in RPCS2 to be in the 0 state on subsequent functions. (Preset mode)

EVEN

Causes PAT to be set to a 1.

ECION

Inhibits the Error Correction Code logic by setting the ECI bit in the RPOF register to a 1.

ECIOFF

Allows ECC operation by resetting ECI to 0. (Preset mode)

HCION

Inhibits the Header Compare function by setting the HCI bit in the RPOF register to a 1.

HCIOFF

Indicates that the Header Compare function is to be performed by resetting HCI to a 0. (Preset mode)

D.12 - RP04/RP05/RP06 Device Routine (Cont'd)

BAION

Specifies that bus address incrementing is not to be performed by setting the BAI bit in the RPCS2 register to a 1.

BAIOFF

Results in BAI being reset to a 0 and bus address incrementing being performed. (Preset mode)

CORON

This statement activates the device routine's programmed function of correcting data on which a correctable ECC error was detected. When the disk detects an error and determines that it can be corrected, the device routine will perform the correction procedures, increment the CORR ECC error count, and continue running. (Preset mode)

COROFF

Disables the programmed recovery of correctable ECC errors. When this type of error is detected, an error display will result unless it is recoverable on retries.

VVON

This statement instructs the device routine to issue the Pack Acknowledge command prior to any subsequent I/O functions, other than PAKACK, RDPSET, or DRESET, if the Volume Valid (VV) bit is not already set. This is the preset mode and allows the program to run without modification on a pack which has just been mounted.

VVOFF

Inhibits the automatic issuing of the Pack Acknowledge command.

D.12.3 RP04/RP05/RP06 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. However, due to the nature of this device, additional information is provided to the user.

The I/O function type of instructions are considered to execute in two stages. The first is the time it takes to acquire the disk and housekeep it, if needed. The second is after the point the specified function is issued and until after the function terminates. Since error conditions may occur in either stage, this device routine indicates which stage it was in when the error was detected. This is accomplished by including the appropriate message of the following two messages in the error display:

```
BEFORE ISSUING I/O CMND

AFTER ISSUING I/O CMND
```

If the error is detected before issuing the specified I/O command and MPG's Continue (CONT) command is issued, the program will be resumed at the statement on which the error was detected. If an AFTER error, the program will resume at the next statement in the user's program.

Also included with every error display, other than the invalid unit number error, is the following message:

```
ERROR BITS:
ddd,ddd,ddd,ddd,ddd,ddd,etc.
```

This message indicates that one or more error status bits have been detected in the device registers. The codes ddd identify which error bits were found. A sixty four byte field will be used to list all or as many of the error bits as possible. In the following list some codes are distinguished by an asterisk (*). These codes will be included in the message if their bit values are 0. All other codes are included if they have a value of 1. The following are the mnemonic codes supported for the ddd fields:

Error bits common to all devices:

SC	= Special Condition
TRE	= Transfer Error
MCPE	= Massbus Control Bus Parity Error
*DVA	= Drive Available
DLT	= Data Late
WCE	= Write Check Error
UPE	= Unibus Parity Error
NED	= Nonexistent Drive
NEM	= Nonexistent Memory

D.12 - RP04/RP05/RP06 Device Routine (Cont'd)

PGE	= Program Error
MXF	= Missed Transfer
MDPE	= Massbus Data Bus Parity Error
ATA	= Attention Active
ERR	= Error
PIP	= Positioning In Progress
*MOL	= Medium On-Line
*DPR	= Drive Present
*DRY	= Drive Ready
*VV	= Volume Valid
DCK	= Data Check
UNS	= Unsafe
OPI	= Operation Incomplete
DTE	= Drive Timing Error
WLE	= Write Lock Error
IAE	= Invalid Address Error
AOE	= Address Overflow Error
HCRC	= Header CRC Error
HCE	= Header Compare Error
ECH	= ECC Hard Error
WCF	= Write Clock Fail
FER	= Format Error
PAR	= Parity Error
RMR	= Register Modification Refused
ILR	= Illegal Register
ILF	= Illegal Function
ATA7	= Attention Active Drive 7
ATA6	= Attention Active Drive 6
ATA5	= Attention Active Drive 5
ATA4	= Attention Active Drive 4
ATA3	= Attention Active Drive 3
ATA2	= Attention Active Drive 2
ATA1	= Attention Active Drive 1
ATA0	= Attention Active Drive 0
PLU	= PLO Unsafe
IXE	= Index Error
NHS	= No Head Selection
MHS	= Multiple Head Selection
WRU	= Write Ready Unsafe
TUF	= Transitions Unsafe
TDF	= Transitions Detector Failure
CSU	= Current Switch Unsafe
WSU	= Write Select Unsafe
CSF	= Current Sink Failure
WCU	= Write Current Unsafe
OCYL	= Off Cylinder
SKI	= Seek Incomplete
DCL	= DC Low
ACL	= AC Low

Error bits unique to the RP04:

ACU	=	AC Unsafe
30VU	=	30 Volts Unsafe
FEN	=	Failsafe Enabled
MSE	=	Motor Sequence Error
UWR	=	Any Unsafe Except Read/Write
VUF	=	Velocity Unsafe
PSU	=	Pack Speed Unsafe

Error bits unique to the RP05/RP06:

ABS	=	Abnormal Stop
RAW	=	Read And Write
OPE	=	Operator Plug Error
WAO	=	Write And Offset
DCU	=	DC Unsafe

Error bits unique to the RH70:

APE	=	Address Parity Error
DPEOW	=	Data Parity Error, Odd Word
DPEEW	=	Data Parity Error, Even Word
WCEOW	=	Write Check Error, Odd Word
WCEEW	=	Write Check Error, Even Word

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

DISK IS OFF-LINE

Immediately prior to initiating an I/O operation, the Medium On Line bit (MOL) in the RPDS register was found to be reset.

DPR NOT SET

While waiting to acquire the disk from the other controller, an interrupt with the drive's ATA bit was received but the Drive Present (DPR) bit in the RPDS register was not set.

ERROR ON INITIATION

Prior to initiating an I/O operation, a condition that sets the SC bit in RPCS1, other than an Unsafe, did not clear after five attempts to reset it. The error bits that caused SC to be set will be listed.

EXHAUSTED RETRIES

Upon detecting one of the seven retryable errors, the device routine has re-issued the failing I/O operation the number of times specified in RTRY without successful completeion of the operation. If RTRY is initially 0, this message will not occur.

INT WITHOUT ATA

This message indicates that an interrupt was received without the drive's ATA bit being set and it was expected to be set. This error may occur before or after issuing the I/O function. When operating a dual controller disk and it is busy on the other controller, the device routine waits for an interrupt with ATA set to indicate that it has acquired the disk. If an interrupt is received and ATA is not set, this error is reported. The other case is where an I/O function (SEEK, SEARCH, etc), that normally terminates with ATA, is issued and an interrupt was received without ATA being set.

INV ECC BIT PATTERN

When ready to apply the correction to the data on a correctable ECC error, the RPEC2 Bit Pattern register was found to contain all 0's.

INV ECC BIT POSITION

When ready to apply the correction to the data on a correctable ECC error, the RPEC1 Bit Position register was found to contain a value greater than octal 10041.

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erroneous unit number. Increments the data error count.

I/O TERMINATION ERROR

Error bits have been detected in the device registers when processing a termination interrupt. This message occurs for all non-retryable errors and retryable errors when RTRY is set to 0.

NON-EXISTENT DRIVE

Immediately after loading a valid unit number (0-7) into RPCS2, the Non-Existent Drive bit (NED) in RPCS2 was found to be set.

NON-INT I/O TERMINATION ERROR

Error bits were found set in the device registers a few microseconds after issuing one of the Non-Interrupt type of I/O functions (PAKACK, REL, etc).

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately 22 seconds when on the PDP-11/45 and with no other user programs executing.

T/O ON CRESET

This error, which indicates that Ready (RDY) did not set within a few milliseconds after setting the CLR bit in RPCS2, may occur when the CRESET statement is issued.

T/O ON DISK ACQUIRE

When ready to issue an I/O operation to a disk that is busy on the other controller in a dual controller configuration, the disk did not become available within approximately 22 seconds. Either the disk is not in the programmable mode or the program operating the disk on the other controller should be modified to include Release commands.

UNEXP ATA COND

When processing an interrupt after issuing an I/O function that should not cause an ATA condition, the disk's ATA bit was found set.

UNSAFE ERROR ON INITIATION

Prior to initiating an I/O operation, the Unsafe bit (UNS) in RPER1 was found to be set. If the previous I/O operation did not result in an error, this error is reported immediately. If there was an error on the previous operation, a drive clear will be issued to reset the condition. If UNS is still set after the drive clear, this error is reported.

D.12.4 RP04/RP05/RP06 SAMPLE PROGRAMS

The following are sample RP04/RP05/RP06 programs that perform the functions indicated in their descriptions:

1. The following program will verify all tracks on the entire disk by writing 1 bits to the data fields. Retries will be inhibited so as to isolate marginal tracks:

```
*ENTER 1 AS TRKCK
?ASGN DEV: RP04,0
?RDIO = 256 / 0
?WRIO = 256 / 11264
?DEV REG = 176700 /
?INT VEC = 000254 /
?BUS REQ = 5   /

ENTER STMNT'S

>0010   LOAD RTRY WITH 0
>0020   FILL D11264 AT WRIO WITH 177777
>0030   WRITE D11264 FROM WRIO
>0040   WRCK D11264 AT WRIO
>0050   STEPUP D22
>0060   IF 6 AT CYL <> 0 GOTO 30
>0070   END
>DONE

*RUN 1
```

2. Do 200 pairs of random seeks to an RP06 disk by using the STEPUP instruction to generate valid disk addresses from random data. Intersperse other instructions to perform data transfers of random data and do data validation at these disk addresses:

```
*ENTER 3 AS RPRAND
?ASGN DEV: RP06.5
?RDIO = 256 / 512
?WRIO = 256 / 512

ENTER STMNT'S

>0010    LOAD TM00 WITH D200
>0020    FILL 6 AT CYL WITH RANDOM
>0030    STEPUP 0
>0040    MOVE 6 AT CYL TO TM03
>0050    FILL D512 AT WRIO WITH RANDOM
>0060    SEEK
>0070    WRITE D512 FROM WRIO
>0080    WRCK D512 AT WRIO
>0090    FILL 6 AT CYL WITH RANDOM
>0100    STEPUP 0
>0110    IF 6 AT CYL = TM03 GOTO 90
>0120    MOVE 6 AT CYL TO TM06
>0130    SEARCH
>0140    WRITE D512 FROM WRIO
>0150    WRCK D512 AT WRIO
>0160    MOVE 6 AT TM03 TO CYL
>0170    SEARCH
>0180    READ D512 INTO RDIO
>0190    VERIFY D512 AT RDIO WITH WRIO
>0200    MOVE 6 AT TM06 TO CYL
>0210    SEEK
>0220    READ D512 INTO RDIO
>0230    VERIFY D512 AT RDIO WITH WRIO
>0240    DECR TM00
>0250    IF TM00 > 0 GOTO 20
>0260    END
>DONE

*RUN 3
```

3. Modify the previous program to run on a dual controller system and share disk time with the other controller:

```
*MODIFY 3

ENTER STMNT'S

>0235   REL
>DONE

*RUN 3
```

D.13 DU11 SYNCHRONOUS LINE INTERFACE DEVICE ROUTINE

The DU11 Device Routine, whose filename is TDUAnm.MPG, supports high level I/O operations for the DU11 Single Line Synchronous Interface.

D.13.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Name and Unit Numbers

In reply to the "ASGN DEV:" message, one model name (DU11) is acceptable and may be accompanied by a maximum of 16 unit numbers. For this device unit numbers are ignored by the Device Routine but effectively act as a pass count for the number of times to repeat the program.

- Interface Addresses

The following are the values preset for the DU11 and may be altered from the console terminal following the "ASGN DEV:" message:

```
Device Register Base Address = 160010
Interrupt Vector Address = 300
Bus Request Priority = 5,5      (Read/Write)
```

- Symbolic Register Names

The symbolic names listed below may be used to reference the DU11 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register.

Name	Displ	Register
RCSR	+0	Receiver Status
RBUF	+2	Receiver Data Buffer
PCSR	+2	Parameter Control
TCSR	+4	Transmitter Status
TBUF	+6	Transmitter Data Buffer

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the DU11. Detailed explanations are listed in section D.13.2.

D.13 - DU11 Device Routine (Cont'd)

```
READ     READY    BITS     FDUPLX   CVSYNC
WRITE    SEND     EVEN     HDUPLX   WAIT
BREAK    RECV     ODD      NORMAL   NOWAIT
CALL     HANGUP   NOPAR    SYSTST   STATUS
LISTEN   CRESET   STRIP    PRESET   COUNTS
ANSWER   MODE     NSTRIP   GENPAR
```

- Information Words

The four words listed below are used to pass information between the user program and the device routine.

SYNC A one word location that contains the sync character that will be loaded into PCSR whenever PCSR's contents are changed. Preset to an octal 026 byte.

SCNT The word location that follows SYNC and contains the count for the number of sync characters to be transmitted when beginning a WRITE or a BREAK and in a mode other than Isochronous. Preset value is 5.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE and BREAK.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement and the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ
BYTES WR:	WRITE, BREAK
READ CMNDS:	READ
WRITE CMNDS:	WRITE
BREAK CMNDS:	BREAK
MISC CMNDS:	CALL, LISTEN, ANSWER, READY, SEND, RECV, HANGUP
ERRORS:	Hardware errors that resulted in an error report. Separate counts are maintained for Parity (PAR), Framing (FRM), Overrun (OVR), Data Set Change (DSC), Data Not Available (DNA) and I/O Timeout (T/O). The DATA error counter is incremented by invalid BITS codes, invalid MODE codes, and errors detected by the VERIFY statement.

INTERRUPTS: Number of entries into the Read and Write interrupt routines.

- OPSW Special Operation Bit Support

Bit 7 of the OPSW (SPOPER) is not interrogated by this Device Routine and its setting has no effect upon DU11 operation.

- Parameter Control Register

This Device Routine does not load the PCSR register until one of the statements, that affects PCSR's contents, is encountered. Since it is a write only register, a base value word is maintained by the Device Routine. When one of the applicable statements is processed, the appropriate bits will be modified in the base value and the entire word will be loaded into PCSR. This base value word is initially set to a value (037026) equivalent to the PRESET instruction being issued. Also, since the sync character is part of the PCSR register, the user must issue one of the instructions that load PCSR (other than PRESET) after he has specified a new sync character. The instructions that load PCSR are:

```
MODE      ODD
BITS      NOPAR
EVEN      PRESET
```

D.13.2 DESCRIPTION OF DU11 INSTRUCTIONS

The DU11 Device Routine supports execution of twenty nine MPG language statements. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2.

```
READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v
```

Transfers the number of data bytes specified by the first v from the receiver to the memory address specified by the second v. All desired DU11 parameters (mode, sync character, which duplex, character length, parity, etc.) must be set up before issuing this statement. Actions performed by this statement consist of setting Search Sync and Receiver Interrupt Enable in RCSR and the processing of the resulting interrupts. After the first interrupt is processed, the Data Set Change Interrupt Enable bit in RCSR is also set. When the last data byte has been received, Search Sync, Receiver Interrupt Enable, and Data Set Change Interrupt Enable will all be reset.

```
WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v
```

Transfers the number of data bytes specified by the first v from the memory location specified by the second v to the transmitter. All desired DU11 parameters (Clear To Send, mode, sync character, which duplex, character length, parity, etc.) must be set up before issuing this statement. Initially resets the Break bit and sets the Send bit in TCSR. Then, the Transmitter Interrupt Enable and Data Not Available Interrupt Enable bits are set in TCSR. When processing the transmitter interrupts, the specified number of the current sync character are issued unless in Isochronous mode. Next, the specified number of data bytes are transmitted. Following the last data byte, a pad byte of 0's is issued. On the interrupt following the pad byte, no data is transmitted but the Send, Transmitter Interrupt Enable, Data Not Available Interrupt Enable, and Break bits in TCSR are all reset followed by Request To Send in RCSR also being reset.

BREAK v

In operation this statement is very similar to the WRITE statement with a couple of exceptions. The v is used to specify a byte count for data bytes and a memory address is not needed. The Break bit in TCSR will be set prior to enabling Transmitter interrupts. Instead of getting data bytes from memory, a byte of all 1 bits (377) will be sent as the data byte. All other actions performed are identical to WRITE.

CALL

Sets Data Terminal Ready in RCSR and then tests Data Set Ready. Does not return to the user program until DSR is set. Used to wait while a call is being initiated from the DU11's MODEM.

LISTEN

Tests the Ring bit in RCSR and returns to the user program when Ring is set. Used to wait for an incoming call.

ANSWER

Sets Data Terminal Ready in RCSR and then tests Data Set Ready. Does not return to the user program until DSR is set. Used to wait until the connection has been completed for an incoming call.

READY

Tests the Carrier bit in RCSR. Returns to the user program when Carrier is set.

SEND

Sets the Request To Send bit in RCSR and then tests the Clear To Send bit. Returns to the user program when CTS is set. This prepares the MODEM for data transmission.

RECV

Resets the Request To Send bit in RCSR. This prepares the MODEM to receive data.

HANGUP

Resets the Request To Send bit in RCSR, delays approximately 15 milliseconds and then resets the Data Terminal Ready bit. This causes the call to be terminated by disconnecting the MODEM.

CRESET

Clears the DU11 by setting the Master Reset bit in TCSR. After setting MSTRST, a 20 microsecond delay will occur to allow for the one-shot in the DU11.

MODE v

Sets the Mode Select bits in PCSR to the value specified by the operand v. The three valid values for v and the modes they select are:

0	=	Isochronous
2	=	External Synchronous
3	=	Internal Synchronous

BITS v

Sets the Word Length Select bits in PCSR to the codes for the number of bits which is specified by v. The valid values for v and the bits per character selected are:

5	=	5 bits
6	=	6 bits
7	=	7 bits
10	=	8 bits
D8	=	8 bits

EVEN

Sets the character parity mode to even by setting both the Parity Enable and the Parity Sense Select bits in PCSR.

ODD

Sets the character parity mode to odd by setting the Parity Enable bit and resetting the Parity Sense Select bit in PCSR.

NOPAR

Disables character parity generation and checking by resetting both the Parity Enable and the Parity Sense Select bits in PCSR.

STRIP

Inhibits sync characters from being passed on as receiver data characters by setting the Strip Sync bit in RCSR.

NSTRIP

Permits sync characters to be recognized as receiver data characters by resetting the Strip Sync bit in RCSR.

FDUPLX

Enables Full Duplex mode of operation by resetting the Half Duplex bit in TCSR.

HDUPLX

Enables Half Duplex mode of operation by setting the Half Duplex bit in TCSR.

NORMAL

Sets the DU11's mode to Normal by resetting both Maintenance Mode Select bits in TCSR to 0's.

SYSTST

Sets the DU11's mode to System Test by setting both Maintenance Mode Select bits in TCSR to 1's.

PRESET

This statement sets all DU11 control bits and parameters to preset values with one statement. This statement is equivalent to resetting the Break bit in TCSR and then issuing the following individual statements:

```
LOAD SYNC WITH 026
LOAD SCNT WITH 5
MODE 3          (Internal Synchronous)
BITS D8
ODD
STRIP
FDUPLX
NORMAL
```

GENPAR v AT v

Using the byte count specified by the first v and beginning at the memory address specified by the second v, this instruction will convert the data bytes to the correct parity bit placement and character length currently in effect. Effectively generates parity the same as would be seen when the data is transmitted and received. Automatically adjusts to the number of character bits currently in effect (BITS or PRESET) and the current parity mode (ODD, EVEN, NOPAR, or PRESET). Useful for converting write data so that it may be compared with the data actually received.

CVSYNC v AT v

Using the byte count specified by the first v and the memory address specified by the second v, this instruction will scan the data bytes and complement any data bytes that match the current sync character. The comparison is based upon the character length currently in effect (BITS or PRESET). Useful for eliminating sync characters in random data or characters that become sync characters when the character length is changed.

WAIT

Standard implementation - see section D.1.1.

NOWAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

COUNTS

Standard implementation - see section D.1.1.

D.13.3 DU11 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. Refer to that section for general information concerning error reporting.

The following message may accompany any hardware type of error message:

ERROR BITS: ddd,ddd,ddd,etc.

This message indicates that one or more error status bits have been detected in the device registers. The codes ddd identify which error bits were found with codes being included if they have a value of 1. The following are the mnemonic codes supported for the ddd fields:

RX	= Receiver Error
OVR	= Overrun Error
FRM	= Framing Error
PAR	= Parity Error
DNA	= Data Not Available

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

DATA SET CHG INT ON READ

Indicates that the Data Set Change bit in RCSR was set when processing an interrupt during read data transfers. On the first interrupt for each READ instruction, this bit is ignored; however, it is tested on all subsequent interrupts.

ERROR ON READ DATA XFER

When processing a receiver interrupt during data transfers, one or more error bits were found set in RBUF. The error bit mnemonics will be displayed with the ERROR BITS message.

ERROR ON WRITE DATA XFER

When processing a transmitter interrupt during data transfers, the Data Not Available bit was set in TCSR. This bit's mnemonic will be displayed in the ERROR BITS message.

MODE NOT 0, 2 OR 3

While executing the program, the MODE statement was being processed and the operand supplied for the instruction did not have a value of 0, 2, or 3. This is an operator programming error and increments the DATA error counter.

OF CHAR BITS NOT 5, 6, 7 OR 8

While executing the program, the BITS statement was being processed and the operand supplied for the instruction did not have a value of 5, 6, 7, or D8 (octal 10). This is an operator programming error and increments the DATA error counter.

TIMEOUT ON I/O

After initiating a READ, WRITE, or BREAK operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately 3 minutes on the PDP-11/45 and with no other user programs executing. Increments the T/O error counter.

D.13.4 DU11 SAMPLE PROGRAMS

The following are sample DU11 programs that perform the functions indicated in their descriptions:

1. Using 201A MODEMS and an Auto-Answer Data Set, the following two programs will establish a connection between two DU11's on the same CPU. Messages will be issued to list the progress of each program. Note that these programs may also be used on separate CPU's.

```
*ENTER 1 AS CALLEE
?ASGN DEV: DU11
?RDIO = 256 / 0
?WRIO = 256 / 0
?DEV REG = 160010 / 160120
?INT VEC = 000300 / 470
?BUS REQ = 5,5 /

ENTER STMNT'S

>0010   CRESET
>0020   PRINT *WAITING FOR CALL
>0030   LISTEN
>0040   PRINT *I GOT A RING
>0050   ANSWER
>0060   PRINT *I ANSWERED
>0070   RECV
>0080   READY
>0090   PRINT *CALLEE IS READY
>0100   END
>DONE

*ENTER 2 AS CALLER
?ASGN DEV: DU11
?RDIO = 256 / 0
?WRIO = 256 / 0
?DEV REG = 160010 / 160130
?INT VEC = 000300 / 500
?BUS REQ = 5,5 /

ENTER STMNT'S

>0010   CRESET
>0020   PRINT *INITIATE CALL
>0030   CALL
>0040   PRINT *CONTACT MADE
>0050   READY
>0060   SEND
>0070   PRINT *CALLER IS READY
>0080   END
>DONE

*RUN 1,2
```

2. With a connection established between two DU11's, have one program (#9) send ASCII data to another (#8) and then have the same data sent back to the originating DU11 where the received data will be compared with the original data. The preset mode provides for 8 bits, odd parity, and Internal Synchronous mode.

```
*ENTER 8 AS DURD64
?ASGN DEV: DU11
?RDIO = 256 / 64
?WRIO = 250/ 0
?DEV REG = 160010 / 160120
?INT VEC = 000300 / 470
?BUS REQ = 5,5 /

ENTER STMNT'S

>0010   PRESET
>0020   RECV
>0030   FILL D64 AT RDIO WITH 0
>0040   READ D64 INTO RDIO
>0050   PRINT D64 AT RDIO IN ASCII
>0060   SEND
>0070   WRITE D64 FROM RDIO
>0080   END
>DONE

*ENTER 9 AS DUWR64
?ASGN DEV: DU11
?RDIO = 256 / 64
?WRIO = 256 / 64
?DEV REG = 160010 / 160130
?INT VEC = 000300 / 500
?BUS REQ = 5,5 /

ENTER STMNT'S

>0010   PRESET
>0020   FILL D64 AT WRIO WITH ASCII
>0030   CVSYNC D64 AT WRIO
>0040   SEND
>0050   WRITE D64 FROM WRIO
>0060   RECV
>0070   FILL D64 AT RDIO WITH 0
>0080   READ D64 INTO RDIO
>0090   PRINT D64 AT RDIO IN ASCII
>0100   GENPAR D64 AT WRIO
>0110   VERIFY D64 AT RDIO WITH WRIO
>0120   END
>DONE

*RUN 8,9
```

3. Modify the programs in Sample # 2 so that 6 bits with no parity will be used. Since six bit information is not directly printable, also modify the data print format. Note that the GENPAR statement will automatically convert the original transmit data to the same format as the received data for the comparison. Also, the CVSYNC statement will ensure that the six data bits of each data character does not match the sync character.

```
*MODIFY 8

>0012   BITS 6
>0014   NOPAR
>0050   PRINT D64 AT RDIO IN OCTAL
>DONE

*MODIFY 9

>0012   BITS 6
>0014   NOPAR
>0090   PRINT D64 AT RDIO IN OCTAL
>DONE

*RUN 8,9
```

4. Using the System Test mode, wrap random data within the DU11 to test the different parity modes of the receiver and the transmitter. Alter the number of and value of the sync characters sent. Use the CVSYNC instruction to ensure that sync characters are not included in the random data. Use the GENPAR instruction to convert the original write data to the same format as what should have been received. Identify and print the first 14 bytes of the data sent and the data received. Finally, repeat the program three times by using dummy unit numbers.

```
*ENTER 5 AS DUTEST
?ASGN DEV: DU11,0,1,2
?RDIO = 256 / 200
?WRIO = 256 / 200

ENTER STMNT'S

>0010   CRESET
>0020   PRESET
>0030   LOAD SYNC WITH O44
>0040   LOAD SCNT WITH D12
>0050   BITS 7
>0060   SYSTST
>0070   PRINT *ODD PARITY
>0080   ODD
>0090   LINK 2000
```

```
>0100   PRINT *EVEN PARITY
>0110   EVEN
>0120   LINK 2000
>0130   PRINT *NO PARITY
>0140   NOPAR
>0150   LINK 2000
>0160   GOTO 3000
>2000   FILL D200 AT WRIO WITH RANDOM
>2010   CVSYNC D200 AT WRIO
>2020   FILL D200 AT RDIO WITH 0
>2030   NOWAIT
>2040   READ D200 INTO RDIO
>2050   WRITE D200 FROM WRIO
>2060   WAIT
>2070   PRINT #WRIO =
>2080   PRINT D14 AT WRIO IN OCTAL
>2090   PRINT #RDIO =
>2100   PRINT D14 AT RDIO IN OCTAL
>2110   GENPAR D200 AT WRIO
>2120   VERIFY D200 AT RDIO WITH WRIO
>2130   RETURN
>3000   END
>DONE

*RUN 5
```

5. Modify Sample program # 4 to utilize 5 bits and operate in the Isochronous mode instead of Internal Synchronous.

```
*MODIFY 5

>0050   BITS 5
>0055   MODE 0
>DONE

*RUN 5
```

D.14 RK06 DISK'S DEVICE ROUTINE

The RK06 Device Routine, whose filename is TR6Anm.MPG, supports the operation of RK06 disk drives on the RK611 UNIBUS controller.

D.14.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Name and Unit Numbers

In reply to the "ASGN DEV:" message, one model name (RK06) is acceptable and may be accompanied by a maximum of 16 unit numbers. For this device the unit numbers must be in the range of 0 thru 7.

- Interface Addresses

The following are the values preset for the RK06 and may be altered from the console terminal following the "ASGN DEV:" message:

Device Register Base Address = 177440
Interrupt Vector Address = 210
Bus Request Priority = 5

- Symbolic Register Names

The symbolic names listed below may be used to reference the RK611 - RK06 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register.

Name	Displ	Register
RKC1	+0	Control and Status 1
RKWC	+2	Word Count
RKBA	+4	Unibus Address
RKDA	+6	Desired Sector/Track Address
RKC2	+10	Control and Status 2
RKDS	+12	Drive Status
RKER	+14	Error Register 1
RKAS	+16	Attention Summary/Offset
RKDC	+20	Desired Cylinder
NOTU	+22	Not used
RKDB	+24	Data Buffer
RKM1	+26	Maintenance 1
RKPO	+30	ECC Position

RKPA	+32	ECC Pattern
RKM2	+34	Maintenance 2
RKM3	+36	Maintenance 3

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the RK06. Detailed explanations are listed in section D.14.2.

READ
WRITE
RDHD
WRHD
WRCK
SPIN
SEEK
OFFSET
UNLOAD
RECAL
CRESET
DRESET
SRESET
STEPUP
STEPDN
SELDRI
WAIT
NOWAIT
STATUS
COUNTS
FMT22
FMT20
PAKACK
BADSEC
BAION
BAIOFF
CORON
COROFF
ODD
EVEN
REL

Certain of the above instructions control the setting of internal flag bits. The preset states of these bits are the same as though the following instructions were issued:

WAIT
CORON
ODD
FMT22
BAIOFF

- Information Words

The six words listed below are used to pass information between the user program and the device routine. All words are preset to 0's except RTRY which is preset to 3.

CYL A one word location that contains the cylinder number in bits 0-9 that will be used in subsequent I/O operations. The contents of CYL will be loaded into RKDC for all I/O operations that require a cylinder address.

HEAD The word location that follows CYL and contains the value of the head (track) number to be used in bits 0-2. The value in this word and in SECT will be merged and loaded into RKDA for all I/O operations that require a head and sector address.

SECT This word contains the sector number in bits 0-4. Located immediately following HEAD in memory.

MSGA This word contains the Message-A obtained with the SELDRI command.

MSGB This word contains the Message-B obtained with the SELDRI command.

RTRY This word specifies the number of additional attempts that the Device Routine will try on an I/O operation before deciding that it is unrecoverable. Applicable only to those types of errors that normally have a possibility of recovery.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, RDHD, WRHD, and WRCK .

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement and the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ, RDHD
BYTES WR:	WRITE, WRHD
BYTES CK:	WRCK
READ CMNDS:	READ, RDHD
WRITE CMNDS:	WRITE, WRHD
CHECK CMNDS:	WRCK
SEEK CMNDS:	SEEK, OFFSET, RECAL
MISC CMNDS:	DRESET, CRESET, SRESET, PAKACK, REL, UNLOAD, BADSEC, SPIN, SELDRI
DEV ERRORS:	All hardware errors that resulted in an error report. Correctable ECC errors and errors that were recoverable when retried will not increment this count.
CORR ECC ERRORS:	Number of ECC errors corrected by software.
DATA/OPER ERRORS:	Invalid unit number errors and errors detected by the VERIFY statement, or invalid drive numbers entered by the operator.
RETRYS:	The number of occurrences for each of the six retryable type of errors. Counts are maintained for DLT, DTE, HVRC, FER, DCK, and WCE.
TOTAL RETRYS:	Total number of retry attempts on all failing I/O operations.
INTERRUPTS:	Number of entries into the interrupt routine.

- Correctable ECC Errors

When encountering a correctable ECC error (DCK), this device routine will automatically correct the data. The contents of the ECC Position register (RKECPS) and the ECC Pattern register (RKECPT) will be used to determine the bits to be modified and their placement in the data field. After the correction has been applied, the CORR ECC count will be incremented and the data transfer will be resumed at its point of interruption.

This automatic data correction will not be performed if the word length is 18 bits (FMT22 reset), or if the user has entered the COROFF statement. In these cases, the DCK will be treated the same as the other retryable errors.

- Retryable Errors

Six errors are classified as retryable by this device routine. When one occurs (DLT, DTE, HVRC, FER, DCK, or WCE), the retry count contained in the interface word RTRY is checked for a zero value. If zero, an I/O Termination error is reported. If non-zero, the retry count for the error and the total retry error count are incremented by one and the I/O operation is re-issued. If the retry is successful, the program will proceed to the next statement in the user's program. If it was not successful, the retry count from RTRY is decremented, the total retry count is incremented, and the command is re-issued. This continues until either the retry is successful or the RTRY count goes to zero. When the latter occurs, the "EXHAUSTED RETRIES" error message will be issued and the DEV error count will be incremented.

When processing retries on failing I/O operations, the original function will just be re-issued. The use of Offset and Recalibrate commands are not employed during error recovery in this device routine.

- Dual Controller Operation

This device routine supports operation of disks connected to two systems through the dual controller option. If the disk is busy on the other controller when the device routine attempts to issue an I/O operation, it will receive a drive not available (bit 0 in RKDS will be reset). In order to make efficient use of the disk on a dual controller system, both programs operating the disk should issue Release commands through use of the REL statement provided by this device routine.

- Instructions And Interrupts

Certain of the I/O functions for the RK06 disks do not normally terminate with interrupts and will not have Interrupt Enable set when they are issued. These functions are termed Non-Interrupt functions and are:

DRESET
PAKACK
REL
CRESET
UNLOAD
SELDRI
SRESET
SPIN

After issuing the above functions, the device routine will delay a few microseconds and then check the status of the error bits. Any found to be set will be reported as a Non-Int Termination error.

The remaining I/O functions, which are listed below, utilize interrupts in their operations:

READ
WRITE
RDHD
WRCK
BADSEC
SEEK
OFFSET
RECAL
WRHD

For the above instructions, tests for error conditions will be made when the terminating interrupt is received.

- Accessing The Data Buffer Register

The contents of the Data Buffer (RKDB) register will be stored only when processing an interrupt with the Output Ready (OR) Bit set in the RKCS2 register. At all other times, this register will not be read and its contents will be displayed as 0's.

- OPSW Special Operation Bit Support

Bit 7 of the OPSW (SPOPER) is not supported by this device routine and its setting has no effect on disk operation.

D.14.2 DESCRIPTION OF RK06 INSTRUCTIONS

The RK06 Device Routine supports execution of thirty one MPG language statements. For certain functions (READ, WRITE, RDHD, WRHD, WRCK, and SEEK), the desired disk address must be loaded into CYL, HEAD, and SECT before performing those functions. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2. Note that if an odd byte count is supplied in any of the following instructions except RDHD, it will effectively be decremented by 1 before being used.

READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v

This instruction transfers the number of bytes, indicated by the first v, from the disk to memory beginning at the memory location specified by the second v.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Transfer the number of bytes from memory to disk beginning at the memory location indicated by the second v. This command will not allow transfers to write data onto the last track of the disk (see description of STEPUP command for further information about this last track.)

The following statements do not have default values for their operands:

RDHD v INTO v

Reads the three words of sector header data into memory using the Read Header command. In order to read a complete sector, the byte count must be 20 or 22, for 20 or 22 sectors per track.

WRHD v

Performs a format type of operation by using the Write Header command to write the header words on disk. The device routine sets up RKWC to be -60 or

-66, for 20 or 22 sectors per track. The data is written on the disk from the address specified by v.

If this command is used, the standard format program should be used to recreate valid header data on the disk pack.

WRCK v AT v

Compares the number of data field bytes in memory to those at the specified disk location. Uses the Write Check command.

SEEK

Performs a seek function to the disk cylinder specified by CYL. This instruction does not terminate until the seek is finished.

The following I/O commands do not utilize the contents of CYL, HEAD, and SECT in their operations:

OFFSET v

The least significant eight bits of the quantity specified by v will be loaded into the lower byte of the Attention Summary and Offset register. The offset command will be used. Examples of the offset obtained with different octal values of v are:

v	Microinches
010	+200
210	-200
020	+400
220	-400
030	+600
230	-600
040	+800
240	-800
060	+1200
260	-1200

SELDRI v

This statement issues the Select Drive command. The operand v is the number of the messages A and B to be returned by the drive and has values of 0-3. The

Message-A will be stored in the MSGA interface word. The Message-B will be stored in the MSGB interface word.

UNLOAD

Places the drive in the Standby state by issuing the Unload command. This command is terminated as soon as the Unload begins.

SPIN

Starts the spindle by issuing the Start Spindle command. This command is not terminated until the drive is on line.

RECAL

Results in the Recalibrate command being issued. This command is not terminated until the drive is on line.

CRESET

This statement performs a controller clear function by setting the CCLR bit in the RKCS1 register.

DRESET

Housekeeping of the drive is accomplished by this statement's issuing of the Drive Clear command.

SRESET

Performs a subsystem clear function by setting the SCLR bit in the RKCS2 register.

PAKACK

Causes the Pack Acknowledge command to be issued.

REL

Causes the drive to become deselected and releases the drive for another port by setting the RLS bit in the RKCS2 register.

The following instructions do not perform I/O operations:

STEPUP v

This command does not perform any I/O functions. It provides easy incrementing of the contents of CYL, HEAD, and SECT. The operand v is the number of sectors that these values are to be incremented. When the SECT word exceeds its maximum valid value, it will be set to 0 and the contents of HEAD will be incremented. When HEAD overflows, it will be set to 0 and CYL will be incremented.

The last track available to this command has an address of cylinder 410, head 1, and sector 21/19. This is not the last track on the disk. The last track has an address of 410,2,21/19 and is not accessed via this command; this track contains data about the disk pack and must not be changed with MPG software. The operand v may be of any value allowed by the disk hardware. If this command recognizes that the last track would be included in the transfer, it forces CYL, HEAD, and SECT to be 0.

Automatically adjusts to the number of sectors on a track (20 or 22) based upon the current setting defined by the FMT20/FMT22 statements.

Note that this instruction and STEPDN will operate with invalid values in CYL, HEAD, and SECT upon execution. Regardless of their initial contents, the contents of all three words will be valid when this instruction completes. This allows the use of the "FILL 6 AT CYL WITH RANDOM" statement followed by "STEPUP 0" to generate random disk addresses. However, one-sector transfers (512 bytes) should be used to avoid any possibility of the program being aborted due to transfers trying to write data on the last track.

STEPDN v

Similar to STEPUP but provides a decrementing capability. The operand v may be of any value allowed by the disk hardware. If this command recognizes that the transfer would begin below a disk address of 0,0,0, the disk address is forced to be 410,1,21/19, and the decrementing resumes from that address.

WAIT

Standard implementation - see section D.1.1.

NOWAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

Included with the standard display is the display of the current contents of the CYL, HEAD and SECT words. These values do not reflect the current position of the disk, but merely the contents of the three words.

COUNTS

Standard implementation - see section D.1.1.

The following statements define to the Device Routine the desired settings of certain bits in the device registers. These statements do not immediately alter the register bits but all subsequent I/O functions and other applicable statements will be performed with the bits set as defined.

FMT22

Sets the number of sectors on a track to 22 and the number of bits in the data field words to 16. FMT22 is the preset mode and results in the CFMT bit in the RKCS1 register being set to a 1.

FMT20

Sets the number of sectors on a track to 20 and the number of bits for the data field words to 18. Resets the CFMT bit in RKCS1 to a 0.

ODD

Causes the Parity Select (PAT) bit in RKMR1 to be in the 0 state on subsequent functions. (Preset mode)

EVEN

Causes PAT to be set to a 1.

BAION

Specifies that bus address incrementing is not to be performed by setting the BAI bit in the RKCS2 register to a 1.

BAIOFF

Results in BAI being reset to a 0 and bus address incrementing being performed. (Preset mode)

CORON

This statement activates the device routine's programmed function of correcting data on which a correctable ECC error was detected. When the disk detects an error and determines that it can be corrected, the device routine will perform the correction procedures, increment the CORR ECC error count, and continue running. (Preset mode)

COROFF

Disables the programmed recovery of correctable ECC errors. When this type of error is detected, an error display will result unless it is recoverable on retries.

BADSEC

This command causes a list of the sectors, which are flagged bad by manufacturing, to be printed on the LIST device. The sectors flagged bad by operating systems are not printed. The information used by this command is taken from the first ten sectors on the last track on the disk.

D.14.3 RK06 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. However, due to the nature of this device, additional information is provided to the user.

The I/O function type of instructions are considered to execute in two stages. The first is the time it takes to housekeep the disk. The second is after the point the specified function is issued and until after the function terminates. Since error conditions may occur in either stage, this device routine indicates which stage it was in when the error was detected. This is accomplished by including the appropriate message of the following two messages in the error display:

```
BEFORE ISSUING I/O CMND

AFTER ISSUING I/O CMND
```

If the error is detected before issuing the specified I/O command and MPG's Continue (CONT) command is issued, the program will be resumed at the statement on which the error was detected. If an AFTER error, the program will resume at the next statement in the user's program.

The following message may also be printed:

```
ERROR BITS:
ddd,ddd,ddd,ddd,ddd,ddd,etc.
```

This message indicates that one or more error status bits have been detected in the device registers. The codes ddd identify which error bits were found. A sixty four byte field will be used to list all or as many of the error bits as possible. In the following list some codes are distinguished by an asterisk (*). These codes will be included in the message if their bit values are 0. All other codes are included if they have a value of 1. The following are the mnemonic codes supported for the ddd fields:

CERR	= Controller Error
SPAR	= Serial Parity Error
CTO	= Controller Timeout Error
DLT	= Data Late
WCE	= Write Check Error
UPE	= Unibus Parity Error
NED	= Nonexistent Drive
NEM	= Nonexistent Memory
PGE	= Program Error
MDS	= Multiple Drive Select
UFE	= Unit Field Error

WRL	=	Write Locked
*DRDY	=	Drive Ready
*VV	=	Volume Valid
DROT	=	Drive Off Track
DSL	=	Drive Speed Loss
ACLO	=	Drive AC Low
DCK	=	Data Check
UNS	=	Unsafe
OPI	=	Operation Incomplete
DTE	=	Drive Timing Error
WLE	=	Write Lock Error
IDAE	=	Invalid Disk Address Error
COE	=	Cylinder Overflow Error
HVRC	=	Header VRC Error
BSE	=	Bad Sector Error
ECH	=	ECC Hard Error
DTYE	=	Drive Type Error
FMTE	=	Format Error
DRPAR	=	Drive Parity Error
NXF	=	Non-executable Function Error
SKI	=	Seek Incomplete Error
ILF	=	Illegal Function
ATA7	=	Attention Active Drive 7
ATA6	=	Attention Active Drive 6
ATA5	=	Attention Active Drive 5
ATA4	=	Attention Active Drive 4
ATA3	=	Attention Active Drive 3
ATA2	=	Attention Active Drive 2
ATA1	=	Attention Active Drive 1
ATA0	=	Attention Active Drive 0

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

EXHAUSTED RETRIES

Upon detecting one of the six retryable errors, the device routine has re-issued the failing I/O operation the number of times specified in RTRY without successful completeion of the operation. If RTRY is initially 0, this message will not occur.

INT WITHOUT ATA

This message indicates that an interrupt was received without the drive's ATA bit being set when it was expected to be set.

INT WITHOUT DI

This message indicates that an interrupt was received without the DI bit in the RKCS1 being set when it was expected to be set.

INV ECC BIT PATTERN

When ready to apply the correction to the data on a correctable ECC error, the RKECPS Bit Pattern register was found to contain all 0's.

INV ECC BIT POSITION

When ready to apply the correction to the data on a correctable ECC error, the RKECPT Bit Position register was found to contain a value greater than octal 10041.

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erroneous unit number. Increments the data/oper error count.

I/O TERMINATION ERROR

Error bits have been detected in the device registers when processing a termination interrupt. This message occurs for all non-retryable errors and retryable errors when RTRY is set to 0.

NON-INT I/O TERMINATION ERROR

Error bits were found set in the device registers a few microseconds after issuing one of the Non-Interrupt type of I/O functions (PAKACK, REL, etc).

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately ten seconds when on the PDP-11/45 and with no other user programs executing.

T/O ON CRESET

This error, which indicates that Ready (RDY) did not set within a few milliseconds after setting the CCLR bit in RKCS1, may occur when the CRESET statement is issued.

T/O ON SRESET

This error, which indicates that Ready (RDY) did not set within a few milliseconds after setting the SCLR bit in RKCS2, may occur when the SRESET statement is issued.

T/O ON REL

This error, which indicates that Ready (RDY) did not set within a few milliseconds after setting the RLS bit in RKCS2, may occur when the REL statement is issued.

UNEXP ATA COND

When processing an interrupt after issuing an I/O function that should not cause an ATA condition, the disk's ATA bit was found set.

UNKNOWN ERROR CONDITION

If the CERR bit of RKCS1 is set, the program examines the various error bits to determine the cause of the error. If no other error bit is found to be set, this message is printed.

SVAL BIT NOT SET IN RKDS

When interrupts are serviced, the SVAL bit of RKDS is checked. If the bit is reset, this message is given. This message is not printed if the interrupt is the 2nd one from a SEEK, OFFSET, or RECAL.

```
OUTPUT NOT READY ON RDHD COMMAND
```

When the RDHD command is executed, the OR bit of RKCS2 is checked. If the bit is reset, this message is printed.

```
SELDRI COMMAND HAS INVALID CODE
```

The operand entered for the SELDRI command is checked for a value of 0, 1, 2, or 3. If the value is not in this range, this message is printed.

```
BAD SECTORS (OCTAL DATA)
PACK #
THIS IS AN ALIGNMENT PACK
RDIO IS TOO SMALL FOR "BADSEC" COMMAND
CAN NOT READ LAST TRACK
NONE
END OF BAD SECTORS
```

These messages may be printed when the BADSEC command is executed. Each message describes the condition which caused it to occur.

D.14.4 RK06 SAMPLE PROGRAMS

The following are sample RK06 programs that perform the functions indicated in their descriptions:

1. The following program will verify all tracks on the entire disk by writing 1 bits to the data fields. Retries will be inhibited so as to isolate marginal tracks:

```
*ENTER 1 AS TRKCK
?ASGN DEV: RK06,0
?RDIO = 256 / 0
?WRIO = 256 / 5632
?DEV REG = 177440 /
?INT VEC = 000210 /
?BUS REQ = 5   /

ENTER STMNT'S

>0010   LOAD RTRY WITH 0
>0020   FILL D5632 AT WRIO WITH 177777
>0030   WRITE D5632 FROM WRIO
>0040   WRCK D5632 AT WRIO
>0050   STEPUP D11
>0060   IF 6 AT CYL <> 0 GOTO 30
>0070   END
>DONE

*RUN 1
```

2. Do 200 pairs of random seeks to an RK06 disk by using the STEPUP instruction to generate valid disk addresses from random data. Intersperse other instructions to perform data transfers of random data and do data validation at these disk addresses:

```
*ENTER 3 AS RKRAND
?ASGN DEV: RK06,5
?RDIO = 256 / 512
?WRIO = 256 / 512

ENTER STMNT'S

>0010   LOAD TM00 WITH D200
>0020   FILL 6 AT CYL WITH RANDOM
>0030   STEPUP 0
>0040   MOVE 6 AT CYL TO TM03
>0050   FILL D512 AT WRIO WITH RANDOM
>0060   SEEK
>0070   WRITE D512
>0080   WRCK D512 AT WRIO
>0090   FILL 6 AT CYL WITH RANDOM
```

```
>0100   STEPUP 0
>0110   IF 6 AT CYL = TM03 GOTO 90
>0120   MOVE 6 AT CYL TO TM06
>0130   SEEK
>0140   WRITE D512
>0150   WRCK D512 AT WRIO
>0160   MOVE 6 AT TM03 TO CYL
>0170   SEEK
>0180   READ D512
>0190   VERIFY D512 AT RDIO WITH WRIO
>0200   MOVE 6 AT TM06 TO CYL
>0210   SEEK
>0220   READ D512 INTO RDIO
>0230   VERIFY D512 AT RDIO WITH WRIO
>0240   DECR TM00
>0250   IF TM00 > 0 GOTO 20
>0260   END
>DONE

*RUN 3
```

D.15 RP02/RP03 DISKS' DEVICE ROUTINE

The RP02/RP03 Device Routine, whose filename is TR3Anm.MPG, supports the operation of RP02 and RP03 disk drives on the standard RP11 controller.

This Device Routine will automatically adjust its operation based upon the device assigned to it. For the RP03 it will adjust to a maximum of 406 cylinders.

D.15.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

In reply to the "ASGN DEV:" message, two model names (RP02 and RP03) are acceptable and may be accompanied by a maximum of 16 unit numbers. For this device each unit number must be in the range of 0 thru 7.

- Interface Addresses

The following are the values preset for the RP02/RP03 and may be altered from the console terminal following the "ASGN DEV:" message:

Device Register Base Address = 176710
Interrupt Vector Address = 254
Bus Request Priority = 5

- Symbolic Register Names

The symbolic names listed below may be used to reference the RP02/RP03 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register.

Name	Displ	Register
RPDS	+0	Device Status
RPER	+2	Error
RPCS	+4	Control and Status
RPWC	+6	Word Count
RPBA	+10	Bus Address
RPCA	+12	Cylinder Address
RPDA	+14	Disk Address
RPM1	+16	Maintenance 1
RPM2	+20	Maintenance 2
RPM3	+22	Maintanance 3
SUCA	+24	Selected Unit Cylinder Address
SILO	+26	Silo Memory

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the RP02/RP03. Detailed explanations are listed in section D.15.2.

READ
WRITE
RDNOSK
WRNOSK
WRCK
SEEK
HOMESK
RECAL
IDLE
CRESET
HDRON
HDROFF
MODE11
MODE10
STEPUP
STEPDN
WAIT
NOWAIT
STATUS
COUNTS

Certain of the above instructions control the setting of internal flag bits. The preset states of these bits are the same as though the following instructions were issued:

HDROFF
MODE11
WAIT

- Information Words

The six words listed below are used to pass information between the user program and the device routine. All words are preset to 0's except RTRY which is preset to 3.

CYL A one word location that contains the cylinder number in bits 0 - 8 that will be used in subsequent I/O operations. The contents of CYL will be loaded into RPCA for all I/O operations that require a cylinder address.

HEAD The word location that follows CYL and contains the value of the head (track) number to be used in bits 0-4. The value in this word and in SECT will be merged and loaded into RPDA for all I/O operations that require a head and sector address.

SECT This word contains the sector number in bits 0-4. Located immediately following HEAD in memory.

RTRY This word specifies the number of additional attempts that the Device Routine will try on an I/O operation before deciding that it is unrecoverable. Applicable only to those types of errors that normally have a possibility of recovery.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, RDNOSK, WRNOSK, and WRCK.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement and the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES RD:	READ, RDNOSK
BYTES WR:	WRITE, WRNOSK
BYTES CK:	WRCK
READ CMNDS:	READ, RDNOSK
WRITE CMNDS:	WRITE, WRNOSK
CHECK CMNDS:	WRCK
SEEK CMNDS:	SEEK, HOMESK, RECAL
MISC CMNDS:	IDLE, CRESET
DEV ERRORS:	All hardware errors that resulted in an error report. Errors that were recoverable when retried will not increment this count.
DATA ERRORS:	Invalid unit number errors and errors detected by the VERIFY statement.
RETRYS:	The number of occurrences for each of the five retryable type of errors. Counts are maintained for TIMEE, CSME, WPE, LPE, and WCE.
TOTAL RETRYS:	Total number of retry attempts on all failing I/O operations.
INTERRUPTS:	Number of entries into the interrupt routine.

- Retryable Errors

Five errors are classified as retryable by this device routine. When one occurs (TIMEE, CSME, WPE, LPE, or WCE), the retry count contained in the interface word RTRY is checked for a zero value. If zero, an I/O Termination error is reported. If non-zero, the retry count for the error and the total retry error count are incremented by one and the I/O operation is re-issued. If the retry is successful, the program will proceed to the next statement in the user's program. If it was not successful, the retry count from RTRY is decremented, the total retry count is incremented, and the command is re-issued. This continues until either the retry is successful or the RTRY count goes to zero. When the latter occurs, the "EXHAUSTED RETRIES" error message will be issued and the DEV error count will be incremented.

- Disk Types

  This device routine does not interrogate the Device Status register to determine which type of disk it is operating on. It relies solely on the model number assigned to the program. If an RPO2 has been assigned but it is actually an RPO3, the device routine will process it as an RPO2.

- Accessing The Data Buffer Register

  The Data Buffer (RPDB) register will not be accessed by this device routine and therefore will not be included in any displays.

- OPSW Special Operation Bit Support

  Bit 7 of the OPSW (SPOPER) is not supported by this device routine and its setting has no effect on disk operation.

D.15.2 DESCRIPTION OF RPO2/RPO3 INSTRUCTIONS

The RPO2/RPO3 Device Routine supports execution of twenty MPG language statements. For certain functions (READ, WRITE, WRCK, and SEEK), the desired disk address must be loaded into CYL, HEAD, and SECT before performing those functions. For the RDNOSK and WRNOSK functions, SECT must contain the desired sector number.

In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2. Note that if an odd byte count is supplied in any of the following instructions, it will effectively be decremented by 1 before being used.

```
READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v
```

This instruction transfers the number of bytes, indicated by the first v, from the disk to memory beginning at the memory location specified by the second v.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Transfer the number of bytes from memory to disk beginning at the memory location indicated by the second v.

The following statements do not have default values for their operands:

RDNOSK v INTO v

Similar to the READ statement but issues the Read (No Seek) command and does not have default operands. Reading will begin on the previously selected head and cylinder and at the sector currently specified by SECT.

Note: The use of RDNOSK and WRNOSK should be avoided when executing multiple programs on the same drive. In this situation I/O operations are interlaced and since these commands do not perform implied seeks, another program may position the drive to a cylinder and/or a head other than the one desired for these commands.

WRNOSK v FROM v

Similar to the WRITE statement but issues the Write (No Seek) command and does not have default operands. Comments for RDNOSK also apply to this statement.

WRCK v AT v

Compares the number of data bytes in memory to those at the specified disk location. Uses the Write Check command.

SEEK

Performs a seek function to the disk location specified by CYL, HEAD, and SECT. This instruction does not terminate until the seek is finished.

HOMESK

Performs a drive restore operation by issuing the Home Seek command.

RECAL

Identical to HOMESK.

IDLE

This statement performs a controller clear function by issuing the Idle command.

CRESET

Identical to IDLE.

The following instructions do not perform I/O operations:

STEPUP v

This command does not perform any I/O functions. It provides easy incrementing of the contents of CYL, HEAD, and SECT. The operand v is the number of sectors that these values are to be incremented. When the SECT word exceeds its maximum valid value, it will be set to 0 and the contents of HEAD will be incremented. When HEAD overflows, it will be set to 0 and CYL will be incremented. When incrementing past the last sector on disk, all three values will be set to 0's. Automatically adjusts to the number of cylinders (203 for the RP02 vs. 406 for the RP03).

Note that this instruction and STEPDN will operate with invalid values in CYL, HEAD, and SECT upon execution. Regardless of their initial contents, the contents of all three words will be valid when this instruction completes. This allows the use of the "FILL 6 AT CYL WITH RANDOM" statement followed by "STEPUP 0" to generate random disk addresses.

STEPDN v

Similar to STEPUP but provides a decrementing capability. When all three values are 0, the next decrement will result in the decimal values for the three words of 202/405, 19, 9.

WAIT

Standard implementation - see section D.1.1.

NOWAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

Included with the standard display is the display of the current contents of the CYL, HEAD and SECT words. These values do not reflect the current position of the disk, but merely the contents of the three words.

COUNTS

Standard implementation - see section D.1.1.

The following statements define to the Device Routine the desired settings of certain bits in the device registers. These statements do not immediately alter the register bits but all subsequent I/O functions and other applicable statements will be performed with the bits set as defined.

MODE11

Sets the data word format to 16 bits (PDP-11 mode) by resetting the MODE bit in RPCS to 0. (Preset mode)

MODE10

Sets the data word format to 36 bits (PDP-10/15 mode) by setting the MODE bit in RPCS to a 1.

HDRON

Allows for Header operations on data transfer commands by setting both the Header bit (HDR) and the Mode bit (MODE) in RPCS to 1's.

HDROFF

Indicates that Header operations are not to be performed by resetting HDR and MODE to 0. (Preset mode)

D.15.3 RP02/RP03 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2.

If error bits are present in the device's registers on any hardware error, the following message will be included:

```
  ERROR BITS:
  ddd,ddd,ddd,ddd,ddd,ddd,etc.
```

The codes ddd identify which error bits were found. A sixty four byte field will be used to list all or as many of the error bits as possible. In the following list some codes are distinguished by an asterisk (*). These codes will be included in the message if their bit values are 0. All other codes are included if they have a value of 1. The following are the mnemonic codes supported for the ddd fields:

ERR	=	Error
HE	=	Hard Error
WPV	=	Write Protect Violation
FUV	=	File Unsafe Violation
NXC	=	Non-existent Cylinder
NXT	=	Non-existent Track
NXS	=	Non-existent Sector
PROGE	=	Program Error
FMTE	=	Format Error
MODEE	=	Mode Error
LPE	=	Longitudinal Parity Error
WPE	=	Word Parity Error
CSME	=	Checksum Parity Error
TIMEE	=	Timing Error
WCE	=	Write Check Error
NXME	=	Non-existent Memory Error
EOPE	=	End Of Pack Error
DSKERR	=	Disk Error
*SURDY	=	Selected Unit Ready

*SUOL	= Selected Unit On Line
HNF	= Header Not Found
SUSI	= Selected Unit Seek Incomplete
SUFU	= Selected Unit File Unsafe
SUWP	= Selected Unit Write Protected
ATTN7	= Attention Drive 7
ATTN6	= Attention Drive 6
ATTN5	= Attention Drive 5
ATTN4	= Attention Drive 4
ATTN3	= Attention Drive 3
ATTN2	= Attention Drive 2
ATTN1	= Attention Drive 1
ATTN0	= Attention Drive 0

The unique error messages, which may occur in the use of this device routine, are as follows:

ATTN NOT SET

This message indicates that an attention interrupt on a SEEK, HOMESK, or RECAL instruction was received without the drive's ATTN bit being set and it was expected to be set. Before reporting this error, it has been ascertained that another drive's ATTN did not cause the interrupt.

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

EXHAUSTED RETRIES ON xxxx

Upon detecting one of the five retryable errors indicated by xxxx, the device routine has re-issued the failing I/O operation the number of times specified in RTRY without successfully completing the operation. If RTRY is initially 0, this message will not occur. The codes for xxxx are:

TMEE	= Timing Error
CSME	= Checksum Error
WPE	= Word Parity Error
LPE	= Longitudinal Parity Error
WCE	= Write Check Error

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erroneous unit number. Increments the data error count.

I/O TERMINATION ERROR

Error bits have been detected in the device registers when processing a termination interrupt. This message occurs for all non-retryable errors and retryable errors when RTRY is set to 0.

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately 10 seconds when on the PDP-11/45 and with no other user programs executing.

T/O ON IDLE/CRESET

This error, which indicates that Ready (RDY) did not set within a few milliseconds after issuing the Idle command, may occur when either the IDLE or the CRESET statements are issued.

UNIT OFF-LINE

Immediately prior to initiating an I/O operation, the Selected Unit On Line bit (SUOL) in the RPDS register was found to be reset.

UNIT NOT RDY

Immediately prior to initiating an I/O operation, the Selected Unit Ready bit (SURDY) in the RPDS register was found to be reset.

UNSAFE ON INITIATION

Prior to initiating an I/O operation, the Selected Unit Unsafe bit (SUFU) in RPDS was found to be set.

D.15.4 RP02/RP03 SAMPLE PROGRAMS

The following are sample RP02/RP03 programs that perform the functions indicated in their descriptions:

1. Verify the ability to read and write the header words on all sectors on cylinder octal 123 and head 3. Use the PRINT instruction to display the before and after data:

```
*ENTER 3 AS HDRTST
?ASGN DEV: RP03,3
?RDIO = 256 / 60
?WRIO = 256 / 60
?DEV REG = 176710 /
?INT VEC = 000254 /
?BUS REQ = 5    /

ENTER STMNT'S

>0010    LOAD CYL WITH 126
>0020    LOAD HEAD WITH 3
>0030    HDRON
>0040    READ D60 INTO RDIO
>0050    PRINT *ORIG HDR DATA:
>0060    PRINT D60 AT RDIO IN OCTAL
>0070    FILL D60 AT WRIO WITH 177777
>0080    WRITE D60 FROM WRIO
>0090    FILL D60 AT WRIO WITH 0
>0100    READ D60 INTO WRIO
>0110    PRINT *TEST HDR DATA:
>0120    PRINT D60 AT WRIO IN OCTAL
>0130    WRITE D60 FROM RDIO
>0140    READ D60 INTO WRIO
>0150    PRINT *RESTORED ORIG HDR DATA:
>0160    PRINT D60 AT WRIO IN OCTAL
>0170    VERIFY D60 AT WRIO WITH RDIO
>0180    HDROFF
>0190    END
>DONE

*RUN 3
```

D.15 - RP02/RP03 Device Routine (Cont'd)

2. The following program will verify all tracks on the entire disk by writing 1 bits to the data fields. Retries will be inhibited so as to isolate marginal tracks:

```
*ENTER 1 AS TRKCK
?ASGN DEV: RP02,0
?RDIO = 256 / 0
?WRIO = 256 / 5120

ENTER STMNT'S

>0010    LOAD RTRY WITH 0
>0020    FILL D5120 AT WRIO WITH 177777
>0030    WRITE D5120 FROM WRIO
>0040    WRCK D5120 AT WRIO
>0050    STEPUP D10
>0060    IF 6 AT CYL <> 0 GOTO 30
>0070    END
>DONE

*RUN 1
```

3. Do 200 pairs of random seeks to an RP03 disk by using the STEPUP instruction to generate valid disk addresses from random data. Intersperse other instructions to perform data transfers of random data and do data validation at these disk addresses:

```
*ENTER 7 AS R3RAND
?ASGN DEV: RP03,5
?RDIO = 256 / 512
?WRIO = 256 / 512

ENTER STMNT'S

>0010    LOAD TM00 WITH D200
>0020    FILL 6 AT CYL WITH RANDOM
>0030    STEPUP 0
>0040    MOVE 6 AT CYL TO TM03
>0050    FILL D512 AT WRIO WITH RANDOM
>0060    SEEK
>0070    WRNOSK D512 FROM WRIO
>0080    WRCK D512 AT WRIO
>0090    FILL 6 AT CYL WITH RANDOM
>0100    STEPUP 0
>0110    IF 6 AT CYL = TM03 GOTO 90
>0120    MOVE 6 AT CYL TO TM06
>0130    WRITE D512 FROM WRIO
>0140    WRCK D512 AT WRIO
>0150    MOVE 6 AT TM03 TO CYL
>0160    READ D512 INTO RDIO
>0170    VERIFY D512 AT RDIO WITH WRIO
>0180    MOVE 6 AT TM06 TO CYL
```

D.15 - RPO2/RPO3 Device Routine (Cont'd)

```
        >0190   SEEK
        >0200   RDNOSK D512 INTO RDIO
        >0210   VERIFY D512 AT RDIO WITH WRIO
        >0220   DECR TMOO
        >0230   IF TMOO > 0 GOTO 20
        >0240   END
        >DONE

        *RUN 7


>
```

D.16 RS03/RS04 DISKS' DEVICE ROUTINE

The RS03/RS04 Device Routine, whose filename is TRSAnm.MPG, supports the operation of RS03 and RS04 disk drives on the standard RH11 MASSBUS controller and on the 11/70 version (RH70) of the RH11.

This Device Routine will automatically adjust its operation based upon the device assigned to it and the CPU that it is operating on. Error bits displayed in the ERROR BITS message will be applicable to the device and the controller. Error bits will be included as necessary for common errors and RH70 errors. Also, two additional device registers are supported for the RH70.

D.16.1 PRESET VALUES AND SUPPORT SUMMARY

- Valid Model Names and Unit Numbers

In reply to the "ASGN DEV:" message, two model names (RS03 and RS04) are acceptable and may be accompanied by a maximum of 16 unit numbers. For this device the unit numbers must be in the range of 0 thru 7.

- Interface Addresses

The following are the values preset for the RS03/RS04 and may be altered from the console terminal following the "ASGN DEV:" message:

Device Register Base Address = 172040
Interrupt Vector Address = 204
Bus Request Priority = 5

- Symbolic Register Names

The symbolic names listed below may be used to reference the RH11/RH70 - RS03/RS04 device registers in MPG instructions. The octal displacement associated with each name is the value that will be added to the device register base address to obtain the actual memory address of the desired register. Note that the last two (RSAE and RSC3) will be supported only when running on an 11/70.

Name	Displ	Register
RSC1	+0	Control and Status 1
RSWC	+2	Word Count
RSBA	+4	Unibus Address

RSDA	+6	Desired Sector/Track Address
RSC2	+10	Control and Status 2
RSDS	+12	Drive Status
RSER	+14	Error Register
RSAS	+16	Attention Summary
RSLA	+20	Look Ahead
RSDB	+22	Data Buffer
RSMR	+24	Maintenance
RSDT	+26	Drive Type
RSAE	+30	Bus Address Extension
RSC3	+32	Control and Status 3

- Supported Instructions Summary

The following is a summary of the instructions supported by MPG for the RS03/RS04. Detailed explanations are listed in section D.16.2.

```
READ      CRESET    WAIT      BAION
WRITE     DRESET    NOWAIT    BAIOFF
WRCK      STEPUP    APORT     ODD
SEARCH    STEPDN    BPORT     EVEN
STATUS    COUNTS
```

Certain of the above instructions control the setting of internal flag bits. The preset states of these bits are the same as though the following instructions were issued:

```
WAIT      ODD       BAIOFF    APORT
```

- Information Words

The six words listed below are used to pass information between the user program and the device routine. All words are preset to 0's except RTRY which is preset to 3.

TRAK or HEAD A one word location that contains the track number in bits 0-5 that will be used in subsequent I/O operations. The contents of TRAK will be loaded into RSDA for all I/O operations that require a track address.

SECT This word contains the sector number in bits 0-5. Located immediately following TRAK in memory.

RTRY This word specifies the number of additional attempts that the Device Routine will try on an I/O operation before deciding that it is unrecoverable. Applicable only to those types of errors that normally have a possibility of recovery.

SIZE Standard implementation - see section D.1.1. Contents are updated by READ, WRITE, and WRCK.

ERR Standard implementation - see section D.1.1.

- Statistical Information

Through use of the COUNTS statement and the REPORT command, statistical information for the program can be displayed on the MPG print device. This information consists of octal formatted binary counts under different catagories. The catagories and the functions from which data will be included under each are:

BYTES READ
BYTES WRITTEN
BYTES CHECKED
READ COMMAND
WRITE COMMAND
CHECK COMMAND
SEARCH COMMAND
DRIVE CLEAR COMMAND

DEV ERRORS: All hardware errors that resulted in an error report. Errors that were recoverable when retried will not increment this count.

DATA/OPER ERRORS: Invalid unit number errors and errors detected by the VERIFY statement.

RETRYS: The number of occurrences for each of the four retryable type of errors. Counts are maintained for DLT, DTE, DCK, and WCE.

TOTAL RETRYS: Total number of retry attempts on all failing I/O operations.

INTERRUPTS: Number of entries into the interrupt routine.

- Retryable Errors

Four errors are classified as retryable by this device routine. When one occurs (DLT, DTE, DCK, or WCE), the retry count contained in the interface word RTRY is checked for a zero value. If zero, an I/O Termination error is reported. If non-zero, the retry count for the error and the total retry error count are incremented by one and the I/O operation is re-issued. If the retry is successful, the program will proceed to the next statement in the user's program. If it was not successful, the retry count from RTRY is decremented, the total retry count is incremented, and the command is re-issued. This continues until either the retry is successful or the RTRY count goes to zero. When the latter occurs, the "EXHAUSTED RETRIES" error message will be issued and the DEV error count will be incremented.

When processing retries on failing I/O operations, the original fu..ction will just be re-issued.

- Disk Types

This device routine does not interrogate the Drive Type register to determine which type of disk it is operating on. It relies solely on the model number assigned to the program. If an RS03 has been assigned but it is actually an RS04, the device routine will process it as an RS03.

- Instructions And Interrupts

Two of the I/O functions for the RS03/RS04 disks will not terminate with interrupts and will not have Interrupt Enable set when they are issued. These functions are termed Non-Interrupt functions and are:

DRESET CRESET

After issuing the above functions, the device routine will delay a few microseconds and then check the status of the error bits. Any found to be set will be reported as a Non-Int Termination error.

The remaining I/O functions, which are listed below, utilize interrupts in their operations:

READ WRITE WRCK SEARCH

For the above instructions, tests for error conditions will be made when the terminating interrupt is received.

- Accessing The Data Buffer Register

The contents of the Data Buffer (RSDB) register will be stored only when processing an interrupt with the Output Ready (OR) bit set in the RSCS2 register. At all other times, this register will not be read and its contents will be displayed as 0's.

- OPSW Special Operation Bit Support

  Bit 7 of the OPSW (SPOPER) is not supported by this device routine and its setting has no effect on disk operation.

D.16.2 DESCRIPTION OF RS03/RS04 INSTRUCTIONS

The RS03/RS04 Device Routine supports execution of eighteen MPG language statements. For certain functions (READ, WRITE, WRCK, and SEARCH), the desired disk address must be loaded into TRAK and SECT before performing those functions. In the following descriptions, data shown enclosed within parentheses indicates the default values if nothing is entered for the statement's operands. The v is used to indicate a variable operand as defined in Appendix B.2. Note that if an odd byte count is supplied in any of the following instructions, it will effectively be decremented by 1 before being used.

READ (D256 INTO RDIO)
READ v (INTO RDIO)
READ v INTO v

This instruction transfers the number of bytes, indicated by the first v, from the disk to memory beginning at the memory location specified by the second v.

WRITE (D256 FROM WRIO)
WRITE v (FROM WRIO)
WRITE v FROM v

Transfer the number of bytes from memory to disk beginning at the memory location indicated by the second v.

The following statements do not have default values for their operands:

WRCK v AT v

Compares the number of data field bytes in memory to those at the specified disk location. Uses the Write Check Data command.

SEARCH

Performs a search function to the sector specified by SECT. This instruction does not terminate until the search is finished.

The following I/O commands do not utilize the contents of TRAK and SECT in their operations:

CRESET

This statement performs a controller clear function by setting the CLR bit in the RSCS2 register.

DRESET

Housekeeping of the drive is accomplished by this statement's issuing of the Drive Clear command.

The following instructions do not perform I/O operations:

STEPUP v

This command does not perform any I/O functions. It provides easy incrementing of the contents of TRAK and SECT. The operand v is the number of sectors that these values are to be incremented. When the SECT word exceeds its maximum valid value, it will be set to 0 and the contents of TRAK will be incremented. When incrementing past the last sector on disk, both values will be set to 0.

Note that this instruction and STEPDN will operate with invalid values in TRAK and SECT upon execution. Regardless of their initial contents, the contents of both words will be valid when this instruction completes. This allows the use of the "FILL 4 AT TRAK WITH RANDOM" statement followed by "STEPUP 0" to generate random disk addresses.

STEPDN v

Similar to STEPUP but provides a decrementing capability. When both values are 0, the next decrement will result in the decimal values for the two words of 63.

WAIT

Standard implementation - see section D.1.1.

NOWAIT

Standard implementation - see section D.1.1.

STATUS

Standard implementation - see section D.1.1.

Included with the standard display is the display of the current contents of the TRAK and SECT words. These values do not reflect the current position of the disk, but merely the contents of the two words.

COUNTS

Standard implementation - see section D.1.1.

The following statements define to the Device Routine the desired settings of certain bits in the device registers. These statements do not immediately alter the register bits but all subsequent I/O functions and other applicable statements will be performed with the bits set as defined.

APORT

Indicates that the value of the PSEL bit in RSCS1 is to be set to 0 (UNIBUS Port A). (Preset mode)

BPORT

PSEL will be set to a 1 (UNIBUS Port B) on subsequent functions.

ODD

Causes the Parity Select (PAT) bit in RSCS2 to be in the 0 state on subsequent functions. (Preset mode)

EVEN

Causes PAT to be set to a 1.

BAION

Specifies that bus address incrementing is not to be performed by setting the BAI bit in the RSCS2 register to a 1.

BAIOFF

Results in BAI being reset to a 0 and bus address incrementing being performed. (Preset mode)

D.16.3 RS03/RS04 ERROR INFORMATION

This device routine processes errors in the standard fashion described in section D.1.2. However, due to the nature of this device, additional information is provided to the user.

The I/O function type of instructions are considered to execute in two stages. The first is the time it takes to housekeep the disk prior to the output of the specified command. The second is after the point the specified function is issued and until after the function terminates. Since error conditions may occur in either stage, this device routine indicates which stage it was in when the error was detected. This is accomplished by including the appropriate message of the following two messages in the error display:

```
BEFORE ISSUING I/O CMND

AFTER ISSUING I/O CMND
```

If the error is detected before issuing the specified I/O command and MPG's Continue (CONT) command is issued, the program will be resumed at the statement on which the error was detected. If an AFTER error, the program will resume at the next statement in the user's program.

Also included with every error display, other than the invalid unit number error, is the following message:

```
ERROR BITS:
ddd,ddd,ddd,ddd,ddd,ddd,etc.
```

This message indicates that one or more error status bits have been detected in the device registers. The codes ddd identify which error bits were found. A sixty four byte field will be used to list all or as many of the error bits as possible. In the following list some codes are distinguished by an asterisk (*). These codes will be included in the message if their bit values are 0. All other codes are included if they have a value of 1. The following are the mnemonic codes supported for the ddd fields:

Error bits common to all devices:

SC	=	Special Condition
TRE	=	Transfer Error
MCPE	=	Massbus Control Bus Parity Error
*DVA	=	Drive Available
*RDY	=	Ready
DLT	=	Data Late
WCE	=	Write Check Error
UPE	=	Unibus Parity Error
NED	=	Nonexistent Drive
NEM	=	Nonexistent Memory
PGE	=	Program Error
MXF	=	Missed Transfer
MDPE	=	Massbus Data Bus Parity Error
ATA	=	Attention Active
ERR	=	Error
PIP	=	Positioning In Progress
*MOL	=	Medium On-Line
*DPR	=	Drive Present
*DRY	=	Drive Ready
DCK	=	Data Check
UNS	=	Unsafe
OPI	=	Operation Incomplete
DTE	=	Drive Timing Error
WLE	=	Write Lock Error
IAE	=	Invalid Address Error
AOE	=	Address Overflow Error
PAR	=	Parity Error
RMR	=	Register Modification Refused
ILR	=	Illegal Register
ILF	=	Illegal Function
ATA7	=	Attention Active Drive 7
ATA6	=	Attention Active Drive 6
ATA5	=	Attention Active Drive 5
ATA4	=	Attention Active Drive 4
ATA3	=	Attention Active Drive 3
ATA2	=	Attention Active Drive 2
ATA1	=	Attention Active Drive 1
ATA0	=	Attention Active Drive 0

Error bits unique to the RH70:

APE	=	Address Parity Error
DPEOW	=	Data Parity Error, Odd Word
DPEEW	=	Data Parity Error, Even Word
WCEOW	=	Write Check Error, Odd Word
WCEEW	=	Write Check Error, Even Word

D.16 - RS03/RS04 Device Routine (Cont'd)

The unique error messages, which may occur in the use of this device routine, are as follows:

DATA ERROR (STMNT # nnnn)

Standard implementation - see section D.1.2.

DISK IS OFF-LINE

Immediately prior to initiating an I/O operation, the Medium On Line bit (MOL) in the RSDS register was found to be reset.

ERROR CN INITIATION

Prior to initiating an I/O operation, a condition that sets the SC bit in RSCS1, other than an Unsafe, did not clear after five attempts to reset it. The error bits that caused SC to be set will be listed.

EXHAUSTED RETRIES

Upon detecting one of the four retryable errors, the device routine has re-issued the failing I/O operation the number of times specified in RTRY without successful completeion of the operation. If RTRY is initially 0, this message will not occur.

INT WITHOUT ATA

This message indicates that an interrupt was received without the drive's ATA bit being set, when it was expected to be set. This error occurs where a SEARCH command is issued and an interrupt was received without ATA being set.

INV UNIT #

The current unit number, which is displayed in octal, is not in the range of 0-7. The ASSIGN command may be used to correct the erronzous unit number. Increments the data/operator error count.

I/O TERMINATION ERROR

Error bits have been detected in the device registers when processing a termination interrupt. This message occurs for all non-retryable errors and retryable errors when RTRY is set to 0.

NON-EXISTENT DRIVE

Immediately after loading a valid unit number (0-7) into RSCS2, the Non-Existent Drive bit (NED) in RSCS2 was found to be set.

NON-INT I/O TERMINATION ERROR

Error bits were found set in the device registers a few microseconds after issuing one of the Non-Interrupt type of I/O functions (DRESET or CRESET)

TIMEOUT ON I/O

After initiating an I/O operation, the terminating interrupt was not received. The time allowed for the interrupt to occur is approximately ten seconds when on the PDP-11/45 and with no other user programs executing.

T/O ON CRESET

This error, which indicates that Ready (RDY) did not set within a few milliseconds after setting the CLR bit in RSCS2, may occur when the CRESET statement is issued.

UNEXP ATA COND

When processing an interrupt after issuing an I/O function that should not cause an ATA condition, the disk's ATA bit was found set.

UNSAFE ERROR ON INITIATION

Prior to initiating an I/O operation, the Unsafe bit (UNS) in RSER was found to be set. If the previous I/O operation did not result in an error, this error is reported immediately. If there was an error on the previous operation, a drive clear will be issued to reset the condition. If UNS is still set after the drive clear, this error is reported.

D.16.4 RS03/RS04 SAMPLE PROGRAMS

The following are sample RS03/RS04 programs that perform the functions indicated in their descriptions:

1. The following program will verify all tracks on the entire disk by writing 1 bits to the data fields. Retries will be inhibited so as to isolate marginal tracks:

```
*ENTER 1 AS TRKCK
?ASGN DEV: RS04,0
?RDIO = 256 / 0
?WRIO = 256 / 16384
?DEV REG = 172040 /
?INT VEC = 000204 /
?BUS REQ = 5    /

ENTER STMNT'S

>0010   LOAD RTRY WITH 0
>0020   FILL D16384 AT WRIO WITH 177777
>0030   WRITE D16384 FROM WRIO
>0040   WRCK D16384 AT WRIO
>0050   STEPUP D64
>0060   IF 4 AT TRAK <> 0 GOTO 30
>0070   END
>DONE

*RUN 1
```

2. Use eight-sector transfers to write random data on the disk. Read each block of eight sectors after it is written, and software compare the data for accuracy.

```
*ENTER 3
?ASGN DEV: RS04,5
?RDIO = 256 / 2048
?WRIO = 256 / 2048

ENTER STMNT'S

>0010   FILL D2048 AT WRIO WITH RANDOM
>0020   WRITE D2048
>0030   READ D2048
>0040   VERIFY D2048 AT WRIO WITH RDIO
>0050   STEPUP D8
>0060   IF 4 AT TRAK <> 0 GO TO 10
>0070   END
>DONE

*RUN 3
```

APPENDIX E

ERROR MESSAGES

E.1 GENERAL INFORMATION

The error messages described in this appendix are independent of those issued by the individual device routines. For a description of the errors produced by a particular device routine, refer to that routine's section in Appendix D - Device Routines.

Upon detecting improper data entries, invalid sequences, or device errors, MPG will display messages on the console terminal that describe the nature of the error. These messages, which are always printed in upper case ASCII, are prefixed with the code *ER* so that they may be distinguished from normal console messages issued by MPG.

The error messages described in this appendix may be divided into four classes according to the MPG component that issues them. These four components are as follows:

- Stand Alone Executive
- MPG Command Processor and Control Routine
- MPG Memory Management Version
- MPG Language Processor and Compiler

The following sections contain a list of the error messages that can be issued by each component and a description of each error.

E.2 EXECUTIVE ERROR MESSAGES

The following is a list of the messages that can be issued by MPG's Stand Alone Executive. Each message is prefixed with the error I.D. code of *ER* and the name of the device, when it is a device type of error, in the format of "xxxx DEV:" where xxxx will be either SAVE, FTCH, LOAD, or LST.

Message	Description
CKSUM ERR	When reading a device routine or a saved user program from the LOAD or FETCH device, the checksum byte computed on the read data does not match the checksum byte contained in the data.

Error Messages (Cont'd)

DEL ERR

When deleting a user program, the MAP bit blocks were exhausted before all bits were processed. An illogical error; probably due to some type of device malfunction.

DEL OLD

MPG has been instructed to save a user program but the name which it is to be saved as already exists. The user must either re-enter the /SAVE command with a different name or delete the existing file.

DEV ERR

A device error (INOP, read error, write protected, etc.) has been detected on the indicated device.

DEV FULL

When attempting to save a user program, either all directory entries are in use or all data blocks are in use.

END OF FILE

Occurs when reading a file and a read for the next block is issued and there is not a next block. An illogical error which may be caused by the previous block being read incorrectly.

FILE NOT FND

This message, which is preceded by the nine character ASCII file name, indicates that the named file was not found on the LOAD or FETCH device.

HW ERR

Issued for the LIST device, this message indicates that an error condition (out of paper, offline, etc.) exists. When the condition is corrected, printing will resume automatically.

INSUF MEM FOR MPG

A minimum of 16K words is required for MPG. Unless additional memory can be brought on-line, you can not proceed any further. May also be caused by a bad read of the MPG program file from the load device. This error message will be followed by a halt. After taking corrective action, if possible, depress CONTINUE for a retry.

INV CMND FOR DEV

This error occurs when a directory type of command (/LIST, /DELETE, /ZERO) is issued to a non-directory type of device (paper tape).

INV DATA

The reply for the console terminal constants was either not in octal format or its value was larger than 377.

INV DATE

An incorrect reply was made for the date request. The format of the reply is DD-MMM-YY.

E.3 MPG COMMAND PROCESSOR AND CONTROL ROUTINE

The messages listed in this section occur during execution of user entered commands and/or user programs. Other than a couple of exceptions, all messages are prefixed with the error I.D. code of *ER*. For errors that occur on commands that may have multiple program numbers, the error message is prefixed with the program's number in the format of "PROG # nn". Also, for messages that report invalid data, the error message will be issued and then followed by another message that displays the data on which the error was detected.

Messages — Description

nnnnnn ADJ TO EVEN

An octal memory address (nnnnnn) for the RDM, WRM, or BOC commands was found to be odd. It is adjusted to the next lowest even address and then used as the address. The *ER* code is not included with this message.

ADR > MEM

An octal memory address, entered on the RDM, WRM, or BOC commands, is not within available memory or the range of device register addresses.

'AS' NOT SPECIFIED

The separator word AS was not specified in the /SAVE or /FETCH commands.

BUF/COM NOT SPECIFIED

The destination name of BUF or COM was not specified in the FILL command. The data that should be BUF/COM will be displayed.

DATA CONV ERROR

Data entered by the user, which was expected to be in either octal or decimal format, contained characters other than 0-7 or 0-9. The invalid data will be displayed.

DATA ERROR (STMNT # nnnn)

A data miscompare has been detected by the VERIFY statement in a user's program.

DATA NOT SPECIFIED

Neither pattern, octal nor ASCII data was specified on the FILL command.

DOESN'T EXIST

A command, which performs an operation on an existing program, is directed at a program slot which does not have a resident program.

INACTIVE

A STOP, CONT, or KILL command was directed at a program that had not been specified for execution.

IN CONTROL

The user has issued a RUN command, while at the user program interrupted command level, for a program that is currently running and was in control at the time of interrupt. Due to stack control, this an invalid operation. The STOP or KILL commands must be issued for the program, execution resumed, and then another user interrupt issued before the RUN command will be accepted for that program.

INSUF MEM FOR DEV ROUT & I/O AREAS

When in the process of entering or fetching a new program, MPG has determined that there is insufficient memory space available.

INV ADR RANGE

The octal memory address entered for the device register address is not equal to or greater than 160010 or, for the interrupt vector address, is not less than 1000. The invalid address will be displayed.

INV BUF/COM #

The value entered on the FILL command was not either 0-15 for the BF operand or 0-9 for the COM operand. The invalid number will be displayed.

INV CMND

When scanning the keyboard data for a command entry, the first word of data does not match any command name either in full or the first two characters. The invalid command entered will be displayed.

INV DATA = xxx...xx

The display of the invalid data. This message will always be preceded by a descriptive error message.

INV DEV TYPE

Either a device which does not have device routine support or the keywords KBOO, KYBD or LOAD have been entered as a program's device type. Or, a device type has been entered for SAVE, FETCH, or LIST and is not applicable for those functions.

INV LINE #

The program statement line number entered on the RUN command does not match any line number in the user's program. The invalid line number will be displayed.

INV MDL NAME

The device's model name entered is not supported by MPG. The name entered will be displayed.

INV NAME

The program name entered on the ENTER, /SAVE, /FETCH, or /DELETE commands either does not consist of 1 to 6 characters or the characters are not A-Z or 0-9.

INV NUMBER

The decimal number entered for the CDB command could not be converted to a one word binary number.

INV PAT #

The pattern number entered on the FILL command was not either PAT0 thru PAT9 or PATA.

INV PROG #

The program numbered entered is not in the range of decimal 1 thru 16.

INV STK ADR

The value of the stack pointer was greater than the maximum allowable value when a user program returned control to MPG. Execution of the user program will be terminated immediately.

INV UNIT #

A unit number entered while assigning a device is not within the range of decimal 0 thru 15.

LINE # NOT SPECIFIED

A program statement line number did not follow the separator word 'AT' in the RUN command.

Error Messages (Cont'd)

MDL NAME NOT SPECIFIED
A device's model name was not entered in reply the ASGN DEV message.

NAME NOT SPECIFIED
A one to six character name to be used as the save I.D. was not entered following the separator word 'AS' on the /SAVE command.

NO OBJ CODE
Either the RUN command or the DISPLAY command with the CODE option specified a program which does not have generated object code.

NO PROG'S FND
The OPSW command was issued for all programs (program number not supplied), but there were no user programs resident.

NO REPT AVAIL
The REPORT command has been issued to a program whose device routine does not support the report facility. The *ER* code is not included in this message.

ODD ADR
The memory address entered for either a device register address or interrupt vector address was odd.

OPERAND NOT SPECIFIED
An operand was not entered for a command that requires at least one operand (other than the command I.D.).

PROG # NOT SPECIFIED
A program number was not entered on either the /FETCH or the ASSIGN commands.

RESTRICTED CMND
While entering commands at the user program interrupt level, a command that would alter the contents of memory was entered. Refer to section 4.3 for more details.

2ND OPERAND NOT SPECIFIED
A command that requires two or more operands did not have a second operand.

STK TOO BIG
This error occurs when a user program returns control to MPG and has left more than thirty words on the stack. Since space is reserved for no more than 30 words, data would be lost for the program. Therefore, this is reported as an error condition and the program's execution is terminated.

UNNEC OPERAND — After processing a command, an unnecessary operand was found in the command statement.

VLD DEV TBL LD ERR, PRESS "CONT" TO RETRY
When attempting to read the Valid Devices Table file (TVDAnm.MPG), the handler has detected an error on the LOAD device. The type of error will be identified by a handler error message issued before this one. Attempt to correct the condition and press the "CONTINUE" switch to retry the read.

'WITH' NOT SPECIFIED — The WITH separator word was not specified in the FILL command.

E.4 MPG MEMORY MANAGEMENT VERSION ERRORS

The error messages listed in this section are generated by the Command Processor and Control Routine but are applicable only to the Memory Management version of MPG (TMMAnm.MPG). They can occur only in the sections of code that are unique to this version.

Messages — Description
-------- — -----------

ADR WITHIN MPG — The memory address entered for the program's area on the ENTER, /FETCH, or SHIFT commands is within MPG's program space.

DESTINATION NOT SPECIFIED
The destination for the SHIFT command (program number, memory address, or MPG) was not entered on the command.

INSUF REGS AVAIL — After loading a device routine, which utilizes the Unibus Map, on the ENTER or /FETCH commands, there were not a sufficient number of contiguous Unibus Map registers available to satisfy the program's needs. Either other programs will have to be deleted or the registers assigned to other programs will have to be reassigned with the UBMAP command to make enough contiguous registers available. This error aborts the ENTER or /FETCH command.

Error Messages (Cont'd)

INSUF ROOM FOR PROG

When using the SHIFT command to move a program to a new address, the program would not be able to fit in the space available. This space is from the supplied starting address to the next highest program in memory or to the end of memory.

INV REG # (SHD BE 7 - 28,29,30)

The register number entered on the reply to the Unibus Map register request message was not in the range of decimal 7 thru 30. Will also occur if two registers are required and 30 was entered or if three registers are required and 29 or 30 was entered.

PROG AREA IN USE

The memory address entered as a program's starting address on the ENTER, /FETCH, or SHIFT commands specified an area which is already occupied by another user program.

RDIO+WRIO > 24,250

The combined size specified for RDIO and WRIO exceed the decimal number of bytes indicated. A program's area, which includes RDIO and WRIO cannot exceed 24,576 bytes (12K words) in this version.

REGS IN USE

The registers specified on the reply to the Unibus Map register request message are already being used by other programs.

'TO' NOT SPECIFIED

The 'TO' separator word in the SHIFT command was not entered.

WARNING!! PROG I/O AREA > 18 BIT ADR - MAY NOT RUN

This message can occur only on a program that would normally use the Unibus Map. It is conditioned by 22 bit addressing being used and the Unibus Map not being used. After a program is Entered or Fetched or after it has been Shifted to a new address, the end of the WRIO area will be checked to determine if it is greater than octal 1000000. If it is, this message is issued. The program may still run if it does not do I/O operations or if the I/O operations do not exceed the octal 1000000 address boundary.

E.5 MPG LANGUAGE PROCESSOR AND COMPILER

The messages listed in this section would be encountered while entering MPG program instructions from the console or retreiving them from the MPG save file storage media. In addition to the specific messages listed below, which are preceded by *ER*, any user entry that cannot be interpreted will be echoed back, preceded by *ER* and surrounded by question marks.

SYNTAX ERROR ON SAVED PROG - END FORCED

An MPG program statement that was in a proper format when stored on the SAVE media, was not in the proper format when retrieved using a FETCH command. An END statement was substituted, and all preceeding statements were compiled.

LAST STMNT NOT AN END

The MPG statement with the highest sequence number is not an END statement. This message only appears after the user attempts to exit using a DONE statement. Since an automatic Compile takes place after the DONE is accepted, the user is not allowed to exit until an END statement is in the proper place.

nnnn NOT FOUND

The line number nnnn could not be located in response to an nnnn DELETE statement.

MEM LIMIT EXCEEDED - END FORCED

The storage requirement for the program being defined, including both the packed source statements and resultant object code, will exceed available memory if the current statement is included. An END statement is substituted, and all previous statements are compiled.

INV LINE #

The first four characters entered by the user are not valid as a line number (not one to four decimal digits).

CAN'T xxxx ON dddd

The MPG instruction xxxx is valid, but is not supported by the assigned Device Routine dddd. An example of this is a REWIND instruction when a Disk is assigned.

Error Messages (Cont'd)

NON-EXISTENT LINE REF ON LINE nnnn

The MPG instruction with sequence number nnnn referenced another MPG sequence number that didn't exist. This is the only error that cannot be detected when statements are entered, and only appears after a DONE statement, while a compile is in progress.

INV FMT ON SAVED FILE

When reading a user program file from the FETCH device, the end of the file has been reached and an END statement has not been processed.

APPENDIX F

USER PROGRAM SUBMITTAL FORM

Since MPG provides the ability to generate programs designed to test various areas of the hardware, it was felt that a mechanism was needed to distribute selected programs written by the MPG users. By using the following form, a user that writes a program, which he feels would be useful to other technical personnel in the factory and/or the field, may submit the program to the Diagnostic Group for review and inclusion in a manual that consists of a catalog of these user programs. This manual will be available through normal distribution and will be updated periodically.

You, as users of MPG, are by far the best source for realistic user programs. Due to your detailed knowledge of the hardware and the obvious fact that the program would not have been written unless there was a need for it, your programs have meaning for their existence. Therefore, why not share your pet programs with other users and thereby reduce duplication of effort. So, keep those cards and letters coming in folks.

Since this manual contains only one form, it is suggested that it be removed from the manual and be copied. This way, if you later have a second program to submit, you will still have the form master.

Please fill in all items on the form. If your program is lengthy, add extra pages as needed. Or, attach a listing of the statements produced by the DISPLAY command. When completed, mail the form through internal mail to:

DIAGNOSTIC ENGINEERING - MPG

ML 21-4 / E10

Page 1 of ...

MPG USER PROGRAM SUBMITTAL FORM

DIAGNOSTIC ENGINEERING - MPG ML 21-4 / E10

Device Tested:......... P.D.P.-.........System

Author's Name:......................... Date:..........

Office Location:............... Mail Code:.........

Brief descriptions of:

A) Area of the hardware tested:

B) Purpose of the program:

Program Name:................

Name of associated program, if any:...............

Device assigned to the program:............

RDIO Size:........ WRIO Size:........ OPSW Setting:........

Uses: COM......... FREE......

BUF/BF...... MIDL......

Other:..

Page ... of ...

Program's Source Statements:

(Include line numbers and use additional pages if needed.)

APPENDIX G

MPG CODING FORM GENERATION

Following this description is a short user program which can be used to quickly generate coding forms formatted for MPG. Its operation is very simple and only requires NONE as the assigned device. The first statement sets the OPSW bit that inhibits the printing of the 'PROG COMPLETED' message. The next two statements set up the Form Feed character and then issues it. The remaining lines initialize the decimal number 10, prints it in decimal, spaces a line, increments the number by 10, and then repeats the sequence. This sample will print line numbers 10 thru 300. By changing the constant in line # 80, any number of lines can be printed. This form will be directed to the MPG LIST device which is either the console terminal or optionally, the printer.

```
0010     SET OPSW BIT 0
0020     LOAD TM00 WITH 14
0030     PRINT 1 AT TM00 IN ASCII
0040     LOAD TM01 WITH D10
0050     PRINT TM01 IN DECIMAL
0060     PRINT *
0070     INCR TM01 BY D10
0080     IF TM01 <> D310 GOTO 0050
0090     END
```

APPENDIX H

MPG REVISION HISTORY

The first section of this Appendix contains a summary of the support provided on the first release of MPG. The remaining sections summarize the more visable differences between each subsequent release and the previous one. Where applicable, the user will be directed to more detailed information contained elsewhere in this manual.

H.1 INITIAL RELEASE OF MPG (VERSION # 1)

The first release of MPG became available from SDC in August of 1975. Being an initial release, it supported a basic complement of devices and features which are listed next.

Executives and Load Media

Four Executives, each of which supports a different primary load device, were supplied on this release. The load media of the different versions, each of which supports the PC11/PR11 Paper Tape as secondary SAVE/FETCH devices, are as follows:

TCMPG	TC11 DECtape
RKMPG	RK11 Disk
TMMPG	TM11 Magtape
THMPG	TM02 Magtape

Also contained in each of the Executives is the driver for the MPG LIST device. The different printers supported for this are the LP11, LS11, and LV11.

MPG Program

The MPG program consists basically of the Command Processor, Language Processor, Compiler, and Program Dispatcher. It requires a minimum of 16K words of memory and will support up to 28K. The commands listed in Appendix A and the user program statements in Appendix B are supported by this program.

Device Routines

Six full support device routines and twelve minimum support device routines are included on this release. The full support device routines provide high level I/O statements (READ, WRITE, SEEK, etc.) for the following devices:

DJ11	16 Line Async Multiplexer
DL11	Single Line Async Interface
DQ11	NPR Sync Interface
RK11	Disk
TC11	DECtape
TM11	Magtape

The minimum support device routines provide symbolic names for device registers and an alterable device register base address for the registers. The model names for the twelve routines are:

CD11
CR11/CM11
DC11
DH11
DN11
KW11
LP11/LS11/LV11
PC11/PR11
RC11
RF11
RP11
TA11

H.2 VERSION # 2 RELEASE

Version # 2 of MPG became available from SDC in April of 1976. The primary thrust of this update was toward Memory Management and Unibus Map support. However, additional load media and device routines were also included.

Before listing the changes for each section of MPG, it should be noted that one modification pertained to all MPG sections. This is the support for operation on an LSI-11. All areas of incompatibility were modified for compatibility across the PDP-11 series of processors.

Executives

Two new versions of the Executive, each of which supports a new load medium, have been added. The names of the Executives and the medium each supports are:

RXMPG	RX11 Floppy Disk
RBMPG	RP04 Disk

A bug, which occurred when an LV11 was assigned as the MPG LIST device, has been corrected in all versions of the Executive. Also, all versions will now determine if Memory Management is present and if it is, will ask if it is to be used.

MPG Program

With this release there are now two versions of the MPG program. The original version has been retained and has approximately the same capabilities. The second and newest version (DTMMA-A) includes support for Memory Management, Unibus Map, and 22 bit addressing. Several enhancements have been incorporated into the original version and have also been included in the Memory Management version. These changes are:

- The table of valid device names is no longer a part of the MPG program. It is now a separate file (TVDAAO.MPG) and will be loaded by MPG immediately after the typing of MPG's title.

- Corrected a bug with the IF instruction.

- Added the WORD instruction. This allows the user to specify binary code in his program. Refer to Appendix B.3.

- When specifying a program name, the user may now use from one to six characters. Previously, it was required to be six characters.

- The FAST option has been added to the /LIST command. This results in only the filenames being listed.

- The FILL command will now accept the symbolic names PATO thru PATA in its operation. This provides for filling the buffers with the predefined bit patterns.

- Another predefined bit pattern has been added. Its name is PATA and will generate count words. Refer to Appendix C.3.

- Immediately after loading a device routine for a user program, MPG will display its filename in full on the console terminal.

- A test is made for the Cache Memory's Hit/Miss register and if found, the timing constants for the DELAY instruction and the I/O timeout will be adjusted.

- Bus addresses generated for all NPR I/O instructions and the READ and WRITE instructions will be two words in length.

The Memory Management version of MPG has some additional capabilities and some restrictions. For more details, refer to Section 9 which has detailed information about this version. A brief summary of these additional features follows:

- Two additional commands are supported. The SHIFT command allows the user to move his programs around in memory and to change their program slots. The UBMAP command lets the user change the Unibus Map registers assigned to his program.

- An existing command has been expanded. The MM (Memory Map) command will now accept program numbers. When this format is used, additional detailed information about the specified programs is listed.

- The RDM, WRM, BOC, ADD, and SUB commands will accept up to 22 bits for their addresses and/or operands.

- The user can now optionally specify the starting memory address for each of his programs.

- The RESET and VECTOR instructions are not supported in this version.

- A common trap catcher has been added. This routine will display pertinent information when traps occur at vectors 4, 10, 34, 114, and 250.

Device Routines

Four full support device routines and two minimum support device routines have been added on this release. Also, all minimum support device routines have been merged into one file which tailors itself to the specified device when loaded.

The four new full support device routines are for the following devices:

DH11 16 Line Programmable Async Multiplexer
LP11/LS11/LV11 Printers
PC11/PR11 Paper Tape Readers and Punch
RP04/RP05/RP06 Disks

The two new minimum support devices are:

RX11/RX01 Floppy Disk
TM02/TU16 Magnetic Tape

The original six device routines were modified for compatibility with the Memory Management version of MPG. Other changes that were included are:

- An I/O timeout error is now treated the same as other device errors. The user may now issue the "CONT p" command to resume execution.

- A bug with the BREAK instruction for communication devices was fixed.

- All device routines will display the unit number in octal when reporting invalid unit number errors.

- The RK11 device routine will now validate the device I.D. bits in RKDS on search complete interrupts.

- The DJ11 and DL11 device routines now keep counts for the number of Read and Write interrupts.

H.3 VERSION # 3 RELEASE

Version # 3 of MPG became available from SDC in August of 1976. This release consisted of the addition of four new device routines and the fixing of three bugs in the MPG programs. Also, a new Pattern format has been added.

MPG Programs

The three bugs corrected applied to both versions of the MPG program and are:

- The binary code generated by the compiler for relative "IF" instructions (IF >, <, =>, =<), contained signed branch instructions. These have been changed to unsigned branch instructions.
- The "BREAK v" and "BREAK v ON v" instructions could not be compiled.
- Octal addresses entered for I/O instructions requiring two word addresses (READ, WRITE, etc.), were not compiled properly.

A new Pattern format (PATB) has been added and will generate the following two word pattern:

```
165555
133333
165555
133333
 etc.
```

The Valid Devices Table file (TVDAB0.MPG), which is loaded by the MPG programs, has been revised to include the model names supported by the four new Device Routines.

Device Routines

Four new full support Device Routines have been added on this release. Their filenames and the devices that each supports, are as follows:

TDUAA0.MPG	=	DU11 Synchronous Line Interface
TRSAA0.MPG	=	RS03/RS04 Disks
TR3AA0.MPG	=	RP02/RP03 Disks
TR6AA0.MPG	=	RK06 Disk

(End of M.P.G. Users' Manual)

```
                                   J03
Spooler runtime 20 Seconds, 84 KCS, 793 disk reads, 3 disk writes, 241 pages
[redacted]
0[redacted]000000000000000000000000000000000000001111111111111111111111111111111110
0[redacted]00000000011111111112222222222223331 2
[redacted]567890123456789012 #
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111110
9000000011111111112222222222333333333344444444445555555555666666666677777777778888888888999999999900000000001111111111222222222233312
```