the digital group
po box 6528 denver, colorado 80206 (303) 777-7133

DISKMON
Operating System
Version 1.0

Written by
David Bryant

1st Edition — April 1978

DISKMON OPERATING SYSTEM INTRODUCTION

The DISKMON Operating System is designed for the Digital Group Z80 microcomputer with one or more disk drives and a disk controller. DISKMON provides a set of executive routines which allow user programs easy access to files and file creation.

Hardware Configuration

The DISKMON Operating System is designed to operate in an 18K or larger Digital Group Z80 system. Minimum hardware includes:

1) Digital Group Z80 CPU card
2) Digital Group input/output card
3) At least 16K of memory
4) Digital Group 16 X 32 or 64 TV readout/cassette interface card
5) Digital Group disk controller card
6) 1 to 4 Digital Group disk drives
7) Digital Group keyboard
8) Digital Group motherboard and power supply
9) Video monitor
10) Optionally 1 to 4 Phideck drives with Phideck controller.

The system will self-configure to utilize all available memory.

Using the DISKMON System Manual

The DISKMON documentation provides a complete user's guide for the DISKMON Operating System. The manual is divided into three parts. Part One contains detailed instructions for getting on line with a new DISKMON system. Part Two describes how to use the DISKMON keyboard monitor commands. Part Three describes how the user can write programs that can access and create files using the DISKMON executive routines, and implement user-made patches.

You, the DISKMON Operating System user, can help make the system easy to use, flexible, and extensive, by critiquing the manual and offering suggestions on how it can be improved. Please report any bugs found in the system and supply as much information as possible as to how they came about. This information should be submitted to:

Software Development/DISKMON
Digital Group, Inc.
P. O. Box 6528
Denver, CO. 80206

A form for submitting DISKMON corrections and suggestions is included at the back of this manual.

PART 1
DISKMON INSTALLATION

The DISKMON Operating System is supplied as a package containing:

1) The DISKMON Operating System manual;
2) A DISKMON System Disk;
3) A DISKMON Z80 Bootstrap PROM.

Memory Allocation Requirements

DISKMON requires that the 2K of memory on the CPU module be jumpered for memory locations 340000 through 347377 (octal). See the Z80 CPU card documentation for jumpering information.

Main memory should be jumpered to start at location 0 and build upward.

Display & Keyboard Port Assignments

The TV Readout is connected to output port 0 and the keyboard is connected to input port 0. The audio cassette input and output are connected to bit 0 of input and output port 1. These are the standard peripheral device assignments as described in the system writeups.

Disk Port Assignments

The DISKMON Operating System assumes that the disk controller will be connected to the system as shown in the disk manual. Disks are referred to as units 0 through 3 in DISKMON . (This corresponds to numbers 1 through 4 in the disk drive documentation.)

Starting the System

1) Make sure that the CPU board memory is jumpered correctly.
2) Make sure the Z80 DISKMON Bootstrap PROM is installed.
3) Turn power on.
4) Place the DISKMON disk in disk drive zero and close the door.
5) Push the RESET switch.
6) The system will automatically bootstrap and display the DISKMON prompt (or execute the BOOT command if one has been specified).

Pressing the RESET button will cause DISKMON to restart itself if resident, or reload itself if not.

It is recommended that the distributed system disk be used only for a master copy. Additional working copies of the system should be built from this master system (see the FORMAT, ZERO, and BUILD commands for details).

The following sequence of commands may be used on a single disk drive system:

```
FORMAT
   (switch disks before responding to the "INSERT
   DISK..." message.)
SZERO!
```

The following sequence of commands may be used on a multiple disk drive system:

```
FORMAT#1
ZERO!#1
```

PART 2
DISKMON MONITOR

The DISKMON keyboard monitor provides communications between the user and the DISKMON overlays by accepting commands from the terminal keyboard. The user can run system and user programs, save programs on disks or cassettes, zero and build system and non-system disks, delete and rename files, make modifications to the system, update disks to include these modifications, and read and write audio format cassette tapes.

Device Names

Each mass storage device in DISKMON is designated with an octal digit 0 through 7. Device numbers 0 through 3 refer to the four disk drives supported with device 0 being the system device. Devices 4 through 7 represent Phideck cassette drives.

DISKMON makes use of the terms "word", "byte", "page", "record", and "block" as units of storage. In directory listings, file lengths are referenced in terms of decimal blocks. The terms are defined as follows:

1 word = 2 bytes (16 bits)

1 block = 1 record = 1 page = 256 bytes

Each byte consists of 8 bits. A full page and byte address consists of 16 bits.

File Names and Extensions

Files are referenced symbolically by a name of up to six alphanumeric characters, starting with an alphabetic character, followed by a period and an extension of two alphanumeric characters. The extension to a file name is used by both system programs and the user to specify and determine the format of the file. System programs will append the correct extension in most cases. For example, saved programs automatically get the extension .GO (meaning memory run file). LOAD and RUN automatically look for .GO files.

A list of extensions currently in use is shown in the table below.

TABLE 2-1 DISKMON FILE NAME EXTENSIONS

Extension	Meaning
.AL	Assembly Language Assembled Source Listings
.AS	Assembly Language Source Files
.BA	Basic Program Listings
.GO	Memory Run Files
.OB	Object Files
.TX	Word Processing Files

System and Non-System Disks and Cassettes

A DISKMON system disk contains a directory, the DISKMON bootstrap code, a set of DISKMON overlays, and the file storage area. A system disk must be inserted in disk unit #0.

A DISKMON non-system disk or cassette contains only a directory and the file storage area. This makes more room for files. Either a system or a non-system disk can be used in disk units 1, 2, or 3. A PHIMON system or non-system tape cassette can be used with DISKMON as file storage cassettes.

The DISKMON commands ZERO and BUILD allow the user to designate disks as either system or non-system. Refer to the writeup on those commands for more details. DISKMON zeros tape cassettes only as non-system cassettes.

Entering DISKMON Command Strings

When the system is waiting for a command, it prompts with a " " at the left of the screen. A cursor shows where the next character will be accepted and indicates that the system is waiting for input. Pressing the 'ESC' key will cause the display screen to be cleared.

Only the required letters of the command need be entered but as many optional letters as desired may be typed. Typing 'RUBOUT' or 'DELETE' will delete the last character typed and 'CTRL/U' will echo "$" and delete the entire line.

If a disk or Phideck unit number is to be specified, the #N must follow the last command word letter typed. If the #N is omitted, the system defaults to disk unit 0, the system disk.

Only 62 characters are allowed in a command string, with additional typed characters being ignored. A command string is terminated with the 'RETURN' key which is represented with a '(CR)' in the command string format examples.

DISKMON Keyboard Monitor Commands

Each of the DISKMON commands are described below. The required letters that must be entered for each command are shown in bold-face type within the format descriptions for the commands. All command examples are shown in the octal entry mode. A similar form would be used in the hex input mode. See the commands OCTAL and HEX.

ALTER

The ALTER command allows the user to fetch DISKMON System overlays into memory from the system overlay area on the system disk. The format of the ALTER command is:

ALTER#N (CR)

The number 'N' represents the octal or hex overlay number that is to be loaded into memory. Control returns to the DISKMON keyboard monitor when the load is complete. The overlay can now be patched for errors or major modifications can be made using DTO or DTH. Overlays are loaded into memory on Page 1 and 2. The overlay can then be written back onto the system disk using the INSERT command. Refer to Part 3 of the DISKMON manual for additional information about changing overlays.

If the user types 0 or a number greater than 30, the system will display the error message "WHAT?"

EXAMPLE:

ALTER#12 (CR)
Overlay #12 will be loaded into page 1 and 2 in memory and control will return to the DISKMON monitor.

BUILD

The BUILD command writes the system area from the system disk (Drive 0) onto a zeroed (! option) disk in one of the other drives. The format of the BUILD command is:

BUILD#N (CR)

where 'N' represents disk drives 1, 2, or 3. The bootstrap code and all of the system overlays will be copied from the system disk to the specified disk. The disk to be built must have been zeroed with the "!" option. (See the ZERO command for more information.)

If '#N' is omitted, or a 0 is typed for 'N', an update of only the boot code, not the overlays, will occur. This will allow the user to install patches in the resident code if necessary and incorporate them into the bootstrap block.

If a disk to be built was not zeroed with the "!" option, the error message "CAN'T BUILD NON-SYS" will be displayed.

The **SBU**ILD form of the BUILD command is the same as the regular form except that DISKMON will display the message "SWITCH DISKS (SPACE)" before it does the building. This is for use in a single drive system. At this point, the user inserts the disk to be built into the system drive and types a space. When the building is complete, the message "SWITCH DISKS (SPACE)" reappears. At this point, the user re-inserts his system disk and types a space to enter the keyboard monitor.

EXAMPLES:

BU#3 (CR) — Write the entire system area from Disk 0 to Unit 3.

BUILD (CR) — Update the system on the system disk.

SBUILD — Update the system on the system disk after allowing the user to switch disks.

BOOT

The BOOT command allows the user to specify any legal DISKMON command to execute upon power-up. The user may, for example, have DISKMON power-up into a device directory or to run a user or system program. The format of the BOOT command is:

BOOT command-string (CR)

where command-string is any valid DISKMON command-string. After this, DISKMON will execute the specified command upon power-up. Typing 'BOOT' with no command deletes the BOOT command and causes power-up to go directly to the DISKMON keyboard monitor. See also PRBOOT and DOBOOT.

EXAMPLES:

BOOT RUN BASIC (CR)
Causes DISKMON to run the program BASIC upon power-up.

BO (CR) — Deletes any current Boot command.

(Warning: BOOT DOBOOT (CR) will cause a self-destruct upon the next power-up or execution of the DOBOOT command.)

COPY

The COPY command is used to copy files from one device (disk or cassette) to another. There are three copy command formats. The first is:

COPY#D<S FILE1.EX,FILE2.EX,FILE3.EX...(CR)

where D is the destination device, S is the source device, and the specified files are separated by commas with a space preceding the first file. The specified files are copied from device S to D and control returns to the keyboard monitor. If more files are to be specified than can fit on a single line, the last character on the line should be a comma. DISKMON will then prompt with an "*" for more file specifications. This may continue until all files are specified.

The second form of the COPY command is the '!' form and has the format:

COPY!#D<S (CR)

where D is the destination device and S is the source device. This form copies *all* files from the source device to the destination device.

The last form of the COPY command is the '?' form and has the format:

COPY?#D<S (CR)

where D is the destination device and S is the source device. This form of the command queries the user as to whether each file on the source device should be copied to the destination device. The file name will be displayed followed by a '?'. A 'Y' reply will cause that file to be copied; an 'N' reply will cause DISKMON to ignore that file when copying.

In all copy forms, omitting either the destination or source device will cause that device to default to zero. While copying files, DISKMON displays the name of each file copied. If any errors occur, these will be displayed next to the filename.

EXAMPLES:

COPY#4<1 FILE.EX,FILE2.EX, FILE3.EX,(CR)
* FILE4.EX(CR) Copies specified files from disk unit 1 to cassette unit 4.

COPY!#1<3 (CR)
Copies all files from disk unit 3 to disk unit 1

COPY?#2<5 (CR)
TEXT.EX?Y Copies the files TEST.EX and
PROG.AS?N PROG.GO from cassette unit 5 onto
PROG.GO?Y disk unit 2. PROG.AS is not copied.

COP!#1 (CR) Copies all files from the system disk, unit 0, onto disk unit 1.

COPY!<4 (CR)
Copies all files from cassette unit 4 onto the system disk, unit 0.

DELETE

The DELETE command allows the user to delete files from any disk or cassette drive. The format of the DELETE command is:

DELETE#N FIL1.EX,FIL2.EX,FIL3.EX...(CR)

The digit 'N' represents one of the disk or cassette drives (0 thru 7) and defaults to unit 0 if not specified. As many files as will fit on the command line may be deleted with commas separating file names. A space must precede the first file name.

Only 62 characters are allowed in the command line with additional ones ignored. If the file name cannot be found in the directory of the specified device, the error message "NAME.EX NOT FOUND" will be displayed.

EXAMPLES:

DEL#2 TEST.GO, TEST.AS, TEST.HL (CR)
Deletes the three files specified from disk drive #2 if found.

DE TEST1.GO (CR)
Deletes the file TEST1.GO from the system disk if found.

DIRECTORY

The DIRECTORY command produces a video display of DISKMON device directories. The format of the DIRECTORY command is:

DIR#N .EX (CR)

The digit 'N' represents one of the disk or cassette drives (0 thru 7) and defaults to unit 0 if not specified. The '.EX' is optional, but, if included, will cause only files with the two letter extension represented by '.EX' to be displayed.

A video display of all the files on the specified device and their size in decimal blocks will appear on the screen. If more than a screenful of files are on the device, typing a space will display the next screenful and so forth. After all the files have been listed out, DISKMON will display the number of empty blocks on the device and return to the keyboard monitor.

While the directory is on the screen, typing a non-space character causes command mode re-entry with the DISKMON prompt displayed.

The user may be interested in displaying only certain types of files stored on the device. Typing a period followed by a two-letter extension will produce a directory display with only those files with the specified extension. For example, the user might want to list only BASIC files stored on the system device. The command string would be: DIR .BA (CR). If the user wants to list all files that can be RUN (i.e., .GO files) on disk unit 2, the command string would be: DIR#2 .GO (CR). See Table 2.1 for a list of standard DISKMON extensions.

The user may be interested in the distribution of empty blocks on the device. Typing DIR!#N .EX (CR) (exclamation point option) will cause DISKMON to include in the display the empty areas on the device and their size in decimal blocks.

EXAMPLES:

D (CR)	Lists all files on the system device, unit 0.
DIR#4 (CR)	Lists all files on Phideck unit 4.
DIRECT#1 .HL (CR)	Lists all files on unit 1 with a .HL extension.
D!(CR)	Lists the files and the empty blocks on disk unit 0.

DOBOOT

The DOBOOT command causes DISKMON to execute the current BOOT command. This can be done to make sure that a just-specified BOOT command does not have any errors. The format of the DOBOOT command is:

DOBOOT (CR)

DTO & DTH

DTX, where X stands for either 'O' meaning octal or 'H' meaning hex, allows the programmer to run his program on the computer, control its execution, and make alterations to the program by typing instructions at the keyboard. The format of the DTX command is:

DTO (CR) or **DTH** (CR)

DTX Features

DTX features include location examination and modification and instruction breakpoints to return control to DTX.

The breakpoint is one of DTX's most useful features. When debugging a program, it is often desirable to allow the program to run normally up to a predetermined point, at which point the programmer may examine and possibly modify the contents of the registers, or various instructions or storage locations within his program, depending on the results he finds. To accomplish this, DTX acts as a monitor to the user program.

The user decides how far he wishes the program to run and DTX inserts an instruction in the user's program which, when encountered, causes control to transfer back to DTX. DTX immediately preserves in the stack the contents of all the registers and flags. It then displays the location at which the breakpoint occurred, as well as a display of the contents of all the registers at that point. DTX will then allow examination and modification of any location in the user's program and modification of any register. The user may also move the breakpoint, and request that DTX continue running his program. This will cause DTX to restore the registers and flags, execute the restored instruction, and continue in the user's program until the new breakpoint is encountered or the program is terminated normally.

Calling and Using DTX

DTX is called by typing: DTO (CR) or DTH (CR) in response to the DISKMON keyboard monitor prompt character. Before DTX is called, the user should have an executable version of the program in memory. The ALTER, LOAD, GET, or READ commands can be used to place the program in memory or a system program such as an assembler or a compiler may load the program. The user can return to the DISKMON keyboard monitor by typing 'G' (causes a jump to location 0 and thus monitor return) or by pressing the 'ESCAPE' key.

If the user is typing any amount of program additions directly into memory, the memory control block of the program loaded may not reflect the true extent of the program. New memory limits may need to be included when using the SAVE command.

Typing in Data or Addresses

When typing in data, leading zeros are not required, and only the last digits are considered (e.g. if 234243232 is typed for an 8-bit octal entry, only 232 is used, and for a 16-bit entry, only 243232 is used. When in DTH, only the last two or four hex digits are used.) The examples are shown assuming octal entry using DTO. Similar hex entries are allowed when DTH is being used.

Commands may be entered in either upper or lower case. Parentheses in the command examples are used for clarity only and are *not* typed.

While in DTH, commands A, B, C, D, & E must be entered as control characters by holding down the control key while typing the letter command. This is not necessary with DTO since octal numbering does not use these letters as digits.

DTX COMMANDS

NOTE: All examples show octal entry to DTO. A similar hex convention is used when running DTH. When using DTH remember to press the control key when typing the commands A, B, C, D, & E.

DTX Address Commands

The first set of DTX commands affect the address in memory currently being examined or modified.

(NNNNNN)O	OPEN LOCATION NNNNNN FOR EXAMINATION OR MODIFICATION

Typing an octal address followed by the letter O causes DTO to produce a memory display showing the contents of 8 memory locations preceding the address and the contents of the specified memory location just opened. The open location can then be modified by the following ALTER commands. Any octal number from 1 to 6 digits in length is legal input. If more than 6 digits are entered, only the last 6 are used by DTO.

(SPACE)	STEP FORWARD ONE LOCATION THROUGH MEMORY

Typing a space produces an adjusted memory contents display with the next location the open location that can be examined or modified.

(—)	STEP BACKWARD ONE LOCATION IN MEMORY

Typing a '—' produces an adjusted memory contents display with the previous memory location the open location that can be examined or modified.

(NNN)L	RESET CURRENT LOCATION TO 'NNN' ON CURRENT PAGE

Typing an octal number of 1 to 3 digits followed by an 'L' will cause the memory contents display to change so that location 'NNN' is now the current open location on the current page.

(NNN)S	SEARCH FOR NNN STARTING AT CURRENT LOCATION

Typing an octal number of 1 to 3 digits followed by an 'S' will cause DTO to search memory for the specified 8-bit byte starting at the current location. If found, the memory display is adjusted with that byte the current location. If the byte is not found, the command has no effect. Typing only an 'S' will use the last byte specified.

(NNNNNN)W	SEARCH FOR NNNNNN STARTING AT CURRENT LOCATION

Typing an octal number of 1 to 6 digits followed by a 'W' will cause DTO to search memory for the specified 16-bit word starting at the current location. If found, the memory display is adjusted with that word in the current location. If the word is not found, the command has no effect. Typing only a 'W' will use the last word specified.

DTX Memory Alter Instructions

The following instructions are used to alter the contents of the current open location. The next location in memory is then opened and the memory content display is updated.

(NNN)(SP)	DEPOSIT 'NNN' INTO CURRENT MEMORY LOCATION

Typing an octal number with 1 to 3 digits followed by a 'SPACE' causes that value to be deposited in the current location and the memory display to be updated with the next location now open. Only the last three digits are used.

A(CHAR)	DEPOSIT THE ASCII CODE IN THE OPEN LOCATION

Typing 'A' followed by any character causes the ASCII code for that character to be deposited in the open location and the next memory location to be opened with an updated memory content display. For example, typing 'A' followed by 'C' will deposit 303 in the open location and update the display with the next location now open.

Z	DEPOSIT A ZERO (000) IN THE CURRENT LOCATION

Typing 'Z' deposits zero (000) in the current location and updates the display with the next location now open. Typing Z's in rapid succession allows a section of memory to be zeroed quickly.

DTX Register Change Commands

The following commands allow changing the contents of the CPU registers.

(NNN)D(R) DEPOSIT (NNN) INTO REGISTER 'R'

Typing an octal number with 1 to 3 digits, the letter 'D' followed by a letter corresponding to a register (A, B, C, D, E, H, L) causes the number to be loaded into the specified register. For register loads, only the last three digits are used. To change the Z80 index pointers, type (NNNNNN)D followed by either 'X' or 'Y'. Typing '(NNN)DF' will set the flag configuration to (NNN).

E EXCHANGE REGISTER DISPLAY AND MODIFICATION MODE TO ALTERNATE SET

Typing an 'E' exchanges the register display and entry mode to the other set. The message "MAIN" or "ALTERNATE" displayed on the screen shows which set is active. The user must ensure that the proper set is active when continuing program execution.

DTX Program Execution Commands

The following commands are used to create breakpoints, to start program execution at specified points, to examine registers after a breakpoint has been encountered, or to continue from a breakpoint.

(NNNNNN)B SET UP A BREAKPOINT AT LOCATION (NNNNNN)

Typing an octal number with 1 to 6 digits followed by a 'B' causes a RESTART 60 instruction (367) to be deposited at location (NNNNNN) and the display of a breakpoint message at the top of the screen saying: "BREAKPOINT: NNNNNN". The previous breakpoint is removed at this time by depositing the original contents back into that location. A breakpoint must be located on an 'INSTRUCTION', not an immediate byte or an address or indeterminate results will be obtained.

B REMOVE EXISTING BREAKPOINT

Typing a 'B' without a preceding number causes the breakpoint to be removed by depositing the original contents back into the breakpoint location. The breakpoint message displayed at the top of the screen changes to: "BREAKPOINT: NONE".

(NNNNNN)G JUMP TO LOCATION (NNNNNN)

Typing an octal address with 1 to 6 digits and a 'G' will cause a jump to the user program at the address specified. Control may be lost if a breakpoint is not encountered. Pressing the 'RESET' button will restart the DISKMON keyboard monitor. Typing 'G' without any preceding digits will cause a jump to location zero and thus return to the DISKMON keyboard monitor. If a breakpoint is encountered, DTX is re-entered with the current register contents display on the screen. The top of the screen displays the message "RETURN: NNNNNN" where 'NNNNNN' is the address of the breakpoint encountered.

C JUMP TO LOCATION OF LAST ENCOUNTERED BREAKPOINT

Typing a 'C' causes a jump to the last encountered breakpoint with all registers restored or updated as modified and continues program execution. The previous breakpoint must be removed or replaced by a new one.

R ENTER REGISTER DISPLAY MODE

Typing 'R' will redisplay the register content display if a breakpoint has been encountered. It will do nothing if a breakpoint has not been encountered.

M ENTER MEMORY DISPLAY MODE

Typing 'M' will cause DTX to return to the memory display mode with the last open location as the current location.

(ESCAPE) DISMISS DTX

Returns to the DISKMON keyboard monitor.

FORMAT

The FORMAT command is used to put a standard IBM soft sectoring format onto a disk. The format of this command is:

FORMAT#N

where N is the disk number (1-3) to be formatted. When the formatting is complete, DISKMON returns to the keyboard monitor.

To format a disk in a single drive system, type the FORMAT command without a unit number. The message "INSERT DISK TO BE FORMATTED (SPACE)" will display. After inserting the disk to be formatted into the system drive, press the SPACE key to begin formatting. When the formatting is complete, DISKMON will display the message "INSERT SYSTEM DISK (SPACE)". After re-inserting the system disk, press the SPACE key to enter the keyboard monitor.

EXAMPLES:

FORMAT#3 (CR)
The disk in disk drive number 3 is formatted.

FOR (CR) A disk is formatted in a single disk system.

INSERT DISK TO BE FORMATTED (SPACE)
INSERT SYSTEM DISK (SPACE)

GET

The GET command allows the user to bring a specified file into memory at a specified address. This is usually used to reload files created with the PUT command. The file names are assumed to have an .OB extension unless otherwise specified. The format of the GET command is:

GET#N NAME NNNNNN (CR)

where NAME is the name of the file (including an optional extension if desired) and NNNNNN is the octal or hex address where the file is to load.

EXAMPLES:

GET#2 PROG 1000
Bring the file on disk unit #2 named PROG.OB into memory at address 1000.

GE TEST.BQ 30000
Bring the file TEST.BQ on the system disk into memory at address 30000.

HEX

The HEX command is used to put DISKMON in the hex mode of command input.

Executing the command DTH will also make this hex input mode change. The format of the HEX command is:

HEX (CR)

This change will only stay in effect until changed or until the system is reloaded. To make the change a permanent part of the bootstrap code, use the BUILD command.

INSERT

The INSERT command allows the user to rewrite updated system overlays back onto the system overlay area of the disk. The format of the INSERT command is:

INSERT#N (CR)

The octal or hex number 'N' represents the overlay number that is to be written back on the system disk. The first byte of page one and the last byte on page two will be checked to see if they are equal to the overlay number thus indicating that the overlay is probably in memory. If the values do match, the overlay will be written onto the disk in the appropriate spot and control will return to the DISKMON keyboard monitor. If the values do not match, the message "IMPROPER OVERLAY" will be displayed and control will return to the DISKMON keyboard monitor. If 0 or a number greater than 30 is typed, the error message "WHAT?" will be displayed.

EXAMPLE:

INSERT#12 (CR)
The contents of Location 0 on page 1 and 377 on page 2 will be checked to see if they are a 12. If so, the page will be written onto the overlay 12 area on the system disk. If not, the "IMPROPER OVERLAY" message is displayed and control returned to the DISKMON keyboard monitor.

LOAD

The LOAD command loads a memory run file (.GO format, not ASCII or any other format) into memory from a DISKMON device. The format of the LOAD command is:

LOAD#N NAME (CR)

The digit N represents one of the disk or cassette drives (0 thru 7) and defaults to unit 0 if not specified. 'NAME' represents the name of the file to be loaded and a .GO file extension is automatically added. The file is loaded into memory with its memory control block which contains information about the file such as its starting address and the areas of memory occupied by the file. After loading the specified file, control returns to the keyboard monitor.

If the file specified cannot be found, the error message "NAME.GO NOT FOUND" will be displayed. If no name is typed, the error message "WHAT?" will display. If the file is not a proper ".GO" file the message "NAME.GO IS NOT AN IMAGE FILE" will display.

The LOAD command is typically used before a debugging or patching session with DTO or DTH. LOAD is used to load the object program into memory, then DTO or DTH is called, and the program can be altered and/or debugged.

See the section on DTO and DTH for more details.

EXAMPLES:

LOAD#2 TEST (CR)
: The FILE TEST.GO if found will be loaded from unit 2.

LO TEST.SP (CR)
: The FILE TEST.GO if found will be loaded from the system device, unit 0. Note that the .SP extension was ignored.

MOVE

The MOVE command moves a specified block of memory from one location to another. The format of the MOVE command is:

MOVE XXXXXX,YYYYYY,ZZZZZZ (CR)

where XXXXXX is the octal or hex starting address of the block of memory to be moved, YYYYYY is the address that the data is to be moved to, and ZZZZZZ is the number of bytes to be moved.

EXAMPLE:

MOVE 1000,40000,1000
: Move the page of memory starting at location 1000 to location 40000.

OCTAL

The OCTAL command is used to put DISKMON in the octal mode of command input.

Executing the command DTO will also make this octal input mode change. The format of the OCTAL command is:

OCTAL (CR)

This change will only stay in effect until changed or until the system is reloaded. To make the change a permanent part of the bootstrap code, use the BUILD command.

PRBOOT

The PRBOOT command causes DISKMON to display the current BOOT command on the video display. The format of the PRBOOT command is:

PRBOOT (CR)

PUT

The PUT command allows the user to save specified areas of memory on disk or tape to be used later by other programs or by the GET command. The files created by the PUT command have an .OB extension (unless otherwise specified) and are direct memory images of the specified memory area. The format of the PUT command is:

PUT#N NAME NNNNNN, PPP (CR)

where NAME is the name of the file to be created (with an extension if desired), NNNNNN is an octal or hex full page and byte address of the area to be saved, and PPP is the number of octal or hex pages to be saved.

EXAMPLES:

PUT#2 PROG 72107,12 (CR)
: Creates a file on disk unit 2 named PROG.OB with the 12 pages of memory starting at 72107.

PU TEST.BQ 1000,3 (CR)
: Creates a file on the system disk named TEST.BQ with the 3 pages of memory starting at 1000.

READ

The READ command allows the DISKMON user to read a standard Digital Group audio cassette tape in the Suding 1100 baud format. The READ command has three formats. The first is:

READ (CR)

This will read a standard audio tape starting at location 1000. Start the tape on leader before pressing the RETURN key.

The second format for the READ command is:

READ SA (CR)

where SA represents a full page and byte address. The audio cassette tape will load starting at location SA and read until the current block of code is read in. Control will then return to the DISKMON keyboard monitor.

The third format for the READ command is:

READ SA-EA (CR)

where 'SA' and 'EA' are full page and byte addresses. The cassette will be read into memory starting at location 'SA' and will end at location 'EA' unless the current block runs out (trailer encountered) before reaching the ending address. Control then returns to the DISKMON keyboard monitor.

If noise occurs while reading leader, the keyboard monitor will be restarted. Retype the command to re-enter the READ routine. The least significant digit of the page being loaded is displayed on the screen as usual. If the memory location does not load properly, a "." will be displayed.

If address limits between 340000 and 347377 are typed, the tape will destroy the DISKMON resident code.

EXAMPLES:

READ (CR) Standard read as if done from Z-80 op system.

READ 20010 (CR)
Start loading at location 20010 and continue loading until the current block is done.

READ 20000-20100 (CR)
Start loading at location 20000 and stop loading at 20100 unless the block runs out first.

RENAME

The RENAME command allows the user to rename a file on any disk or cassette drive. The format of the RENAME command is:

RENAME#N OLDNAM.EX, NEWNAM.EX (CR)

The digit 'N' represents one of the disk or cassette drives (0 thru 7) and defaults to 0 if not specified.

If the file OLDNAM.EX is not found, the message "OLDNAM.EX NOT FOUND" will be displayed. If the file NEWNAM.EX is already in the directory, the message "NEWNAM.EX ALREADY EXISTS" will be displayed.

EXAMPLES:

RENAME#1 FROWN.GO,SMILE.GO (CR)
The file FROWN.GO on drive 1 will be renamed to SMILE.GO.

REN FUM.BA,FOO.BA (CR)
The file FUM.BA on the system disk will be renamed to FOO.BA.

REWIND

The REWIND command rewinds the specified cassette drive and returns to the DISKMON keyboard monitor. The format of the REWIND command is:

REWIND#N (CR)

where N is a cassette drive number to be rewound (4 through 7).

MAXIC BASIC 01000 - 62000
ST 5000

MAXIC GAMES 63235 -
ST 5000

MINI BASIC 1000 - 25000
ST 5000

MINI GAMES 34350 -
ST 5000

RUN

The RUN command loads a memory run file (.GO format, not ASCII or any other format) and its memory control block and automatically starts execution at the starting location specified in the memory control block. The format of the RUN command is:

RUN#N NAME (CR)

The digit 'N' represents one of the disk or cassette drives (0 thru 7) and defaults to unit 0 if not specified. 'NAME' represents the name of the file to be run and a .GO file extension is automatically added. The RUN command is equivalent to a LOAD followed by a START command.

If the file specified cannot be found, the error message "NAME.GO NOT FOUND" will be displayed. If no name is typed, the error message "WHAT?" will display. (See LOAD.)

EXAMPLES:

RUN#1 TEST (CR)
: The file TEST.GO if found on Unit #1 will be loaded and started automatically.

R BASIC (CR)
: The program BASIC.GO if found will be loaded from the system device (Unit 0) and automatically started.

RUN TEST.SP (CR)
: The program TEST.GO if found will be loaded from the system device and automatically started. Note that the .SP extension was ignored.

SAVE

The SAVE command allows the user to save the program currently in memory on a specified disk or tape unit. The format of the SAVE command is:

SAVE#N NAME XX-YY* SA (CR)

The digit 'N' represents one of the disk or cassette drives (0 thru 7) and defaults to unit 0 if not specified. 'NAME' represents the file name the user is assigning. The extension .GO is automatically added to the file to show that this is a memory run file produced by the SAVE command. 'XX' represents the starting octal or hex page number and 'YY' the ending page number of the memory area that is being stored away (i.e., pages XX to YY inclusive will be saved). 'SA' is the full octal or hex page and byte address (location 23 on page 10 enters as *10023) of the starting point for this program. This information will be stored away with the program as a memory control block. If these items are not specified, the required information will be taken from the current memory control block (refer to the LOAD and RUN commands for more information.)

When saving a program, the directory of the specified device will be searched for the file name specified and, if found, that program will be deleted after the new one is saved.

If the page numbers are typed incorrectly, or if no file name is included, the error message "WHAT?" will be displayed. If insufficient space is available, the message "NO SPACE" will be displayed. If an error occurs during a SAVE, the program currently in memory has not been saved. The memory image, however, is still intact.

It is possible to attempt to SAVE memory from pages 340 to 347. This should not be done since DISKMON's resident code is located in this area. The DISKMON resident code can be destroyed by attempting to LOAD or RUN such a program.

To incorporate error correction patches in saved programs, type LOAD NAME (CR). Use DTO or DTH to make the desired changes. Then type SAVE NAME (CR) The old version is deleted and a new version created using the same memory limits and starting address as those for the old version.

EXAMPLES:

SAVE#3 PROG 1-37* 1000 (CR)
: Saves the program PROG.GO located at page 1 thru page 37 inclusive with a starting address of 1000 (i.e., location 0 on page 1) on disk drive 3.

SA TEST* 1010 (CR)
: Saves the program TEST.GO located at the page limits currently in the memory control block and with a starting address of 1010 on the system disk, drive 0.

SA TEST.SP (CR)
: Saves the program TEST.GO using the memory limits and starting address currently in the memory control block on the system disk, drive 0. Note that the .SP extension is ignored and a .GO extension is automatically added.

START

The START command starts execution of a program at a specified memory location. The format of the START command is:

START SA (CR)

where 'SA' is a full octal or hex page and byte starting address. If 'SA' is not specified, execution will start at the 'SA' of the last program that was specified in a LOAD, RUN or SAVE command.

EXAMPLES:

START 10000 (CR) — Start program execution at location 0, page 10.

ST (CR) — Start program execution at the starting address included in the last LOAD, RUN or SAVE command.

TV

The TV command allows the user to specify whether the output should be formatted for a 16 x 32 or a 16 x 64 TV display. The format of the TV command is:

TV NN (CR)

where NN is either 32 or 64.

This change will only stay in effect until changed or until the system is reloaded. To make the change a permanent part of the bootstrap code, use the BUILD command.

WRITE

The WRITE command allows the DISKMON user to record a standard Digital Group audio cassette in the Suding 1100 baud format. The WRITE command has the following format:

WRITE SA-EA (CR)

where 'SA' and 'EA' represent full octal or hex byte and page addresses. These values will be used as the starting and ending addresses for the cassette write.

The display "WRITING" will appear and control will return to the DISKMON keyboard monitor after the cassette has been written.

EXAMPLES:

WRITE 10000-20100 (CR) — Writes a standard audio cassette after loading the starting and ending addresses with 10000 and 20100 (i.e., memory from 10000 to 20100 will be written out).

ZERO

The ZERO command zeros the directory of the specified disk or cassette unit. An option ("!") allows zeroing and copying the DISKMON system overlays onto the disk to make it a system disk. The format of the ZERO command is:

ZERO#N (CR)

The digit 'N' represents one of the disk or cassette drives (0 thru 7) and defaults to zero if not specified. Specifying unit zero or defaulting to unit zero will zero the system disk. This is not necessarily a desirable thing to do since few people want the system disk zeroed unless they are trying to operate on a single drive system. Therefore, the system will display the message: "ARE YOU SURE?" and wait for a character to be typed. If 'Y' is typed, it assumes that you do want a zeroed system disk and proceeds to do it. Typing any other character returns the user to the DISKMON keyboard monitor. A disk zeroed in this way is a storage-only disk and cannot be used as a potential system disk since it does not contain room for the DISKMON system overlays.

If the user wants the specified disk to be zeroed as a system disk the command string would be:

ZERO!#N (CR)

This zeros the directory of the disk unit number 'N' providing room for the system overlays, and then copies the system onto that disk. If '#N' is omitted, the system will assume that you want the system cassette zeroed and will give you the "ARE YOU SURE?" message and wait for the keyboard response described above.

To zero a cassette the user types the command with a cassette device number (4-7) followed by a space and the length of the cassette. The command

ZERO#4 45

will adjust for a C-45 cassette. (NOTE: Cassettes are always zeroed as non-system cassettes.) If no cassette length is specified, a C-30 is assumed.

The **SZE**RO form of the command is the same as the regular form except that DISKMON will display the message "SWITCH DISKS (SPACE)" before it does the zeroing. This is for use in a single drive system. At this point, the user inserts the disk to be zeroed into the system drive and types a space. When the zeroing is complete, the message "SWITCH DISKS (SPACE)" reappears. At this point, the user re-inserts his system disk and types a space to enter the keyboard monitor.

EXAMPLES:

ZERO#2 (CR) — Zeros the directory of disk unit 2 as a storage-only disk.

ZERO!#1 (CR) — Zeros the directory of disk unit 1 as a system disk and copies the system onto that disk.

ZERO! (CR)
ARE YOU SURE?

YES	**Zeros** the directory of the system disk (unit 0) if the response to the question is 'Y'. The system will display "YES" and the zeroing takes place.
ZE#5 60	**Zeros** cassette number 5 as a non-system C-60 cassette.

PART 3
DISKMON SYSTEM STRUCTURE

DEVICE LAYOUTS

System Disk Layout

Block	0-3	Device directory
	4-11	Resident boot code
	12	1 page scratch area (used by DTX and PHIWRT)
	13-14	Reserved
	15-20	1K directory scratch area
	21-100	Overlays
	101-3350	File storage

Non-System Disk Layout

Block	0-4	Device directory
	5-3350	File storage

Phideck Cassette Layout

Block	0-4	Device directory
	5-7	Overrun guard blocks
	10+	File storage

SYSTEM OVERLAYS

Overlay	Blocks	Routines
1	21-22	Main line, LINPUT, EXECUT
2	23-24	Directory handling
3	25-26	Error routine
4	27-30	Phideck routines
5a	31	Phideck record routine (relocated)
5b	32	DTX register display routine (relocated)
6	33-34	SAVE, LOAD, RUN, START, GET and PUT commands
7	35-36	DIRECTORY, DELETE, RENAME commands, DONAME and PRNAME routines.
10	37-40	ZERO, SZERO, BUILD, SBUILD, BOOT, DOBOOT, and PRBOOT commands.
11	41-42	ALTER, INSERT, OCTAL, HEX, MOVE, READ and WRITE commands.
12	43-44	DTX memory commands
13	45-46	DTX register commands
14	47-50	FORMAT command, ARESUR routine.
15	51-52	COPY command
16	53-54	More copy command (loads into pages 6-7)
17-23	55-66	Reserved for future use by The Digital Group, Inc.
24-30	67-100	User defined.

FILE NAMES AND DIRECTORY STRUCTURE

When file names are typed in, they consist of up to six letters followed by a period and up to a two-letter extension. When they are utilized in the system, they are modified to 8 bytes with nulls filling out omitted characters and the period eliminated. The MSB of each character byte must be high.

Two DISKMON routines are available for converting names from one form to another. See the section on Miscellaneous Routines for more information.

Directory Format

Every DISKMON device contains a directory on blocks 0-3. Every directory has the following format:

Byte (octal)	Function
0	First block of file storage
1	Number of directory entries (counting empties)
2-13	First directory entry
14-25	Second directory entry and so on to the end of block 3

Each directory file entry consists of the following:

1st 6 bytes	File name
Next 2 bytes	File extension
Next 2 bytes	Length of file

The MSBs of the filename and extension are always high. If the first byte of the file entry is a 0, this is an empty entry and represents available file storage space.

'GO' FILE FORMAT

Structure:

1st block	Memory control block
Remaining blocks	Byte-for-byte image of memory contents saved

The memory control block contains:

Bytes	0-1	Program execution starting address
	2	First page file loads into
	3	Last page file loads into
	4-377	Unused

DISKMON SYSTEM DESCRIPTION

The DISKMON operating system uses the 2K on pages 340_8-347_8 for its resident routines, variable storage, and overlay area. All other memory must be contiguous from page 1. Although DISKMON uses portions of lower memory to execute commands, it saves and restores those areas on the system device, thereby remaining invisible to any user programs. Only the COPY, BUILD, and ZERO (with exclamation point option) commands leave lower memory altered.

DISKMON's code is divided into its resident routines and its overlay routines. All routines are available to user programs.

The resident code includes the disk driver routines, the TV driver routines, the directory and scratch area routines, and the routine used to access non-resident routines. A jump table is provided to all major resident routines to allow future versions of DISKMON to be compatible with user-written programs.

The following is a description of each of the resident routines provided in the jump table. The octal address of each routine in the jump table is provided.

READ (340034) and WRITE (340037)

The READ and WRITE routines are DISKMON's I/O handling routines. The calling sequences of the two are:

A = unit number (0-3 for disks, 4-7 for Phidecks)
BC = block count
DE = tape or disk block
HL = buffer address

If the I/O operation executes properly, the A register is returned 0 and the zero flag is set. If an I/O error occurs, the zero flag is returned reset and the A register indicates the type of error:

1 = read CRC error, or write-fault on disks during write
2 = ID read error
3 = device error

Directory and Scratch Area Routines

DISKMON maintains a directory buffer at the top 4 pages of lower memory. This address is stored at the location DIRBUF (345016). Whenever DISKMON uses this area it saves the original data on the scratch area of the system disk (blocks 15-20) and reads it back when it has finished.

SCRTCH (340050)

Writes the directory buffer area onto the system disk (blocks 15-20) if it has not already been written there.

READIR (340056)

Reads directory of the device indicated by the A register into the directory buffer. Saves directory buffer (using SCRTCH) if it has not been done. READIR will also write out a resident directory if it detects one that has been modified.

UNSCRH (340053)

Reads the scratch area of the system disk (blocks 15-20) back into the directory buffer area if it has been written there. If a modified directory exists in the directory buffer it will be written out first. This routine is called to terminate any extended mode directory accesses (see Directory Handling Routines).

WRTDIR (340061)

Writes the resident directory back onto its respective device if it has been modified.

TV Routines

SPACE (340064)

Prints a space on the video terminal. Register A returned 0.

TV (340067)

Prints the ASCII character in the A register on the video terminal. A returned 0.

EDITOR (340072)

HL point to message in memory to be printed.

Coding of message:

0 = end of message
1-177$_8$ = number of spaces to be printed
200-376$_8$ = character to be printed
215$_8$ = finish line (i.e. carriage return)
377$_8$ = erase screen and leave cursor at home position.

A register returned 0
B register may be returned 0

MLTSPC (340075)

Prints the number of spaces contained in the A register on the video terminal. A and B returned 0.

ERASE (340100)

Erases video and leaves cursor at home position. A, B, and C registers returned 0.

FINLIN (340103)

Prints spaces to the end of the current line (i.e., carriage return). A and B returned 0.

BS (340106)

Backs up video one position. A returned 0.

KEY (340111)

Displays a cursor and waits for a keyboard entry. The character is returned in the A register.

Non-Resident Routines

The following is a description of the various DISKMON non-resident routines. The octal number given for each routine is that routine's number. Therefore, it is the number that should follow the call to SUBRTN.

LININP (1)

Linput is DISKMON's line input routine. Calling sequence:

C = prompt character
E = number of characters allowed in line (extras ignored)
HL = line buffer pointer (returned unaltered)

If the user types an 'ESC' while LINPUT is executing, it will restart DISKMON's keyboard monitor. LINPUT will accept 'RUB' and 'CTRL-U'. A carriage return will terminate the input line. This will be represented by a 0 at the end of the line input buffer.

EXECUT (2)

EXECUT is DISKMON's command line execute routine. When called, EXECUT will execute any legal command pointed to by the HL register pair. The command characters must have their MSBs high. The line is terminated with a 0.

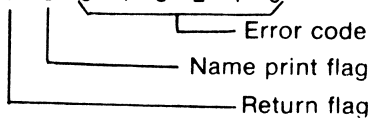
If the command executes properly, control will return to the calling program (except in the case of RUN, START, DTO, or DTH) with the carry flag reset. If the command is unrecognized or if there is a syntax or logic error in its arguments, EXECUT will return with the carry flag set. If another type of error occurs, the DISKMON error routine will be executed with eventual return to the DISKMON keyboard monitor.

ERROR (10)

ERROR is DISKMON 's error handling routine. It prints out an error message and then either returns to the calling program or to the keyboard monitor.

Calling sequence:

A register: 7 6 5 4 3 2 1 0



Return Flag: = 1 Return to caller
= 0 Return to keyboard monitor

Name Print Flag = 1 Print file name at NAME (345002) before printing error message
= 0 Do not print name

Error Codes:

1-3 I/O errors
If an I/O error occurs, the source of the error can also be printed. This is indicated by the E register:
0-no source to print
1-DIRECTORY
2-SCRATCH
3-SYSTEM
4-FORMAT

4- IMPROPER CLOSE
5- DIRECTORY FULL
6- NO ROOM
7- IS NOT AN IMAGE FILE
10- NOT FOUND
11- ALREADY EXISTS
12- CANT BUILD NON-SYS
13- IMPROPER OVERLAY

Directory Handling Routines

To simplify file calls, all directory handling routines (except RENAME) have the same calling and returning register format:

A = unit number (returned unaltered)
BC = block count
DE = disk or tape block
HL = file name pointer

LOOKUP (3)

Looks up specified file in specified directory.

Calling:
A = unit number
HL = address of file name

Return:
BC = blocks in file
DE = first block of file
CARRY flag = 1 if file is not found

This is a typical file read sequence:

```
LD      A, UNIT     A = unit
LD      HL, NAME    HL = name pointer
CALL    SUBRTN      CALL LOOKUP
JP      C, NOTRND
DB      3
LD      HL, BUFFER      HL = buffer pointer
CALL    READ        Read File
JP      NZ, IOERR
```

DELETE (4)

Deletes specified file from specified directory:

Calling:
A = unit number
HL = address of file name

Return:
CARRY flag = 1 if file is not found

RENAME (5)

Renames specified file in specified directory.

Calling:
A = unit number
DE = pointer to new file name
HL = pointer to old file name

Return:
CARRY flag = 1 if file is not found

Miscellaneous Routines

OVRLAY (340045)

Reads overlay indicated by A register into the overlay buffer (pages 346_8 and 347_8). This routine should be used only rarely, as calls to routines on overlays should normally be done with the SUBRTN routine.

SUBRTN (340042)

This routine is used to call subroutines on the various DISKMON overlays. Each non-resident routine has a number. Using the routine lookup table RTNTAB (345200), SUBRTN brings in the correct overlay and turns execution over to the correct routine. When the routine is finished, SUBRTN reloads the overlay that was originally resident if SUBRTN detects that the call originally came from within the DISKMON code area. This allows routines in one overlay to call routines in another overlay.

All calls to SUBRTN are followed by the number of the routine that is desired. When the routine is finished, control is returned to the instruction following the routine number.

```
CALL  SUBRTN  Call to SUBRTN
DB    N       Routine number
              Control returns
              here
```

ENTRY (6)

Searches specified directory for empty area big enough for specified length to be saved.

Calling:
A = unit number
BC = desired number of blocks

Return:
BC = actual number of blocks in empty area
DE = first block of empty area
CARRY flag = 1 if no empty area was big enough for desired file

If ENTRY is called with BC=0 a search will be made for the largest empty area on the device. In this case the CARRY flag is returned=1 only if no empty area was found.

CLOSE (7)

Inserts new file name into specified directory.

Calling:
A = unit number
BC = number of blocks contained in new file
DE = first block of new file (must be the first block of an empty area, but the new file need not fill up the empty area)
HL = pointer to name to be inserted into directory

An error encountered in the close routine will result in the message "IMPROPER CLOSE" to be displayed.

Typical File Save Sequence:

```
LD    A, UNIT      A=unit
LD    BC, BUFSIZ   BC=blocks needed
CALL  SUBRTN       Call ENTRY
DB    6
JP    C,NOROOM
LD    BC,BUFSIZ
LD    HL,BUFFER    HL=buffer address
PUSH  DE           Save 1st block
CALL  WRITE        Write file
JP    NZ, IOERR
POP   DE           Get 1st block
LD    A,UNIT       A=unit
LD    BC,BUFSIZ    BC=length
LD    HL,NAME      HL=name pointer
CALL  SUBRTN       Call CLOSE
DB    7
```

Extended Mode Directory Accesses

When using the directory handling routines extensively, it may be useful to have DISKMON retain the directory of a certain device in memory through a number of directory operations. For example, if the user program wanted to LOOKUP a large number of files at the same time it would be wasteful of time to bring the directory in more than once.

To keep a directory in after an access, call the desired directory handling routine with the MSB of the A register set to a 1. This will cause DISKMON to leave the directory resident. The directory may now be used as many times as desired provided the directory routines are called with the MSB of the A register set to a 1.

Calling a directory routine with the MSB of the A register set to a 0, accessing another directory, returning to DISKMON , or calling UNSCRH (340053) will cause the directory to be written out if modified and will cause the scratch area to be read back in. The user should remember that while using this feature, he should not try to use the upper 1K of memory, as this is where the directory is located. The original contents of this area are read back when the directory access is finished.

Miscellaneous Routines

PHIRD (11) and PHIWRT (12)

Phideck read and write routines. These routines are used by the main READ and WRITE routines.

REWIND (13)

Rewinds Phideck unit specified by A register.

SUDRD (14)

Reads audio cassette into memory until byte count is exhausted or trailer is reached.

Calling:
BC = byte count
HL = buffer pointer

Return:
BC = actual bytes read

BYTERD (15)

Reads a single byte from audio cassette. Byte is returned in the D register.

LEADER (16)

Writes 5 seconds of leader/trailer onto audio cassette.

BYTEWR (17)

Writes a single byte of data in the A register onto audio casette.

DONAME (20)

Converts file name from typed format pointed to by the HL register pair to directory format and leaves the name in the name buffer (345002). HL is returned pointing to the first character beyond the end of the typed format name.

PRNAME (21)

Prints the file name contained in the name buffer (345002).

GETNUM (22)

Converts the ASCII number pointed to by the HL register pair into binary. The binary value is returned in HL, and DE is returned pointing to the first character beyond the number. The number mode (OCTAL or HEX) is determined by the current radix setting. This information is stored at location RADIX (345030). Radix = 0 for octal or 1 for hex.

BRKPNT (23)

BRKPNT is used by DISKMON to enter the breakpoint register display upon any RST60.

ARESUR (24)

Prints the message "ARE YOU SURE?" and waits for a keyboard entry. A "Y" will cause ARESUR to print "YES" and return to the calling program. Any other response will echo "NO" and will return to the keyboard monitor.

Adding Commands

Routine numbers from 34 to 70 correspond to the DISKMON keyboard commands. All commands expect the HL register pair to point to the command line and the A register to indicate any specified unit number. The commands in the routine table are in the same order as the commands in the character lookup table for the EXECUT routine. To add commands, therefore, the user must only add the command letters to that table on overlay 1 and put in an overlay and overlay position number at the end of RTNTAB. See the source listing for details.

Appendix A
DISKMON ERROR MESSAGE SUMMARY

I/O Error Message Format

(SOURCE) (ERROR TYPE) ERROR ON #N

(SOURCE) indicates where the error occurred. This may be a file name, DIRECTORY, SCRATCH, SYSTEM, or FORMAT.

(ERROR TYPE) indicates what type of error occurred. This may be CRC READ, ID READ, READ or WRITE DEVICE ERRORS, or WRITE-FAULT.

N = unit on which error occurred

Miscellaneous Errors

WHAT? Unrecognized command, or syntax or logic error in command arguments.

IMPROPER CLOSE
A user program called CLOSE with improper arguments.

DIRECTORY FULL
A CLOSE was requested for a device with 102 directory entries.

NO ROOM
There was no room on the specified device to execute the specified SAVE or PUT command.

NAME.EX IS NOT AN IMAGE FILE
The file specified in a RUN or LOAD command does not have a proper 'GO' header.

NAME.EX NOT FOUND
The file specified in a command was not found on the specified device.

NAME.EX ALREADY EXISTS
An attempt was made to RENAME a file to a name that already existed on that device.

CANT BUILD NON-SYS
A BUILD was attempted on a non-system disk.

IMPROPER OVERLAY
An INSERT was requested but the overlay was not present in pages 1 and 2. Note that the first and last byte of each overlay must equal the overlay number.

SYS ERR N
This message, followed by a CPU halt, occurs if DISKMON is loading an overlay when an error occurs. After the error condition is fixed, press RESET to restart DISKMON . The number in the message indicates the type of error:

1—CRC error while reading overlay
2—ID read error while reading overlay
3—Device error while reading overlay
4—Non-system disk inserted in system drive