FLOPPY DISK OPERATING SYSTEM FOR HP 21XX SERIES COMPUTERS FDOS-HPC

Dicom Industries
715 N. Pastoria Ave.
Sunnyvale, CA 94086

Telephone (408) 732-1060
TWX(910)339-9285

P/N 933-05098-01 Rev. F.
January 1975

Manual P/N 505-01176-01 F

CONTENTS

I. INTRODUCTION

The Dicom Floppy Disk System (FDS) is complemented by a complete line of supporting software for the Hewlett Packard 21XX Series computers. The FDS can be used as a peripheral in either the SIO or BCS environment, or it can be used as the primary I/O device under Dicom's Floppy Disk Operating System (FDOS).

An operating system is an organized collection of programs which increases the PRODUCTIVITY of a computer system. An operating system's main function is to aid in the preparation, translation, debugging, loading, and execution of programs. This is accomplished by including a peripheral device in the system that can be completely controlled by the computer, thus providing a large program/data base for use by the operating system. In FDOS, up to a half-million words of random-access storage is provided via disk, with an additional half-million bytes of serial-access storage provided via cassettes.

The various translators, loaders, and other software modules of the operating system are stored on the library device (in this case a disk) for use only when needed; thus the amount of core memory needed for efficient operation is significantly reduced. Since the operator requests a translator (assembler, compiler, etc.) from disk via an executive command typed on the console device keyboard (instead of loading it from paper tape), the overhead time can be significantly reduced and the efficiency can be significantly increased. Also, since the computer has complete control of the media containing the input data, the translation of the input data can be accomplished automatically, thus eliminating operator intervention and increasing system throughput (most translators must make multiple passes over the input data).

FDOS is a collection of routines which control the operation of application programs and provide them with certain important capabilities, primarily in the area of access to peripheral devices, i.e., I/O. FDOS supports the HP Assembler, Cross-reference Table Generator, Editor, FORTRAN Compiler, and other HP and user-provided application programs. It allows these programs to perform I/O operations using a wide variety of peripheral equipment, including floppy disks, cassettes, paper tape, teletype and lineprinter. These devices are treated uniformly by the operating system, so as to allow the application programs to be device-independent.

FEATURES OF FDOS

* Immediate usefulness; no special languages or complicated procedures to learn.

* Immediate (random) access to all system programs.

* Uniform treatment of peripheral devices.

* Keyboard and batch processing modes.

* Small (approximately 140B words) resident system with all remaining memory available to user programs.

* SIO driver package which combines console device, floppy disk, cassette, high-speed paper tape, and line printer drivers, using common routines to conserve memory.

* Standard programming packages integrated into FDOS, with complete floppy disk, cassette, paper tape, and line printer I/O facilities: includes editor, assemblers (Floating Point, EAU, and Non-EAU), cross-reference table generator, debugger, FORTRAN compiler, relocating loader, relocatable library, and others.

* Facilities for converting programs between the various media supported (paper tape, disk, and cassette).

* Configurable for I/O configuration and memory size at system generation time.

MINIMUM FDOS HARDWARE CONFIGURATION

* HP 21XX Series computer with 8K of memory (DMA, EAU, memory parity, memory protect, time base generator, etc., are NOT required).

* Console device (HP-interfaced teleprinter or cassette system "deck zero" TTY/CRT).

* Single drive Floppy Disk System.

ADDITIONAL HARDWARE SUPPORTED

* Up to a total of four floppy disk drives

* Up to a total of 32K memory.

* Dicom Cassette Magnetic Tape System (CMTS) or Cassette Magnetic Tape Terminal (CMTT).

* High-speed paper tape reader.

* High-speed paper tape punch.

* Line printer (HP-interfaced or cassette system "deck zero" line printer).

* Other peripheral devices with user-supplied SIO or BCS compatible drivers.

FDOS SOFTWARE MODULES

The FDOS user is supplied:

* Skeleton system disk containing:

  Configured FDOS for immediate use.
  Unconfigured Bootloader.
  System Generator.
  Unconfigured System Loader.
  Unconfigured SIO Driver Package.
  Unconfigured Executive.
  Assemblers (Floating Point, EAU, and Non-EAU).
  Editor.
  Cross-Reference Table Generator.
  Debugger and Disk Utility Package.
  FORTRAN Compiler (Floating Point/EAU and Non-EAU).
  Prepare Control System.
  Relocating Loader, BCS Drivers, and .IOC.
  Relocatable Libraries (Floating Point, EAU, and Non-EAU).

* Diagnostic (on paper tape, cassette or disk)

  A configured, "standard" FDOS software system is provided with a pre-determined hardware configuration; this allows the user to put the FDOS "on-the-air" immediately.

  The System Generator is run to produce a configured System Loader, SIO Driver package, and Executive. These can then be written out to the "System" files on disk, and the System Generation process need not ever be performed again.

The Assemblers, Editor, Cross-Reference Table Generator, FORTRAN Compilers, Disk Prepare Control System, and Relocating Loader modules are modified versions of standard HP programs. The Relocatable Libraries, the .IOC. and BCS Drivers are unmodified HP programs. The Debugger is a stand-alone octal debugger designed for use either with an HP TTY, or with a cassette system "deck zero" console device.

II. FDOS ORGANIZATION

The FDOS is a collection of programs designed to work together, giving the user a unified treatment of programs and data files. FDOS is specially designed to assist in the development and execution of programs.

To this end an editor, assembler, FORTRAN compiler, debugger, and other software are stored on disk for quick access when needed. These programs are designed to use any FDOS I/O device for the various aspects of their input and output, and programs generated by the assembler or FORTRAN compiler can later be integrated into the system. Such user-generated programs can be stored on disk, can make calls on system I/O drivers and other utility routines, and can communicate with the "Executive" for the purpose of accepting parameters at run time, of loading other programs, or of returning control to the "Executive".

It will be especially easy for CMTOS users to learn FDOS, since the commmand structures of the two operating systems are quite similar.

II.A. FDOS PERIPHERAL DEVICES

The FDOS is designed to allow uniform handling of data from different peripheral devices regardless of the media involved. Under FDOS, each peripheral device (i.e., paper tape reader, paper tape punch, line printer, etc.) or sub-device (Teletype keyboard, Teletype printer, disk drives D, E, F, and G) is considered to be a physical device.

In order to take maximum advantage of the disk drives, they are broken down further. Each disk drive has four logical units assigned to it; D1, D2, D3, D4, E1, E2, ... , G4; each of these sixteen (possible) logical units are considered to be a physical device. Each of the logical units within each disk has an associated Logical Unit Directory (LUD) which resides on a certain fixed sector of each disk. The contents of each LUD, which can be examined by the Logical Unit Directory command (LD), is a list of file names. Thus, for example, if the LUD for unit E1 consisted of the names

SRC1
SRC2
SRC3

and the LUD for unit E2 consisted of the names

SRC3
SRC2
SRC1

then a command to rewind logical unit E1 and copy three files from that unit to, say, cassette deck 2 would leave the three files SRC1, SRC2 and SRC3 on the cassette in deck 2 in the given order. If the same command were given using logical unit E2, then deck 2 would receive the same three files, but in the opposite order.

In addition to the logical unit structure of disk cartridges (drives), there is also a structure of named files. There is a Disk Directory, kept in a certain fixed disk sector, which lists the names and locations of all files actually on that disk. A list of the names of ALL files actually on a specific disk can be obtained via the Disk Directory command, DI. It should be noted that the names of the files in the LUDs will also be printed IF there is actually data in the associated file.

Peripheral devices of FDOS that are actually file-oriented, e.g. the cassette decks and the disk logical units, can contain more than one file. To this end, commands are provided to position to specific files (Skip, Queue, Rewind, etc.), and to manipulate files (Copy, Verify, Save, List, etc.).

II.B. FDOS DEVICE STRUCTURING

The uniformity in device handling is accomplished in FDOS by having two sets of devices: physical devices, which correspond to actual system hardware (peripheral devices), and functional devices, which correspond to tasks performed by the system.

Programs such as the assembler refer only to functional devices. The system translates these I/O requests into operations on physical devices and handles all device-specific considerations. For example, the normal operation of the assembler requires it to position the input unit, read through the file, position the input unit again, read through the file again, producing binary output on the punch unit, list output to the high-speed list unit or program list unit, write an EOF after the binary file on the punch unit, and, if no cross-reference has been requested, write an EOF after the list file on the high-speed list unit. The assembler does not need to make special provision for paper tape input or output (such as halting between TTY print and punch, generating leader/trailer, etc.).

It does not need to set up a special pointer for lineprinter output, or avoid writing file marks on tape or disk when the listing has gone to the lineprinter. All it needs to do is make calls to the system I/O driver to implement the sequence of operations specified above, and the system I/O driver handles all of these device peculiarities.

This arrangement is similar to the HP BCS arrangement in which an EQT and an SQT are defined.

The physical and functional units recognized by FDOS are shown in Table II-1. This table is the key to understanding FDOS device handling. The left side of the table shows all of the physical devices which can be supported by the system. Any particular system has some subset of these, specified at system generation time.

A table, called the Physical Device Table (PDT) is set up during system generation. For each of the devices on the left side of table II-1, an entry in the PDT specifies its existance and capabilities on that particular system. For example, if the system has a high-speed reader but not a punch, the high speed tape entry will show input capability but not output capability.

The right side of table II-1 lists the functional devices recognized by the system. The Functional Device Table (FDT) specifies which physical device is assigned to any given functional device, and also contains information about the binary vs. list capabilities of the particular functional device. A functional device can be either a binary device or a list-capable device. List-capable devices interpret I/O calls specifying negative counts as page control commands, while binary devices interpret these as commands to output in binary mode (B or BA, as specified in Section II.C).

In the HP SIO environment a peculiar arrangement is provided for formatting listings. Normally an SIO driver accepts calls of the form

```
              (A) = Data count
              (B) = Buffer address
              JSB Driver.
```

The data count specified is either the number of ASCII characters to be output or minus the number of binary words to be output. That is, a negative data count specifies that binary data is to be output.

Certain devices, such as a lineprinter, are not normally used for binary output. The SIO drivers for such devices are programmed to interpret certain "binary" calls as page spacing requests. The driver keeps track of the current page position by means of a line counter. When the line count reaches the maximum number of lines allowed on a page, the driver causes the device to do a "page eject", either by means of a Form Feed, or by means of six Line Feeds.

In FDOS a similar page spacing facility is provided. Basically, paging properties are associated with <u>functional</u> units. Each functional unit has associated with it a 5-bit code which specifies

(1) The unit is binary; that is, no page spacing requests are recognized,

- - - or - - -

(2) The paging parameters:

  (a) three bits specifying one of seven (1 to 7) system line counters assigned to the unit; a line counter of Ø specifies <u>no</u> line counting (thus, no page spacing) for this unit.

  (b) two bits specifying one of four page feed character options;

    i) L - an appropriate number of Line Feeds (to bring the line count to 66) is output, and the line counter is reset.

    ii) F - a Form Feed. (ASCII 14) is output and the line counter is reset.

    iii) N - no output accompanies the page advance. The line counter is reset.

    iv) X - the extra page character (specified at System Generation Time) is output, and the line counter is reset.

The FDT entry for a list-capable units contains an index (between Ø and 7) to a table of line counts and a code for the choice of page feed character (Line Feed, Form Feed, Null, or "extra page character").

The FDT entry for any particular functional unit can be modified by the operator by means of the assign (AS) command of the Executive.

To use this command, the operator types

```
    AS IUAS FU,PU,LC,PC
```

where FU = mnemonic code for the functional unit.
      PU = mnemonic code for the physical unit.
      LC = index of the line counter.
      PC = code for page character.

If the LC and PC parameters are not specified, they are left alone in the FDT entry. Since the operator will almost never need to change these, the ordinary format of the AS command is simply

```
          AS FU,PU
```

to assign the physical device PU to the functional device FU.

Since it is important for the operator to be able to keep track of the FDT assignments, the Executive provides ways to do this. To find any given assignment, the operator can type

```
          AS FU,?
```

The system replies by typing

```
          PU Name  Line-counter  Page-character
```

where PU is the mnemonic code (see table II-1). If the operator wants the entire FDT printed, the command

```
          LA UNIT
```

can be used. The entire table is printed on "UNIT" with an entry in the above format for each of the FDT entries.

In order that the user need worrry as little as possible about the FDT, a complete FDT is assembled with the Skeleton FDOS. At System Generation time any of these can be modified by statements in the format used with the Assign command (FU,PU,LC,PC).

The FDT is stored in the "warm start" file on the system disk (SYS2). Thus, an X7542B, X7544B or X77ØØB restart will restore the FDT to its system generation time values, but an X754ØB restart, such as used by ASMB, EDIT, etc., to reload the Executive, will not modify the FDT.

II.C. FDOS DATA STRUCTURE

All information stored or accessed by FDOS programs is organized into files. These files are composed of records. A record is a collection of characters and must conform to one of four basic formats (which conform to standard HP philosphy):

* A - ASCII. Each record consists of 7-bit ASCII codes and ends with a line feed character (octal 12). On output, the eighth (parity) bit is always set. On input, the eighth (parity) bit is always cleared.

* B - Relocatable binary. Each record consists of an even number of 8-bit characters, to be taken in pairs to represent 16-bit computer words. The first character of each pair represents the most significant half (bits 15-8) of the corresponding word.

  The first character (left half of the first word) of the record specifies the total number of words in the record. The last word of the record is the checksum (the sum of the second through the next-to-last words of the record).

* BA - Absolute binary. Each record consists of an even number of 8-bit characters, to be taken in pairs to represent 16-bit computer words. The first character of each pair represents the most significant half (bits 15-8) of the corresponding word.

  The first character (left half of the first word) of the record specifies the total number of words in the record, minus 3. E.g., if there are 128 words in the record, this count will be 125. Absolute binary format differs from relocatable binary format in this respect. The last word of the record is the checksum (the sum of the second through the next-to-last words of the record).

* Fixed length binary (for BCS programs only). Each record consists of an even number of 8-bit characters, to be taken in pairs to represent 16-bit computer words. The first character of each pair represents the most significant half (bits 15-08) of the corresponding word. The record has no included word count; exactly as many words (or characters) are read as were asked for in the call.

When taking input from paper tape, on which there are no physical record delimiters, the data mode must be known in advance, and is used to determine record boundaries. Thus, e.g., a program reading an ASCII record will continue to take characters from the paper tape until a line feed character is encountered. If asked to read an absolute binary record, the program will take the first character read as a data word count (n), fill out the first word with the second character from tape, read the next 2(n+1) characters as the next (n+1) words of the record, while computing the sum (modulo 2^{16}) of those (n+1) words, then, finally read the next two characters from the tape as the last word of the record, and compare that last word (the checksum) with the sum of the previous (n+1) words.

In reality, the checksum computation described in the above example would not be performed by an I/O driver asked to read a record in absolute or relocatable binary format. Instead, the (n+3) or (n) words of the record are simply transmitted to the calling program, which can then perform the checksum computation itself.

The file structure of the disk follows the paper tape record format more closely than the cassette format, since no special record delimiters are recognized by the disk hardware. The disk structure supported by the hardware is its division into tracks and sectors. The number of sectors per track is a power of two, and the "disk address" contains the sector address in the least significant portion of the word with the track address in the next most significant bits. For this reason, the division into tracks is largely unimportant to the software, and we speak of the disk address or sector address of data without explicit mention of the track.

A disk sector contains 128 16-bit words; these are assigned in FDOS as follows:

First word:	"Name"; A word derived from the file name and serving as a validity check.
Second word:	"Pointer"; Sector address of the next 128-word block of the file. Zero denotes end-of-file.

Remaining 126 words: Data words.

The data words are used for storing information in the same format as used with paper tape. Thus, a disk file can be thought of as a paper tape laid out on the disk in a "chain" of sectors. The first sector contains the first 252 characters of the "tape" in its third through 128-th words. The 253-rd through 504-th characters are in the third through 128-th words of the second block, and so on. The last block of the file contains an end-of-file indication in its "pointer" word, marking the end of the "tape", and unused characters of the last block are filled with nulls, the "trailer" of the "tape".

Since the blocks of a disk file are chained together, it is only necessary to record the address of the first block of the file. This is the function of the disk directory. The disk directory contains a four-word entry for each file on the disk. Three words are for the name; the fourth contains the sector address of the first block of the file.

In order to go through a file of data stored in this manner it is only necessary to know the location of the first sector, since each sector contains a pointer to the next sector of the file. The disk directory, stored in track zero, sector 1 of each disk, contains the names (up to five characters) and starting sectors of the files on that disk.

The structure of files described above is appropriate to the random-access nature of the disk media. Many programs, however, are designed for use with a sequential access device. FORTRAN statements such as REWIND and ENDFILE have direct interpretations for cassettes but not for disks. Similarly, programs running in an SIO environment have no means of making random-access references to their I/O units. Thus, in order to achieve uniform treatment of all system I/O devices wherever possible, a system of "logical units" has been established for each disk.

There are four logical units on each disk. Each has assigned a sector in which its "logical unit directory" or LUD is stored. This LUD is a list of file names in a specific order. Those files, in that order, constitute the given logical unit, making it a sequential access device. The contents of the LUD's, and therefore the files in these logical units, are under control of the user (see QU, AQ, and LD commands).

II.D: Table II-1; FDOS FUNCTIONAL AND PHYSICAL DEVICES

P H Y S I C A L	D E V I C E S		F U N C T I O N A L	D E V I C E S	
Ø	Ø	CMTS Deck Ø	4Ø	DI	Data Input
1	1	CMTS Deck 1	41	PL	Program List
2	2	CMTS Deck 2	42	DP	Data Output
3	3	CMTS Deck 3	43	PK	Program Keyboard
4	HT	High-speed Paper Tape	44	HL	High-speed List
5	LT	Low-speed (TTY) Paper Tape	45	SC	Scratch
6	TY	Teletype keyboard/printer	46	CK	Executive Keyboard
7	LP	Lineprinter	47	CL	Executive Listing
1Ø	P1Ø	Unused	5Ø	AI	Auxiliary Input
11	P11	"	51	AL	Auxiliary List
12	P12	"	52	AP	Auxiliary Output
13	P13	"	53	AK	Auxiliary Keyboard
14	P14	"	54	AH	Auxiliary High-speed List
15	P15	"	55	AS	Auxiliary Scratch
16	P16	Nonexistent Device	56	BC	Batch Command
17	BK	Bit Bucket	57	BM	Batch Messages
2Ø	D4	Logical Unit D4	6Ø	S2Ø	Unused
21	D1	" " D1	61	S21	"
22	D2	" " D2	62	S22	"
23	D3	" " D3	63	S23	"
24	E4	" " E4	64	S24	"
25	E1	" " E1	65	S25	"
26	E2	" " E2	66	S26	"
27	E3	" " E3	67	S27	"
3Ø	F4	" " F4	7Ø	S3Ø	"
31	F1	" " F1	71	S31	"
32	F2	" " F2	72	S32	"
33	F3	" " F3	73	S33	"
34	G4	" " G4	74	S34	"
35	G1	" " G1	75	S35	"
36	G2	" " G2	76	S36	"
37	G3	" " G3	77	S37	"

II.E: TABLE II-2; SYSTEM TASK ASSIGNMENTS

The functional devices are assigned to system tasks in the following way.

1. Executive.

CK	On-line mode keyboard input.
CL	On-line mode executive messages.
BC	Batch mode command input.
BM	Batch mode executive messages.

2. Assembler.

DI	Source input.
DP	Binary output.
PL	Talk to operator (e.g., "TYPE CONTROL STATE.") Listings if HL not requested.
PK	Input from operator (e.g., ASMB statement).
HL	Listing output unless PL specified.

3. Editor

DI	Data input if "/D" specified.
AI	Data input if "/A" specified.
DP	Data output if "/D" specified.
AP	Data output if "/A" specified.
PL	Talk to operator.
PK	Answers from operator. Input of edit file if "/T" specified.
AK	Input of edit file if "/A" specified.

4. Cross-reference Table Generator

DI	Source input.
PL	Talk to operator. Listings if HL not requested.
PK	Input from operator (e.g., symbol range).
HL	Listings unless PL specified.

5. FORTRAN Compiler

DI	Source input.
PL	Talk to operator. Listings if HL not requested.
SC	Intermediate binary scratch file.
HL	List unless PL specified.
DP	Relocatable binary output.
PK	Input from operator.

6. Disk Prepare Control System

DI	Relocatable binary input.
PL	Talk to operator.
DP	Absolute binary output.
PK	Input from operator.

III. FDOS SOFTWARE STRUCTURE

III.A. FDOS BOOTLOADER

The FDOS Bootloader is a short program which is stored in the protected area of memory. After the Bootloader has been configured (see Section V.D), it is used to load the System Loader from disk; the Bootloader has a second entry that can be used to load a file from disk whose initial sector address is in the Switch Register. A listing of the Bootloader is contained in an appendix of this manual.

III.B. FDOS SYSTEM LOADER

The FDOS System Loader is the core-resident portion of FDOS; it is used to load absolute binary files into memory from disk. Most FDOS systems programs (as well as user-added programs, if directed within the program) return to a System Loader entry which causes the remainder of the operating system to be loaded and executed. The System Loader is also used by the Run command of the FDOS Executive to load files from disk. A listing of the System Loader is contained in an appendix of this manual.

III.C. FDOS SIO DRIVER

The SIO Driver provides device control and input and output of records in the ASCII, relocatable binary and absolute binary modes using disk logical units, cassettes, paper tape, Teletype, or lineprinter. It also provides a "system" entry, DIO, which accepts control commands and provides status returns (as opposed to SIO error halts).

III.D. FDOS MEMORY LAYOUT

The FDOS System Loader resides near the top of memory; it occupies locations X7534B through X7677B, where X = 1,2,3. . . for 8K, 12K, 16K, ... Just below it in memory is the Communications Region, which is used for interprogram communication. Below that, going down to around X34ØØB (depending upon the configuration) is the SIO Driver package. Programs which use the SIO Driver package (e.g., ASMB, EDIT, CROS) must preserve this area (the LWA of available memory can be found in location 1Ø6B). The Executive is normally overlaid by these programs. The Executive resides in locations X34ØØB through (X-1)4ØØØB. Its restart address is (X-1)4ØØØB.

The editor, assembler, etc., load into the lower portion of memory (preserving 1ØlB-1Ø4B and 1Ø6B) and use as much memory as they find available (the remainder minus the SIO Driver package, Communications Region, and the Bootloader in the protected area of memory).

III.E. FDOS DISK LAYOUT

The FDOS takes advantage of the <u>removability</u> of the floppy disk media by making each disk a self-contained entity. Each disk has a Disk Directory in track zero, sector one, and logical unit directories in track zero, sectors 4, 5, 6, and 7. These correspond, respectively, to logical units 4, 1, 2, and 3. In addition, five "fixed address files" (sectors 13B, 14B, 15B, 16B and 17B) are reserved for system use. The remaining sectors of the disk (2ØB thru 1777B) are originally placed on a free storage list, and then assigned to files as needed.

III.F. FDOS EXECUTIVE

The FDOS Executive provides batch or keyboard control of utility functions and program loading. Commands include copying of data from one device to another, control of cassettes or disk logical units, preparation and examination of directories, and running named files.

III.G. MEMORY RELEASING

The FDOS SIO Driver combines the drivers for the various peripherals supported into one Driver, thus eliminating redundant (unnecessary) coding and conserving space. During System Generation (see Section V.F) the user specifies which peripherals are not to be used; the corresponding section of the SIO Driver is thus released, and the appropriate portions of FDOS are moved up to give the released area to the user.

III.H FDOS SOFTWARE STRUCTURE OVERVIEW

FDOS is a software/hardware package designed to provide the user of HP 2100 or related computers with the utility functions necessary for the use of their computers with a wide varity of peripheral equipment. The FDOS software provides a modified SIO environment in which the exec, Assembler, Editor, FORTRAN and associated programs run. It provides the necessary interface with the HP BCS environment, so that relocatable programs generated by FORTRAN or the Assembler can be loaded in the usual fashion and run on the same (or other, if desired) hardware as the FDOS SIO environment programs.

The basic I/O device of FDOS is the floppy disk. However, SIO software is record-oriented. Furthermore, FDOS achieves a uniform treatment of I/O devices. Thus, the FDOS disk format is designed to make disk files look like very long paper tapes. Each disk file is a chain of disk sectors (128 words per sector). The first word of each sector is a file identifier derived from the file name. It is used by the driver for checking file integrity. The second word of the sector is a pointer to the next sector of the file. Special codes mark End-of-File or End-of-Disk. The pointer contains a disk address (track and sector). No drive bits are included in the address, since a given disk cartridge can be inserted into any drive.

The sectors of the disk are assigned to files from a list of free blocks, that is, a file composed of all of the unused sectors. The address of this free list and the names and addresses of all disk files are kept in the Disk Directory, which is stored in a certain fixed sector of each disk cartridge.

Files are attached to a logical unit by means of the QU and AQ commands. The files comprising a given logical unit may or may not be actual disk files. If a file name appears in a LUD but is not an actual disk file, then a file with that name is created if output is directed to it. If input from a non-existant file is requested, an End-of-Tape condition occurs on that logical unit.

The disk logical units are sequential-access devices, designed to resemble tape drives. The files of a logical unit are in a fixed sequence. A rewind command positions the unit to the start of the first file of that unit. Reaching an End-of-File on a logical unit causes the unit to be positioned to the start of the next file of that unit. Proceeding beyond the end of the last file of a logical unit creates an End-of-Tape condition.

FDOS supports up to four floppy disk drives, each having four logical units. In addition to floppy disks, FDOS supports a variety of other peripheral devices. The various physical devices supported are assigned codes by which references to them are to be made in the various Executive commands. The following codes and corresponding physical devices are recognized by the Executive:

Ø	CMTS Deck Zero. This is the "off-line" device of the Dicom Cassette Magnetic Tape System. Normally either a console (CRT or TTY) or a lineprinter is attached.
1 - 3	CMTS cassette decks one thru three.
HT	High-speed paper tape. This refers to two devices - the photoreader and the high-speed punch. Output to HT refers to the punch; input refers to the photoreader. Either, both, or neither can be present in the system hardware configuration.

LT	Low-speed paper tape. This refers to the TTY reader and punch devices of the HP-interfaced teleprinter.
TY	This refers to the keyboard/printer or keyboard/CRT interfaced by means of the standard HP TTY interface.
LP	This refers to the standard HP-interfaced Lineprinter. It is an output device only.
D1-D4	Logical units of drive D. The four floppy disk drives supported by FDOS are refered to by the letters D, E, F, and G. The logical units of each are treated as physical devices.
E1-E4	Logical units of drive E.
F1-F4	Logical units of drive F.
G1-G4	Logical units of drive G.
BK	The Bit Bucket (an infinite write-only memory).

In order that application programs which are independent of the hardware configuration can be provided with the system, a set of Functional Units is maintained. Each of these corresponds to a particular system function. The actual physical device involved is determined by a table in the FDOS SIO Driver. The assignment of Physical Units to Functional Units is made at system generation time, and can be (temporarily) changed by the use of the AS command. The codes and corresponding Functional Units recognized by the exec are as follows:

DP	Data output unit (SIO 1Ø3B device).
DI	Data input unit (SIO 1Ø1B device).
PL	Program list unit (SIO 1Ø2B device).
PK	Program keyboard unit (SIO 1Ø4B device).
SC	Scratch unit (i.e., pass 1 output of FORTRAN Compiler).
HL	High-speed list unit (lineprinter or disk/tape file).
BC	Batch command input unit.
BM	Batch message unit.
AP	Auxilary data output unit.
AI	Auxilary data input unit.
AH	Auxilary high-speed list unit.
AL	Auxilary program list unit.
AK	Auxilary program keyboard unit.
AS	Auxilary scratch unit.
CL	Executive's console list unit.
CK	Executive's console keyboard unit.

The FDOS I/O Driver supports sixteen other units. Users wishing to refer to these for use with their own application programs can use the codes SØ, S1, . . ., S15. Physical and Functional devices are kept track of by the FDOS I/O Driver. It maintains two tables; the Functional Device Table (FDT) and the Physical Device Table (PDT).

The Functional Device Table consists of 32 16-bit entries, which correspond to the unit numbers 4Ø thru 77B. Each entry has the following format:

Bits Ø5-ØØ	Physical Unit (or another Functional Device).
Bits 11-1Ø	Paging character specification; Ø Use Null 1 Use multiple Linefeeds 2 Use Extra Paging character (set during sys-gen) 3 Use Formfeeds
Bits 14-12	Index to table of seven line counts for page spacing. All devices have 66 lines per page, and do page spacing after 6Ø lines. Ø means no line counting. Bits 14-1Ø not Ø means binary requests (negative word count) are treated as form-control commands.

The Physical Device Table consists of 16 8-bit entries, which correspond to the unit numbers Ø thru 17B. The disk Logical Units are handled seperately (units 2ØB thru 37B). The eight bits are interpreted as follows:

Bit 6	Input capability.
Bit 5	Output capability.
Bit 4	Disk command capability.
Bit 3	Always zero.
Bit 2	Defines a non-file oriented device (viz, Deck Zero, High-speed Tape, Low-speed Tape, TTY, Lineprinter).

This arrangement corresponds to the contents of the Command Requirement Table, indexed by driver commands.

The FDOS I/O Driver provides a "sense" call for sensing the capabilities and identity of a Functional or Physical Device. The status words returned have the following format:

B Register

Bits 4-Ø	Physical Unit.

A Register

Bit 14	Device has input capability.
Bit 13	Device has output capability. (Bits 14-13 both zero if device doesn't exist).
Bit 12	Device accepts disk commands.
Bit 11	(Last) Functional Device does line counting. If a zero, device accepts binary output commands; if a Physical Unit is specified in the call, this bit is zero, i.e., all Physical Devices which do output have binary rather than list capability. Only <u>Functional Devices</u> can have page formatting (line counting).
Bit 1Ø	Set if device has file-oriented (logical) hardware.
Bits 6-4	Line counts (from 14-12 of FDT or command).
Bits 3-2	Paging Character (from 11-1Ø of FDT or command).

Bits 6-2 are taken from the noted FDT locations, except for three cases;

1. If the device has no list capability (bit 11 = Ø), then bits 6-2 are zero.
2. If the ultimate Physical Device is one which ignores functional line counts and page spacing characters (viz, Lineprinter, TTY, Deck Zero, Bucket), bits 6-4 are zero and bits 3-2 are 3 (Formfeed).
3. If the ultimate Physical Device is a disk logical unit, then bits 6-4 reflect the actual line counter, but bits 3-2 are 3 (Formfeed), since Formfeeds are always used for page spacing on disk logical units.

IV. SIO DRIVER PACKAGE DETAILS

The SIO Driver package contains a combined driver capable of supporting the FDS, the cassette system, console device, line printer, and high-speed paper tape equipment; a System Loader; and a Communication Region. The package also includes a configuration section to enable the user to adapt the package to a particular hardware arrangement and memory size.

IV.A. THE SYSTEM LOADER

The System Loader is used to load other programs under Executive control, or for automatically reloading the Executive after the execution of another program. This latter function is accomplished by means of a transfer to location X754ØB, X7542B, or X7544B (see Section V.E.3). The locations X7534B-X7677B must not have been changed by the other program. These 144B locations constitute the only portion of the system which must remain resident in memory for continuous automatic operation.

The function actually accomplished by the transfer to X754ØB, X7542B, or X7544B is the automatic loading of one of the fixed address files (SYS1, SYS2, and SYS3), thus bringing the system up from one of the three levels of restart (see Section V.B). If the SIO Driver package is to be used without the Executive, these functions must be modified, or appropriate programs must be placed in those files.

The loader also has available another entry, ELOAD, described below. This entry allows the specification of the sector from which to start loading. As in the case of the other System Loader entries, the address of the first location loaded is immediately stored in location 3B, and JMP 3B,I is executed upon completion of the load, unless another address is specified in STRTA (see below).

IV.B. THE COMMUNICATION REGION

The Communication Region is a set of locations referred to by other programs. They are gathered together into the area directly below the System Loader so that their actual octal addresses can be kept constant, despite any changes which may occur from time to time in the driver or loader programs. The following items make up the communication region:

LERR (X7537B). This cell has two uses. First, the System Loader uses it to indicate that there has been an error in loading the Executive or some other program. In this case, LERR is set to -1, and a transfer is made to X7544B to reload the system via a "cold start".

The second use for LERR is for interpass loading. LERR is set to the address of the program name in core (three words containing up to five valid ASCII characters, which are left-justified and zero-filled; this is preferably on the base page so that the Executive will not overlay it), and a transfer is made to X754ØB to load the Executive, which in turn loads the named program.

BTCHF (X7536B). This word specifies to the Executive whether control input is to come from the CK unit (on-line mode, BTCHF = Ø) or from the BC unit (batch mode, BTCHF = -1).

STRTA (X7535B). This word is used to pass a starting address to the loader. If this is set to zero, the starting address will be determined as explained above. Otherwise, the contents of STRTA will be used as the address to which to transfer after loading.

JSYST (X7534B). This word contains a jump to X754ØB.

PARAM (X7524B - X7533B). These locations are used for passing parameters between programs. The Executive Run command passes up to seven parameters according to the following convention:

PARAM: Contains the number of parameters passed.

PARAM+n: Contains the value of the n-th parameter.

The Executive's interpass loader does not modify the parameter area.

ELOAD (X7523B). This cell contains a jump to the generalized load entry point. A transfer to ELOAD with a sector address in the A register will cause one file to be loaded starting at that sector. Control will then pass to the loaded program as described above (see STRTA).

AFLG (X751ØB). This word is used by the SIO driver routines to indicate the type of binary data being processed. AFLG = 1Ø1B for absolute binary; for relocatable binary AFLG = Ø. This must be set by the calling program.

GSFLG (X7515B). This word is used by the Driver routines to control treatment of "group seperator" (Form Feeds and the Extra Paging Character). If GSFLG = Ø, then group seperators are not passed to the calling program; if GSFLG ≠ Ø, then group seperators are treated the same as other characters. GSFLG is normally set to zero by the Executive before executing a Run command.

P.FDT (X7521B). This cell contains the address of the functional device table. This is a table used by the driver in processing references to functional devices.

P.PDT (X752ØB). This cell contains the address of the physical device table. This table is used by the driver to store information about the availability and capabilities of the physical devices supported by the system.

P.DIO (X7522B). This cell contains the address of DIO, the "system" entry to the FDOS I/O driver (see Section IV.C.1).

P.CIO (X7512B). This word contains the address of the CIO (Custom SIO) entry to the driver. The program may provide itself with SIO style calls (A = count, B = buffer, JSB, return to following location) and implicit commands (see Section IV.C.2c).

P.XIO (X7517B). This cell contains the address of XIO, the "tailored" entry to the driver (see Section IV.C.2b).

NLOGU (X7513B). This word contains the negative of the number of Logical Units for FDOS (four per drive).

NBUFS (X7514B). This word contains the negative of the number of buffers allocated to the FDS driver. NBUFS is established at Sys-Gen time (see Section V.F.14).

Data is stored on disk in fixed, 128-word blocks (sectors); this is a basic hardware characteristic of the disk. From a software standpoint, data must be transmitted in record-oriented form; record length, however, is variable (usually approximately 36 or 60 words - see Section II.C). Therefore, the FDS Driver uses no relationship between records and sectors; rather, data are read from or written to disk in sector blocks, using core-memory buffers to parse out records or accumulate records. Operation of the sector buffering is transparent to the user; if a record starts in one sector and continues to another, the next sector is managed automatically.

The buffers are pooled and shared, and hueristically assigned and used. The greater the number of buffers available for disk usage, the less overlap in the usage; therefore, the number of disk accesses is decreased and thruput is increased. Since each buffer requires 129 words of core, a balance between the number of buffers and the amount of available memory must be determined for optimum operation; the minimum number of buffers is two, and the maximum is fourteen (see Section V.F.14).

It should be noted that the assignment hueristics allow data to accumulate in the buffers. It is the purpose of the ZS command of the Executive to clear out these buffers before switching disks so that sectors from the old disk, which are kept in core, are not interpreted as being from the new disk.

IV.C. FDOS DRIVER CALLS

1. The FDOS "system" driver entry is DIO. The calls to DIO have the following form:

```
              JSB P.DIO,I
              DEF TABLE    ADDRESS OF ARGUMENT TABLE
              (returns here with A & B set per IV.C.1d)
               .
               .
               .
       P.DIO  DEF X7522B,I POINTER IN COMM. REGION TO DIO
               .
               .
               .
       TABLE  ABS (command)SPECIFIES OPERATION AND DEVICE
       NUMB   DEC (count)  SAME AS SIO (A) SETTING FOR TRANSFERS
       BUFF   DEF (address)SAME AS SIO (B) SETTING FOR TRANSFERS
```

1a. The Command word specifies the operation in bits 9 thru 6; these are coded as follows:

CODE	OPERATION
ØØ	Skip to start of next file
Ø1	Rewind
Ø2	Read a record
Ø3	Write a record
Ø4	Write an End-of-File
Ø5	Name Hookup
Ø6	Verify

Ø7	(reserved)
1Ø	(reserved)
11	(reserved)
12	Skip a record
13	(not used)
14	(not used)
15	Sense
16	(not used)
17	(not used)

The Device is specified in bits 5 thru Ø (see Table II-1). Bits 14 thru 1Ø may be used to specify line counter/page character as in the Functional Device Table (FDT). Bit 15 must be zero.

1b. The Count word specifies the number of operations or the amount of data:

For skip operations, specifies the number of files or records to skip; use a positive count for ASCII data, and a negative count for binary data.

For Rewind, WEOF, and Sense operations, count is ignored.

For transfer operations, count specifies that length of data to be output or buffer available for input. If negative, it is minus the number of 16-bit words. If positive, it is the number of 8-bit ASCII characters.

If ASCII is specified (positive A on call), records are read up to the carriage return and line feed codes These codes are also written at the end of each ASCII record output. They are not transferred to the buffer on input, and they are not counted in determining the total number transferred on either input or output. If an odd number of characters is read, an extra space is supplied to fill out the last word.

If binary is specified (negative A on call), data words are written from or read into ascending core addresses with the high order half-word corresponding to the first character input or output for that word. If an odd number of characters is read in a binary read, an extra zero is supplied.

For Name Hookup, a code to specify the type of positioning; Ø = disk file name, -1 = a numeric positioning, such as a cassette file number.

1c. For transfer operations, the Address word specifies the address of the buffer. On Name Hookup, the Address specifies the address of the name (3 words) or the address of the number (1 word).

1d. On return from the DIO call, the contents of B contains the physical device number (see Table II-1), and the contents of A is set as follows:

(A) = 1 for successful skip, rewind, write, write EOF

(A) = number of words or characters for successful read

(A) = ∅ if file mark detected

ERRORS: (A) = -1 if not ready

(A) = -2 if write locked

(A) = -3 if end-of-tape while reading

(A) = -4 if error status while writing

(A) = -5 if error status while reading

(A) = -6 if end of tape while writing

(A) = -7 if format error on FDS (wrong name in file)

(A) = -8 if nonexistent device*

(A) = -9 if illegal device reference

(A) = -1∅ if nonexistent command

*Non-existent device, physical device 17B (A = -8, B = 17B), means FDT assignment loop.

NOTE: If an error occurs and bit 15 of B is set, a drive related error has occured (not related to a logical unit such as disk full, format error, or write protected) and bits 5-∅ are coded as for a logical unit.

2. Other FDOS Driver entries are SIO, XIO, and CIO. To use any of these entries, a JSB to it should be made with the following register contents:

(A) = count (see Section IV.C.1b)

(B) = address of buffer (see Section IV.C.1c)

Successful return is made to the location following the JSB, with the following register settings:

(A) = one for successful Skip, Rewind, Write or WEOF operations;

(A) = zero if file mark detected; or

(A) = number of words or characters transfered for read operation.

(B) = physical device number used for operation (see Table II-1).

When these entries are used, error halts occur, as opposed to status returns as in DIO. These error halts are:

HLT 61B	Device not ready.
HLT 62B	Device write locked.
HLT 63B	End of tape in read.
HLT 64B	Error during write.
HLT 65B	Error during read.
HLT 66B	End of tape during write.
HLT 67B	Disk format error.
HLT 7ØB	Non-existant device.
HLT 71B	Illegal device reference.
HLT 72B	Non-existant command.

When these halts occur, the registers are set as follows:

(A) = address of call

(B) = physical device used (see Table II-1). If bit 15 is set, then bits 2 and 3 define the drive used when accessing the disk directly (without using disk logical units); bits 2 and 3 = ØB define drive D, = 1B define E, = 2B define F, and = 3B define G.

Pressing RUN will cause the command to be re-executed. Multiple record or file skips will re-execute the entire number, even if one or more has already been skipped.

2a. Standard Hewlett-Packard software, which uses SIO driver calls, makes use of fixed locations in memory:

1Ø1B	P.RE	Address of SIO input entry
1Ø2B	P.LI	Address of SIO list entry
1Ø3B	P.BO	Address of SIO output entry
1Ø4B	P.KY	Address of SIO keyboard entry
1Ø5B	P.FW	First word address of available memory
1Ø6B	P.LW	Last word address of available memory
1Ø7B	P.MT	A flag used by HP MTS (not usable in FDOS)

The Dicom combined SIO driver package sets all of these (except P.FW, which is normally set by individual programs)as follows (see entry names above):

```
1Ø1B  P.RE   DEF SIO.I (uses DI)

1Ø2B  P.LI   DEF SIO.O (uses PL)

1Ø3B  P.BP   DEF SIO.P (uses DP)

1Ø4B  P.KY   DEF SIO.K (uses PK)

1Ø6B  P.LW   DEF (last word before driver package)

1Ø7B  P.MT   NOP (not usable - must be Ø)
```

The SIO calls have implicit commands associated with them; that is, read or write on functional units.

2b. The XIO calls allow explicit commands to be used, but are treated like SIO calls rather than DIO calls in regards to errors. The XIO calls are of the form:

```
            LDA (count)
            LDB (address)
            JSB P.XIO,I
            ABS (command)
            (return)
             .
             .
             .
     P.XIO  DEF X7517B,I POINTER IN COMM. REGION TO XIO
```

The command may be any valid command (see Section IV.C.1a).

2c. The CIO calls also allow explicit commands to be used, but are of a double subroutine level; the outer level uses implicit commands. Errors are treated in the SIO fashion as explained above. The CIO calls are of the form:

```
            LDA (count)
            LDB (address)
            JSB SIO.U
            (return)
             .
             .
             .
    SIO.U  NOP
           JSB P.CIO,I
           ABS (command)
            .
            .
            .
    P.CIO  DEF X7521B,I POINTER IN COMM. REGION TO CIO
```

Note that no return is needed from the SIO.U routine; if an error halt occurs, then the A Register will contain the address of the JSB SIO.U. However, SIO.U must be exactly three words long (entry JSB, and command). ABS can be any legal command; see Section IV.C.1a.

3. Utility commands exist within the combined driver. A command of minus one causes all the sector buffers to be cleared (see Section IV.B); this is what occurs when the Executive's ZS command is used. This clearing of buffers should be accomplished prior to removing a cartridge from a drive, thus removing holdover sectors from the buffers.

It should be noted that even though the address word (B in the SIO/XIO/CIO calls, or table entry three in DIO calls) is not used, it is evaluated, and therefore must not cause an indirect loop.

Other utility commands (negative value command word) exist in the combined driver; the user is refered to the listing of the driver for these.

IV.D. FDOS DRIVER OPERATIONS

The action and idiosyncracies of the SIO/XIO/DIO/CIO commands and devices are as follows:

1. Skip to start of file, octal Ø+UNIT; skips past the number of End-of-File marks specified by the count. On file-oriented devices (disk or cassette) this is done by moving over files, but on paper tape type devices the data must be read to determine the End-of-File marks; therefore, the count must specify the mode (positive for ASCII, negative for binary). The sign of the count has no effect on a file oriented device.

A count of zero results in no action being taken: If an error halt occurs, continueing will cause the entire number to be skipped again. If, on a paper tape type device, a multiple file skip is done and an intermediate End-of-File mark consists of over 20 nulls, then the first ten constitute the EOF and the subsequent group of ten nulls are ignored until a non-Null character is found. However, if the last file mark of the skip consists of over 20 nulls, the skip ends when ten are found, and a subsequent read (without manual intervention) will find ten more, resulting in an EOF (most programs ignore an initial EOF). A device must have input capabilities to do a skip; the Address parameter is ignored.

2. Rewind, octal 1ØØ+UNIT; on a file-oeiented device (disk or c cassette), positions to in front of file #1. On paper tape type devices, executes a HLT 6ØB (even if a DIO call) to allow manual intervention before continuing.

A device must have input capabilities to do a rewind. On disk, even if no files are in the LUD, there is no End-of-Tape indication (until some forward movement is given). The count and the address parameters are ignored.

3. Read a record, octal 2ØØ+UNIT; reads a record in the specified mode into the specified buffer of a specified length, one character at a time. The Address word may be indirect. The count must be positive to indicate the number of ASCII characters, or negative to indicate the number of binary words. Characters are intrepreted as they are read. In binary, only the first character, as the record length; in ASCII, all characters as to meaning (Nulls ignored, Rubout deletes line, Return ignored, Linefeed ends record, CNTRL/H deletes preceeding character of line, Extra paging character converts to Formfeed, Formfeeds ignored or passed according to GSFLG).

If the buffer address is zero, then the count is ignored and no data is transmitted to the user (this is a skip one record); otherwise, if the count is zero, no action takes place and A is returned as 1. Reading continues until a record containing some data.is found. If a record is larger than the specified buffer length, then the extra characters are not transmitted and input continues to the end of the record (to leave the device properly positioned for the next record). Finding a physical EOF (on file-oriented devices) or finding ten Nulls at the start of a read operation is interpreted as EOF (file oriented devices are always left past the physical EOF). If an odd number of characters is read in a record, the last word of the buffer is filled with a zero if binary, or a space if ASCII (although the space is not included in the count returned). In ASCII, if a Linefeed is not preceeded by a Return or a Linefeed, then a Return is echoed if appropriate for the device (i.e., on TTY, and Deck Zero). A device must have input capabilities to do a read.

4. Write a record, octal 3ØØ+UNIT; transfers the specified number of words or characters from the specified address in the specified mode, or does a format control to the device. In binary transfers, the count is the negative number of words, and those words are transfered with no processing, but some (physical device dependent) processing is done at the end of the record - paper tape type have four Nulls following, cassette has a physical end-of-record following which signifies logical end-of-record on binary input, disk has no processing (and records could theoretically be concatenated).

In ASCII transfers, the count is the positive number of characters, and these characters are transferred (except for nulls and except for the last character if it is a backarrow). A Return and Linefeed are output unless the last character of the record is a backarrow. If the Return and Linefeed are output and a line count is being kept (and the count goes to 6Ø) then the specified paging is done for the device and the line counter is reset.

In format control, a zero count always means output a Return and a Linefeed. If the device is a list-capable unit, then the negative count means formatting rather than binary. A minus one means top of form. All other negative values are ignored. Format control ignores the address parameter. A device must have output capability to do a write.

5. Write an End-of-File, octal 4ØØ+UNIT; on a file-oriented device, a physical End-of-File is written and, on disk, the unit is positioned to the begining of the next file. On the lineprinter, two pages are ejected. On paper tape type devices, if it is a list-capable device, a Formfeed, a Rubout, and a Linefeed are output followed by, or if a binary device, only, 22 nulls (to allow for over ten nulls and tear-off space on Low-speed Tape, or to allow time to tear off the page on TTYs), and, if High-speed Tape, 1ØØ nulls for leader/ trailer. A device must have output capabilities to do a WEOF.

6. Name Hookup, octal 5ØØ+UNIT; passes to disk logical unit a name. The count parameter must be zero (a flag) and the address

points to the name (up to five valid ASCII characters, left-justified and zero filled in three words). The logical unit [illegible] temporarily remembers the name, and, with the next access, proceeds with that file (randomly accessed via the Disk Directory) as if it had been in the LUD at that point. The name is not remembered after the first motion access to that logical unit, but the logical unit will be in the file until it leaves for some reason (Rewind, Skip, WEOF, or EOF read, etc.). If the logical unit leaves the logical unit at the EOF (rather than a Rewind) it will be positioned as it was before the Name Hookup, unless it was in a file, in which case it will be at the begining of the following file.

The Name Hookup command provides for a number to be passed (count = -1, address points to number) for count-selection type purposes in other type devices. A device must have disk command capabilities to do a Name Hookup.

7. Verify a record, octal 6ØØ+UNIT; a record is read into core (under the same conditions as read) and, character-by-character on the fly, if it differs from what was there the verify fails. A is returned as 1 if all characters were the same, and is zero if any characters differed. The new characters are in the buffer.

This command is not expected to be used with ASCII records containing Rubouts or CNTRL/H, as the buffer and the success/fail have already been affected by the deleted chatacter(s).

8. Skip record, octal 12ØØ+UNIT; the specified number of records are read (under the same conditions as read) in the specified mode, but no data is passed to the user.

9. Sense device, octal 15ØØ+UNIT; the status words returned as defined in Section III-H.

[illegible]

[illegible]

V. FDOS CONFIGURATION

The FDOS software is shipped in both a "standard" configuration and a "skeleton" configuration. Thus, the user can configure his hardware per the "standard" software configuration and get the FDOS "on-the-air" immediately; or the user can use the "skeleton" portion and tailor the software for the memory size, interface channel assignments, and functional unit assignments that match his specific requirements.

V.A. FDOS "STANDARD" CONFIGURATION

The "standard" software configuration shipped can be installed per the "Installation of Software" appendix of this manual. This software configuration is as follows:

Memory Size - 12K

Channel Assignments -
* 1ØB = FDS Data channel
* 11B = FDS Control Channel
* 12B = Console Device Interface channel (HP-interfaced teleprinter)
* 13B = Photoreader Interface channel
* 14B = Cassette System Interface channel
* The Line Printer (if available) is attached to Physical Device zero (cassette system "deck zero")

Functional Unit Assignments -
* Disk drive D as the "system library device" used by the Bootloader, System Loader, and the various system processors
* E1 as the SIO Input Unit (DI) used by the Executive default conditions (when applicable) and the various system processors
* E2 as the SIO Output Unit (DP) used by the Executive default conditions (when applicable) and the various system processors
* E3 as the High Speed List Unit (HL) used by the various system processors
* E4 as the scratch area (SC) used by the FORTRAN Compiler as temporary storage for the intermediate file (between pass one and two)
* All keyboard units (CK, PK, and AK) assigned to the HP-interfaced Teletype (TY)
* All low-speed list units (CL, PL, AL, and BM) assigned to the HP-interfaced Teletype (TY)

* Auxiliary Input (AI) assigned to the photoreader
* Auxiliary Output (AP) assigned to cassette deck one
* Auxiliary Scratch (AS) assigned to cassette deck two
* Auxiliary High-speed List assigned to the cassette system "deck zero" (lineprinter)
* Batch Command assigned to D2
* D4 as the library device for the Relocatable Libary used by the Absolute BCS file

V.B. FDOS INITIALIZATION AND LOADERS

The FDOS software makes available a number of different levels of restarts and initializations in order to accomodate the varying needs of users and software packages. The FDOS software is supplied on a "skeleton disk" which contains seven files (designed specifically to facilitate startup procedures) plus the other FDOS program files. The seven specific files are:

* The unconfigured Bootloader
* The configured System Loader
* The "Skeleton" file (SYSØ)
* The "cold start" file (SYS1)
* The "warm start" file (SYS2)
* The "Executive" file (SYS3)
* The "System Map" file (SYSM)

1. The unconfigured Bootloader is designed to be loaded by the Keyed-In Loader (see Section V.C). Once the unconfigured Bootloader has been loaded, it can be configured for channel assignments and memory size, and the configured Bootloader moved to the protected area of memory (see Section V.D).

2. The "Skeleton" file (SYSØ) contains the unconfigured System Loader, unconfigured SIO Driver package, unconfigured Executive, and the System Generator. Special precautions are taken in the FDOS software to protect this file (see Section VI.F.2); thus, it is always available to the user for use in building a new FDOS software configuration.

3. The configured System Loader resides in sector zero of the disk, and is normally loaded via the configured Bootloader. The System Loader is the "core resident" portion of FDOS. On the "skeleton disk" shipped from the factory, the System Loader is configured per Section V.A; when the configuration (see Section V.F) is performed, a new System Loader is written in sector zero.

4. The "System Map" file (SYSM) contains a listing of all peripheral device channel assignments, all functional unit assignments, and other system configuration details. This file can be printed on the Teletype with the following command string (see Section VI.F.2):

```
SE;TQ D1,SYSM;LI D1,CL
```

   Section V.G shows a printout of SYSM for the "standard" software configuration. The Functional Unit assignment portion of the System Map can be printed directly with the List All units (LA) command.

5. The three remaining files (SYS1, SYS2, and SYS3) contain the configured (per Section V.A) FDOS software system; when the system generation is performed, a new software system is written into these files. Special precautions (see Section VI.F.2) are also taken here, so the System Generator is the only package that can normally write in these files.

6. Track zero of all disks is reserved for system control information (directories, etc.); the various "system" files start and/or reside on track zero. This track is allocated as follows:

Sector Ø	Configured System Loader
Sector 1	Disk Directory
Sector 3	Unconfigured Bootloader
Sector 4	Logical Unit Directory, unit 4
Sector 5	Logical Unit Directory, unit 1
Sector 6	Logical Unit Directory, unit 2
Sector 7	Logical Unit Directory, unit 3
Sector 13	Starting Sector of the "System Map" file (SYSM)
Sector 14	Starting sector of the "Skeleton" file (SYSØ)
Sector 15	Starting sector of the "Cold Start" file (SYS1)
Sector 16	Starting sector of the "Warm Start" file (SYS2)
Sector 17	Starting sector of the "Executive" file (SYS3)

   All other sectors of track zero (sectors 2,10,11, and 12) are reserved for expansion. The remaining disk area (tracks one through 64) is used by the remaining FDOS programs, or is available to the user.

V.C. FDOS KEYED-IN LOADER

The FDOS Keyed-In Loader is a program used to load the unconfigured Bootloader from sector three of disk D. The Keyed-In Loader is designed to be as short as possible, since it is intended to be entered from the front panel of the computer. Since the unconfigured Bootloader is in a special format, it must be loaded with the Keyed-In Loader; the Keyed-In Loader cannot be used for loading any other file from disk.

To commence the configuration process, enter the Keyed-In Loader shown below (refer to "A Pocket Guide to the 2100 Computer" or to "A Pocket Guide to Hewlett-Packard Computers"); then configure the remainder of FDOS per Sections V.D and V.E.

```
* FDOS KEYED-IN LOADER
*
* THIS PROGRAM LOADS THE UNCONFIGURED BOOTLOADER INTO MEMORY
* (STARTING AT LOCATION Ø76ØØB) AND HANGS.  WHEN THE PROGRAM
* IS STARTED, THE DISK WILL CLICK; THE UNCONFIGURED BOOT-
* LOADER IS IN CORE 'INSTANTLY'.  PRESS HALT.  NOW CONFIGURE
* THE BOOTLOADER PER SECTION V.D.
*
  Ø2ØØØ               ORG 2ØØØB
*
  Ø2ØØØ 1Ø31DC  START CLF DAT        MAKE SURE FLAG IS CLEAR
  Ø2ØØ1 Ø62Ø14        LDA READ       READ COMMAND
  Ø2ØØ2 Ø66Ø15        LDB CORE       ADDR TO LOAD INTO
  Ø2ØØ3 1Ø26CC        OTA CMD        SEND
  Ø2ØØ4 1Ø37CC        STC CMD,C        THE COMMAND
  Ø2ØØ5 1Ø23DC  LOOP  SFS DAT        WAIT FOR
  Ø2ØØ6 Ø26ØØ5        JMP *-1          DATA FLAG
  Ø2ØØ7 1Ø25DC        LIA DAT        INPUT A WORD
  Ø2Ø1Ø 1Ø37DC        STC DAT,C      ACKNOWLEDGE
  Ø2Ø11 17ØØØ1        STA B,I        STORE WORD
  Ø2Ø12 ØØ6ØØ4        INB            BUMP STORAGE POINTER
  Ø2Ø13 Ø26ØØ5        JMP LOOP       GET ANOTHER WORD
  Ø2Ø14 Ø2ØØØ3  READ  OCT 2ØØØ3      DISK D, SECTOR THREE
  Ø2Ø15 ØØ76ØØ  CORE  OCT 76ØØ
*
*
* DC IS THE DATA CHANNEL (NORMALLY 1ØB)
* CC IS THE CONTROL CHANNEL
*
```

V.D. FDOS BOOTLOADER CONFIGURATION

The FDOS Bootloader is designed to load the configured System Loader from sector zero of disk D; it has a second entry that can be used to load any absolute file from disk D whose initial sector address is contained in the Switch Register. The Bootloader is supplied on the "skeleton disk" in sector three; it is not configured.

1. To configure the Bootloader, enter the Keyed-In Loader (see Section V.C) and place the "skeleton disk" in drive D. Preset the computer, then start the Keyed-In Loader at Ø2ØØØB to load the unconfigured Bootloader and its configurator. Halt the computer, then restart it at Ø76ØØB; the configurator program will come to a series of halts:

HLT	74B	Enter the disk data channel in the Switch Register (normally 1ØB) and press RUN.
HLT	75B	Enter the disk control channel in the Switch Register (normally the lowest priority channel, although it can be of higher priority than any asynchronous device) and press RUN.
HLT	76B	Enter the memory size mask (X77ØØB) in the Switch Register, enable the loader area, and press RUN. The Bootloader will be configured and moved to X77ØØB, i.e., the protected area of memory. Use X = 1 for 8K, 2 for 12K, 3 for 16K, etc.
HLT	77B	Configuration complete; protect the loader area.

2. The configured Bootloader has two entry points:

X77ØØB	Load a file from disk D, sector zero; this is normally the configured System Loader.
X77Ø1B	Load a file from disk D whose initial sector address is contained in the Switch Register. (If using a 2114, the LOADER ENABLE switch on the inside of the front panel must be in the ON position to start at location 177Ø1B.)

After execution at either of these entry points, the Bootloader will halt with the P Register equal to 2B; if the configured System Loader was the file loaded, it can be executed by just pressing RUN. This is possible, since the System Loader (as it is being loaded by the Bootloader) sets location 2B to a jump indirect through 3B, and sets location 3B to X7544B (the "cold start" entry point); upon successful loading, the Bootloader halts with the P Register set to 2B. Since most absolute program files also set locations 2B and 3B in this manner, this feature can be used to load and execute those files.

V.E. FDOS SYSTEM LOADER

The System Loader is intended to remain in memory at all times. When not resident in memory, it can be loaded by the protected Bootloader. The System Loader has three entry points:

X754ØB	Standard Exec reload; load the file starting at sector 17B. This entry is used by the ASMB, EDIT, CROS, etc., programs. It leaves the SIO area, Communications Region, Logical Unit Parameter Tables, and buffer status intact.
X7542B	Load the file starting at sector 16B, the "Warm start"; the SIO and Communications Region are reloaded, and a transfer to X754ØB is made to reload the Exec. The Logical Unit Parameter Tables and buffer status tables are left intact. This is intended to restore the original SIO configuration after a temporary reconfiguration.
X7544B	Load the file starting at sector 15B, the "Cold start"; this is intended to be used when the contents of memory can no longer be counted on, such as after a program which does not preserve the SIO area has been run. This start must be used if the system ever "crashes". This start initializes the Logical Unit Parameter Tables and the buffer status tables, and transfers to X7542B for a "warm start".

On the "skeleton disk" shipped from the factory, the System Loader is configured per Section V.A.; when the configuration (see Section V.F) is performed, a new System Loader is written in sector zero.

V.F. FDOS "SKELETON" CONFIGURATION

The "skeleton" file (SYSØ) contains an interactive System Generator section, the unconfigured SIO Driver, the unconfigured System Loader, and the unconfigured Executive. When SYSØ has been loaded and executed, the user can tailor FDOS to a particular hardware configuration, memory size, unit assignment, etc. This is accomplished by typing responses to questions asked by the System Generator; after the configuration parameters have been entered and verified, the System Generator (optionally) writes a configured FDOS software system into SYS1, SYS2, SYS3, and SYSM of disk D. It should be noted that this System Generator can be used to prepare an FDOS disk for a computer system with a different hardware configuration from the one it is being run on; the only restrictions are that the current system has at least as much memory as the target system, that the disk data and control channels are the same on both systems, and that the console device has the same channel number on both systems.

Following is the configuration procedure:
1. Bootload SYSØ using the X77Ø1 entry of the Bootloader with the Switch Register set to 14B. To execute the file after such a load, press RUN; the computer will execute a HLT 5ØB.

2. Set the Switch Register to the channel number of the console device (functional units CK and CL); if the console device is interfaced thru the cassette system deck zero, set bit 15 also. Press RUN.

3. The program types

```
        SYSTEM ID:
```

requesting up to sixty characters of ID information to be used to label the resultant configured disk. This ID information will be written in the System Map (SYSM) file if "write system to disk" is specified later on in this system generation procedure.

> NOTE: Responses to questions are terminated by Carriage Return-Line Feed. If an error is made in answering the questions in Step 4 and beyond, the System Generator can be restarted at Step 3 by typing "ER". The valid responses are "NO" to indicate that the device is not available, a two-digit octal number defining the device's channel number, or responses defined in the descriptions below.

4. If bit 15 was not set in Step 2 above, the program types

        TELETYPE CHANNEL: CC

where "CC" is the two digit octal number that was entered in the Switch Register in Step 2 to define the HP-interfaced Teletype channel. If an HP-interfaced Teletype was not defined in Step 2 (bit 15 set), the program types

        TELETYPE CHANNEL?

requesting the HP-interfaced Teletype channel number. Reply appropriately.

5. The program types

        USES PAGING OPTION?

requesting the option to be used to format printed pages on this Functional Unit (CL). Respond by typing "L" for Line-Feed, "F" for Form Feed, "N" for no page spacing, or "x" for the Extra Paging Character (see Step 18).

6. The program types

        HAS LOW SPEED TAPE?

thus asking if the HP-interfaced Teletype is to be used for paper tape input or output. Reply by typing "YES" or "NO". If "YES" is typed, the program types

        PUNCHING AND PRINTING SEPARATE?

thus asking if the HP-interfaced Teletype is an ASR33 or ASR35. Type "NO" if ASR33; type "YES" if ASR35. If an ASR33 is defined, then under normal FDOS operation the computer will halt (HLT 56) before outputing to the ASR33 paper tape punch to allow the operator to turn on the punch; when punching is complete, the computer will halt (HLT 55) to allow the operator to turn the punch off.

7. The program types

        HIGH SPEED PUNCH CHANNEL?

requesting the high-speed paper tape punch channel number. Reply appropriately.

8. The program types

```
        HIGH SPEED READER CHANNEL?
```

requesting the channel number of the photoreader. Reply appropriately.

9. The program types

```
        LINE PRINTER CHANNEL?
```

requesting the line printer channel number; reply appropriately. The SIO Driver supports the HP 2767 (Data Products) or 2607 (Tally) printers; for the HP 2778 (CDC) or other non-compatible printers, the driver must be slightly modified.

10. If bit 15 was set in Step 2, the program types

```
        CMTS CHANNEL: CC
        DECK ZERO EXISTS
        HAS INPUT CAPABILITY
        USES PAGING OPTION?
```

where CC is the channel number defined in Step 2. The response to the paging option question is the same as in Step 5.

11. If bit 15 was not set in Step 2, the program types

```
        CMTS CHANNEL?
```

requesting the cassette system channel number. If "NO" is typed, the program proceeds to Step 12. If "YES", the program types

```
        DECK ZERO EXSISTS?
```

asking if a device (TTY, CRT, line printer, etc.) is interfaced to the computer via deck zero. If "YES" is typed, the program types

```
        HAS INPUT CAPABILITY?
```

asking if the device has a keyboard on it. Reply appropriately. The program types

```
        USES PAGING OPTION?
```

The response to the paging option question is the same as in Step 5.

12. The program types

```
          FLOPPY DISK DATA CHANNEL?
```

requesting the FDS data card channel number. Since this must be defined, a "NO" response causes the question to be re-asked.

13. The program types

```
          FLOPPY DISK CONTROL CHANNEL?
```

requesting the FDS control card channel number. A "NO" causes steps 12 and 13 to be repeated.

14. The program types

```
          NUMBER OF BUFFERS?
```

Reply appropriately (see Section IV.B). For optimum operation in an 8K system, three buffers are recommended; in 12K or larger systems, four buffers are recommended.

15. The program types

```
          NUMBER OF DRIVES?
```

Reply appropriately (one to four), depending on how many drives are implemented in the FDS.

16. The program types

```
          DISPLAY DISK COMMANDS?
```

inquiring as to whether disk commands are to be displayed in the Switch Register during FDOS SIO operation; reply appropriately. Commands are not displayed during BCS operation.

17. The program types

```
            FUNCTIONAL UNIT ASSIGNMENTS:
          ?
```

At this point the operator may change the physical unit vs. functional unit assignment for a particular configuration of hardware. Until some experience has been gained, it is recommended that the user respond with only "/E", to cause the standard assignments that have been assembled into sys-gen to be used. To change the Functional vs. Physical Unit assignments, use the format FU,PU,LC,PC (see Section II.B). Typing "A?" will list the entire FDT; "/E" terminates and moves on to the next step. Restarting without reloading ("ER" input) does not restore the FDT.

18. The program types

```
        EXTRA PAGING CHARACTER?
```

Unless the system has a list device that is capable of responding to a page-spacing character other than Linefeed or Formfeed (such as Vertical Tab), it is sufficient to respond "NO". Otherwise, respond with a two-digit octal number representing the ASCII character desired.

19. The program types

```
        CORE SIZE (IN K)?
```

Reply appropriately.

20. The program types

```
        WRITE SYSTEM TO DISK?
```

Reply "YES" if it is desired to write this newly configured system to disk (in SYS1, SYS2, SYS3, & SYSM) later on in this program.

21. If "YES" was typed in Step 20, the program types

```
        FORMAT DISK?
```

inquiring as to whether a Format Disk command to drive D is desired. Type "YES" if there is a virgin disk in drive D.

22. The program types

```
        CONFIGURATION MAP?
```

Reply "YES" if it is desired to have one printed on the console device at this time; if "YES,unit" is typed, the map will be written to the specified unit.

23. The program types

```
        OKAY AS IS?
```

If the configuration is as desired, type "YES". At this point the unused portion of DIO (drivers for non-exsistent peripherals) is released and the system is configured in core. If the program was directed to write the system in step 20, SYS1, SYS2, SYS3, SYSM, and the configured System Loader are written onto disk D (see Section V.B), all of memory below X7534B is set to zero, and a "cold start" is made via a transfer to X7544B; if the program was not directed to write the system, a transfer is made to the Executive. If the configuration is not as desired, type "NO".

24. The program types

    START OVER?

If "YES" is typed, the program goes back to the "SYSTEM ID" question of step 3; if "NO" is typed, the program goes back to step 23.

25. The following error halts are applicable to the System Generator:

HLT 31B	The value stored at the address displayed in the Switch Register did not hold (either non-existant memory, faulty memory, unattached or unplugged memory, etc.). Correct condition and press RUN, or reload SYSØ and restart the System Generator.
HLT 32B	The process of moving the system has caused an over-lap of sections of relocatable code. This is a program safety check to insure the resultant system will fit in the available memory; it is fatal and irrecoverable. Continuation will repeat HLT 32B. Reload SYSØ and restart the System Generator.
HLT 47B	The processing of devices has caused an impossible situation. Press RUN to restart the configurator. Theoretically, this can not happen!

26. It is not necessary that the configurator write out the new system on the "skeleton" disk shipped from the factory; in fact, it is best if it does NOT, since this disk should always remain write-protected and used as the "master". The "skeleton" disk could have been replaced by any (virgin) disk after SYSØ was loaded from it. However, if a freshly formatted disk is used to write the just-configured system onto, SYSØ (sector 14B) will be an empty file, but sector 14B will be preserved for the

initial sector of SYSØ. This means that the "skeleton" file (SYSØ) does not get written during the configuration process; however, the unconfigured Bootloader does get written into sector three. The SYSØ file could be copied from the "master" disk after the configuration process if it were desired to have it on the "working" disks. The copying of SYSØ (plus all other files on the "master" disk) can be accomplished easily with the CODSK batch file; see the INSTALLATION OF SOFTWARE Appendix of this manual.

V.G. SAMPLE SYSTEM GENERATION PRINTOUT

```
SYSTEM ID:
SAMPLE SYSTEM GENERATION PRINTOUT

TELETYPE CHANNEL: 12
  USES PAGING OPTION? N
 HAS LOW SPEED TAPE? YES
  PUNCHING AND PRINTING SEPARATE? NO

HIGH SPEED PUNCH CHANNEL? NO

HIGH SPEED READER CHANNEL? 13

LINE PRINTER CHANNEL? NO

CMTS CHANNEL? 14
 DECK ZERO EXISTS? YES
  HAS INPUT CAPABILITY? NO
  USES PAGING OPTION? N

FLOPPY DISK DATA CHANNEL? 10

FLOPPY DISK CONTROL CHANNEL? 11
 NUMBER OF BUFFERS? 4
 NUMBER OF DRIVES? 2
  DISPLAY DISK COMMANDS? YES

<<FUNCTIONAL UNIT ASSIGNMENTS: >>
 ? /E

EXTRA PAGING CHARACTER? NO

CORE SIZE (IN K)? 12

WRITE SYSTEM TO DISK? YES
FORMAT DISK? YES

CONFIGURATION MAP? YES


SYSTEM ID:  SAMPLE SYSTEM GENERATION PRINTOUT

CMTS CHANNEL: 14
 DECK ZERO EXISTS
  USES PAGING OPTION: N
HIGH SPEED READER CHANNEL: 13
TELETYPE CHANNEL: 12
  USES PAGING OPTION: N
 HAS LOW SPEED TAPE

FLOPPY DISK DATA CHANNEL: 10
FLOPPY DISK CONTROL CHANNEL: 11
 NUMBER OF BUFFERS: 4
 NUMBER OF DRIVES: 2
  DISPLAY DISK COMMANDS
```

SAMPLE SYSTEM GENERATION PRINTOUT (Continued)

```
<<FUNCTIONAL UNIT ASSIGNMENTS: >>
DI   E1,B
PL   TELETYPE,Ø,L
DP   E2,B
PK   TELETYPE,Ø,L
HL   E3,Ø,F
SC   E4,B
CK   TELETYPE,Ø,L
CL   TELETYPE,Ø,L
AI   HS TAPE,B
AL   TELETYPE,Ø,L
AP   1,B
AK   TELETYPE,Ø,L
AH   DECK ZERO,2,F
AS   2,B
BC   D2,B
BM   TELETYPE,Ø,L
S2Ø  16,B
S21  16,B
S22  16,B
S23  16,B
S24  16,B
S25  16,B
S26  16,B
S27  16,B
S3Ø  16,B
S31  16,B
S32  16,B
S33  16,B
S34  16,B
S35  16,B
S36  16,B
S37  16,B


CORE SIZE (IN K): 12

FORMAT DISK
WRITE SYSTEM TO DISK



OKAY AS IS? YES
*
```

```
*DA D
.FREE.    1321  ** ADDRESS OF FIRST FREE SECTOR
SYS0        14  ** UNCONFIGURED BOOTLOADER, SIO PACKAGE, AND EXECUTIVE
SYS1        15  ** "COLD START" FILE
SYS2        16  ** "WARM START" FILE
SYS3        17  ** STANDARD EXECUTIVE RELOAD
SYSM        13  ** SYSTEM MAP
SRCE       161  ** DEMO PROGRAM SOURCE (ASSEMBLY LANGUAGE)
FSRC       205  ** DEMO PROGRAM SOURCE (FORTRAN)
LIBN       215  ** NON-EAU MATH LIBRARY
LIBE       342  ** EAU MATH LIBRARY
LIBF       462  ** FLOATING POINT MATH LIBRARY
CODSK      573  ** BATCH FILE FOR COPYING MASTER DISK
EDIT       603  ** EDITOR
ASMBN      626  ** NON-EAU ASSEMBLER
CROS       666  ** CROSS-REFERENCE GENERATOR
FORT       671  ** FORTRAN PASS ONE
FOR2N      762  ** NON-EAU FORTRAN PASS TWO
DUP       1002  ** DISK UTILITY PACKAGE
DEBUG     1012  ** SIO ENVIRONMENT DEBUGGER
EXER      1005  ** FDS DIAGNOSTIC
ABCS      1035  ** CONFIGURED BCS FILE
DPCS      1075  ** DISK PREPARE CONTROL SYSTEM
D.36      1111  ** FDS BCS DRIVER
D.36X     1135  ** FDS BCS DRIVER EXTERNALS
D.00      1137  ** TTY BCS DRIVER
D.35      1152  ** CMTS BCS DRIVER
.IOC.     1151  ** INPUT/OUTPUT CONTROL
RLOA      1157  ** RELOCATING LOADER
SD36X     1210  ** SOURCE OF FDS BCS DRIVER EXTERNALS
ASMBE     1201  ** EAU ASSEMBLER
ASMBF     1241  ** FLOATING POINT ASSEMBLER
FOR2E     1301  ** EAU/FLOATING POINT FORTRAN PASS TWO

*LD D1;LD D2;LD D3;LD D4
D1:
SRCE
FSRC
SD36X

D2:
CODSK

D3:
D.36
D.36X
D.00
D.35
.IOC.
RLOA

D4:
LIBN
LIBE
LIBF
```

VI. FDOS EXECUTIVE COMMANDS

A summary of the Executive commands appears in an appendix of this manual. When the Executive is ready for a command, it types "*" on the console device, rings the console device bell, and waits for a command line to be entered. A command line consists of one or more commands, each, except the last, terminated by a semicolon. A command consists of a two-character command code, optionally followed by a space and a variable number of parameters, each, except the last, followed by a comma. If a given command can accept parameters but not all are supplied, the Executive supplies "default" values for the missing ones.

The parameters for a given command must be supplied in a fixed order. If it is desired to omit (i.e., use the default value for) a given parameter but specify some or all of the following ones, simply type the comma which would normally follow it, but with no non-blank character between it and the comma following the previous parameter, if any, or the blank following the two-character command code. For example,

```
CO ,,2
```

requests the Executive to perform a copy function using default values for the first, second, and fourth parameters and the value "2" for the third parameter (in this example, two files of data are copied from DI to DP in the ASCII mode).

Several commands may be typed on one line (or placed in one record) by separating them with semicolons. For example, if it were desired to skip two files on logical unit El, copy the third file (in absolute binary format) from El onto cassette deck two, write an end-of-file on deck two, and then rewind deck two, the following command sequence would be used:

```
SK El,2;CO El,2,,BA;WE 2;RE 2
```

An important feature of FDOS is the handling of the Batch mode. Operation in the Batch mode of the Executive differs from normal on-line operation in that command lines are taken from the Batch Command unit (BC), and messages, including the actual command strings, are sent to the Batch Message unit (BM). Batch command lines can be intermixed with data input if BC is set to the same unit as DI (see example under Write command) and the CODSK batch file in the INSTALLATION OF SOFTWARE Appendix).

The Batch mode is entered via the Batch command (BA). The Executive remains in the Batch mode until one of the following occurs:

* A Batch Exit command is received from BC.
* An EOF is detected on BC.
* An I/O error is detected.

In addition to the standard HP feature of RUBOUT deleting (via the SIO driver) the current ASCII input line, FDOS has the following convenience features:

* Character delete; CTRL/H (hold CTRL key depressed, then strike H) deletes the previous character and echoes it back.
* Carriage Return echo; if a LINE FEED is typed to terminate a line, a RETURN is automatically echoed if one was not typed prior to the LINE FEED.

The Executive commands include the following general types.

VI.A. UNIT MANIPULATION COMMANDS

The unit manipulation commands facilitate file positioning; these include Rewind, Skip, and Write End-of-File. If an illegal command is attempted on a unit (e.g., Writing an End-of-File on the photoreader), the command is ignored and "IMPROPER REQUEST (unit)" is printed on the console device.

VI.B. DATA UTILITY COMMANDS

These commands facilitate the transfer of data to and from the various units supported by FDOS. The data utility commands include Assign, Reserve File, Copy, Verify, Dump, List, and Write.

VI.C. PROGRAM UTILITY COMMANDS

These commands deal with the handling of data in the standard (absolute) binary format recognized by the System Loader. They include the Batch, Go To, If Disk, Save Start Address, [illegible]ands. Save, and Run commands.

VI.D. DISK SYSTEM COMMANDS

These commands are concerned with the Disk Directories, the Logical Unit Directories, and disk formats. They include Format Disk, Disk Directory, Logical Unit Directory, Rename, Delete, Queue, Temporary Queue, Restore, and Zero System commands. These disk system commands have no meaning for other peripherals supported by FDOS.

VI.E. DESCRIPTION OF FDOS COMMANDS

The following pages give a description of each of the FDOS commands.

VI.E.1 ADD TO LOGICAL UNIT DIRECTORY

Purpose: Add files to the end of a logical unit directory.

Format:

```
AQ DUNT,NAME1,NAME2,...
DUNT,NAME1,etc., are as for QU.
```

Comments: Same as QU command, except that names are appended to the end of the current LUD.

VI.E.2 ASSIGN

Purpose: Assign a physical unit to a functional unit.

```
Format:  AS FUNIT,PUNIT,LCTR,PGCHR
 where:  FUNIT = a functional or physical unit
         PUNIT = a physical unit
         LCTR  = index to the system's table of line counters
         PGCHR = N  None        F Formfeed
                 L  Linefeed    X Special character set during Sys-Gen.
```

Comments: FUNIT is set to point to PUNIT.

If PUNIT, LCTR and PGCHR are not specified, they are not changed. If "AS FUNIT,?" is typed, the assignment of the unit is listed. LCTR and PGCHR may be replaced with a "B" to specify that the unit is a binary unit (not capable of page formatting).

VI.E.3 BATCH

Purpose: Enter Batch mode

Format: BA NAME
NAME = name of file containing the Batch commands.

Comments: A "batch" flag is set in the loader area (X7536B). Subsequently, all command lines are input from BC, rather than CK as in normal Executive operations. A return to the normal "on-line" operation of the Executive occurs when an EOF is detected on BC, when a BX command is received from BC, or if an I/O error is detected.

During Batch operation, Executive console messages (such as "EOF" reports during a copy) and echoing of Batch commands use BM for output, rather than CL as in normal operation.

If NAME is not specified, the next file on BC is used. NAME can only be used if BC is a disk logical unit.

VI.E.4 BATCH EXIT

<u>Purpose</u>: Return from Batch mode to the on-line (interactive) mode of Executive command input.

<u>Format</u>: BX NAME
NAME = File name to be used for next BA command.

<u>Comments</u>: A TQ of NAME to BC is done. A subsequent QU,RE, etc. of BC will override this TQ. If NAME is not specified, no action is taken with respect to BC.

VI.E.5 COMMENTS

Purpose: Allow comments to appear on Batch or on-line message units.

Format: ** Text

Comments: All characters after ** are ignored, up to the next Linefeed.

N.B.: A space must follow the **.

VI.E.6 COPY

Purpose: To duplicate one or more files

Format:
```
CO SRCE,DEST,#,MODE
Where SRCE = a physical or functional unit
      DEST = a physical or functional unit
      #    = number of files to copy
      MODE = A(ASCII), BA(absolute binary), or B(relocatable
             binary)
```

Comments: The remainder of the current file and the #-1 subsequent files of SRCE are copied onto DEST, each followed by a file mark.

If Ø is specified for #, 1 is used, but no file mark is written onto DEST. This feature, in conjunction with the Save command, is very useful in "patching" absolute binary programs. For example, suppose it were desired to change the contents of location 1Ø1B (the SIO Input pointer) whenever the Assembler (ASMB) is run. Then the following procedure could be used:

1. Load the system (by starting at X7544B). Halt the computer and change 1Ø1B to the desired value. Restart at the Executive restart address ((X-1)4ØØØB).

2. Type
```
QU D4,ASMB;QU D3,ASMB
CO D4,D3,Ø,BA;SA D3,1Ø1B;WE D3
RE D4;RE D3;VE D4,D3,Ø,BA;DU D3,,,,BA
```

3. If the response on the console device to the last line above is

```
EOF D4
ØØØ4ØØ  ØØ1Ø1  (new contents)  (checksum__)
EOF D3
```

then the operation has been performed successfully.

Although this example would accomplish the desired re-assignment of the input device (DI), a more convenient way would be via the Assign command.

If SRCE is not specified, DI is used.

If DEST is not specified, DP is used.

If # is not specified, one is used.

If MODE is not specified, ASCII is used.

VI.E.7 DISK DIRECTORY

<u>Purpose</u>: To produce a list on a specified unit the names of programs residing on the disk in the specified drive.

<u>Format</u>:
```
DI DISK,DEST
Where DISK = D,E,F, or G (floppy disk drive)
      DEST = a physical or functional unit
```

<u>Comments</u>: Each disk has a disk directory recorded in sector 1 of track 0. This is different from the logical unit directories in that:

(1) It contains only names of files actually recorded on the disk.

(2) It contains the starting sector address of each file.

(3) It contains the starting sector address of the chain of free disk blocks.

(4) The order of files listed in it is of no significance.

(See comments under LD command)

If DEST is not specified, CL is used.

VI.E.8 DISK DIRECTORY WITH ADDRESSES

Purpose: To produce a list on a specified unit the names of programs and their initial sector address.

Format: DA DISK,DEST
Where DISK = D,E,F, or G (floppy disk drive)
DEST = a physical or functional unit

Comments: This command is similar to the Disk Directory command (DI); in addition to the name of the program, the initial sector address of that program is printed. In addition, the symbol ".FREE." is printed, followed by the address of the first available (but not necessarily lowest numerical) sector of the free storage list.

If DEST is not specified, CL is used.

VI.E.9 DELETE

Purpose: To remove a file from the specified disk and free the sectors comprising it for reuse.

Format:
```
DL DISK,NAME
Where DISK = D,E,F,or G (floppy disk drive)
      NAME = name of file to be deleted
```

Comments: The file name is removed from the directory, and the chain of sectors comprising it are added to the beginning of the free storage list, i.e., the disk directory's free storage pointer is set to the first sector of the deleted file, and the last sector of the deleted file, which previously contained an end-of-file indication, is set to point to what was previously the first sector of the free storage list.

If the file deleted is one of the Fixed Address files (SYSØ, SYS1, SYS2,SYS3 or SYSM) then the first sector (14B., 15B, 16B, 17B or 13B) is not added to the free storage list, but the remaining sectors of that file are added to the free storage list.

VI.E.10 DUMP DISK

<u>Purpose</u>: Produce a listing on a specified unit of the contents of a specified range of disk sectors.

<u>Format</u>:
```
DD DEST,DISK,LO,HI
Where DISK is a floppy disk drive
      DEST is a physical or functional unit
      LO and HI are the first and last (octal) disk
      addresses of the block of sectors to be dumped.
```

<u>Comments</u>: The dump is printed in the following format:

```
(sector address)

    Ø:    word    word     . . .    word   (8 words per line)
   1Ø:    word    word     . . .    word
                   .
                   .
                   .
  17Ø:    word    word     . . .    word
(sector address)
    Ø:    word    word     . . .    word
                   .
                   .
                   .
```

If DEST is not specified, CL is used.
If LO is not specified, 2 is used.
If HI is not specified, LO is used.
DISK must be specified.

VI.E.11 DUMP MEMORY

Purpose: Produce a listing on a specified unit of the contents of a specified range of memory locations.

Format: DM DEST,LO,HI

Where DEST is a physical or functional unit
LO and HI are the first and last (octal) addresses of the block of memory to be dumped.

Comments: The dump is printed in the following format:

```
address:    word    word    ...    word    (8 words per line)
address:    word    word    ...    word
                     .
                     .
                     .
```

If DEST is not specified, CL is used.
If LO is not specified, 2 is used.
If HI is not specified, LO is used.

VI.E.12 DUMP RECORDS

<u>Purpose</u>: To provide a listing of specified records of a (possibly) binary file.

<u>Format</u>:
```
DU SRCE,DEST,LO,HI,MODE
Where SRCE = a physical or functional unit
      DEST = a physical or functional unit
      LO   = First record to be dumped (counting the next
             record as #1)
      HI   = Last record to be dumped
      MODE = A(ASCII), BA(absolute binary), or B(relocatable
             binary)
```

<u>Comments</u>:

If SRCE is not specified, DI is used.
If DEST is not specified, CL is used.
If LO is not specified, 1 is used.
If HI is not specified, dumping proceeds until an end-of-file on SRCE.
If MODE is not specified, A is assumed.

The process of dumping proceeds as follows. Each specified record is read from SRCE in the MODE specified. The characters of the record are then packed two per word, first the high order half, then the low order half. A listing of these words is then printed. In the listing, each word appears as six (6) octal digits, and these are printed eight (8) per line, using as many lines as necessary. Thus, for example, if the second record of the third file of logical unit D2 contains the ASCII character string ABCDEFGHIJKLM, then the first line of the output from the command

```
RE D2;SK D2,2;DU D2,,2,5
```

will be

```
Ø4Ø5Ø2  Ø415Ø4  Ø425Ø6  Ø4351Ø  Ø44512  Ø45514  Ø464ØØ
```

Note that an assumed zero character fills the right half of the last word.

In most applications, CL or LP would be used for DEST.

VI.E.13 FORMAT DISK

<u>Purpose</u>: To prepare a virgin disk for use with FDOS

<u>Format:</u> FD DISK
Where DISK = D,E,F, or G (floppy disk drive)

<u>Comments</u>:

(1) The sectors of the disk, aside from those of track zero, are chained together into a free storage list. They are not chained in simple ascending numerical order of address.. For each track, the chain starts at sector zero, proceeds through the even sectors (∅,2,4,6,1∅,12,14,16), then through the odd sectors (1,3,5,7,11,13,15,17).

(2) An initial disk directory is written into sector 1 of track zero. This contains 2∅ (a pointer to the first block of the free storage list) in its first location and the five Fixed Address files (SYS∅, SYS1, SYS2, SYS3,and SYSM) and addresses (14B, 15B, 16B, 17B, and 13B)as entries.

(3) Empty logical unit directories are written into sectors 4,5,6, and 17 of track zero.

(4) The Fixed Address files are generated as empty (EOF) and written out.

(5) The unconfigured Bootloader is written in sector three of track zero.

(6) The configured System Loader is written in sector zero of track zero.

VI.E.14 GO TO

<u>Purpose</u>: To allow the user to transfer from the Executive to a specified memory location.

<u>Format</u>: GO N
N is the octal address to which to transfer.

<u>Comments</u>: This command is useful in transferring to a core-resident program (such as the Debugger).

VI.E.15 IF DISK

Purpose: Determine whether or not the specified unit is a disk logical unit.

Format: ID UNIT
Where UNIT = a physical or functional unit

Comments: If UNIT is a disk logical unit, the next command in the command line is executed. If not, the remainder of the command line is ignored.

VI.E.16 LIST

Purpose: To provide a listing of specified records of an ASCII file.

Format: LI SRCE,DEST,LO,HI

```
        Where SRCE = a physical or functional unit
              DEST = a physical or functional unit
              LO   = First record to be listed (counting the next
                     record as number 1).
              HI   = Last record to be listed.
```

Comments:

If SRCE is not specified, DI is used.
If DEST is not specified, CL is used.
If LO is not specified, 1 is used.
If HI is not specified, listing continues until an End-of-File on SRCE.

In most applications, CL or LP would be used for DEST. If a cassette deck or disk logical unit is specified, the effect is to copy the specified records from SRCE to DEST, a function which cannot otherwise be performed. To skip records, BK could be used for DEST.

If one wished, for example, to determine the contents of the second file of the cassette in deck 2 by listing the first five lines, this could be accomplished by using the command

```
RE 2;SK 2;LI 2,,,5
```

VI.E.17 LIST ALL UNITS

Purpose: To print the contents of the Functional Device Table (FDT).

Format: LA UNIT
Where UNIT = unit on which to produce the listing.

Comments: The FDT is printed on UNIT in the same format as during the system generation.

If UNIT is not specified, CL is used.

VI.E.18 LOGICAL UNIT DIRECTORY

<u>Purpose</u>: To list the file names comprising the specified logical unit.

<u>Format</u>: LD DUNT

```
Where DUNT = D1,D2,D3,D4 (logical units of disk drive D)
             E1,E2,E3,E4 (logical units of disk drive E)
             F1,F2,F3,F4 (logical units of disk drive F)
             G1,G2,G3,G4 (logical units of disk drive G)
```

<u>Comments</u>: A logical unit consists of a list of names (up to five alphameric characters, first alphabetic) in a fixed order. These names correspond to files potentially, but not necessarily actually, on the corresponding disk. These lists, corresponding to the four logical units of the given disk, are kept in four sectors of track zero of that disk:

Sector (octal)	Logical Unit
4	4
5	1
6	2
7	3

These directories can be set up by the user through use of QUEUE command.

VI.E.19 QUEUE

Purpose: To set up a directory of the specified logical unit.

Format:
```
QU DUNT,NAME1,NAME2,. . .
Where DUNT = D1,D2,D3,D4 (logical units of disk drive D)
             E1,E2,E3,E4 (logical units of disk drive E)
             F1,F2,F3,F4 (logical units of disk drive F)
             G1,G2,G3,G4 (logical units of disk drive G)
Where NAME1, NAME2,. . . are file names.
```

Comments: (See comments under the Logical Unit directory command - LD.)

The LUD for DUNT is set to consist of NAME1,NAME2,..., and UNIT is rewound, so that it is positioned at NAME1.

QU DUNT deletes all names from the LUD, but does not delete the files from the disk (or Disk Directory).

When the command QU DUNT,NAME1 is given, NAME1 gets added to the LUD, but NAME1 does not get added to the Disk Directory until it is used for an output function.

VI.E.20 RESERVE FILE

Purpose: Reserve a file of a specified length.

Format:

```
RF DISK,NAME,#
Where DISK = D,E,F,G
      NAME = File name
      #    = Number of sectors
```

Comments: A file of the specified length and with the specified NAME is created on the specified disk.

If a file called NAME already exists on the given disk, its length is checked, and additional sectors are added to it if it is shorter than the number of sectors specified in the command.

The purpose of this command is to create a file in advance of operating with it. This reduces the amount of head motion (therefore, time) required in actually writing a file, since the free storage list (in track zero) does not have to be accessed each time another sector is required.

If, during subsequent output to this file, all reserved sectors are not used when the file is closed (by writing an EOF) the unused sectors are returned to the free storage list. If more sectors than were reserved are required during the actual output to the file, additional sectors are taken from the free storage list. Thus, one needs only to approximate the number of sectors required for the file.

VI.E.21 RESTORE

Purpose: Restore the original SIO configuration.

Format: RS

Comments: A "warm start" entry is made to the System Loader, and the SIO driver area is restored from disk, wiping out the effect of any Assign commands.

VI.E.22 RENAME

Purpose: To change the name of a specified file on a specified drive.

Format: RN DISK,OLDN,NEWN

```
        Where DISK = D,E,F,G  (floppy disk drive)
              OLDN is the name of the file before renaming;
              NEWN is the name of the file after renaming.
```

Comments: The name is changed in the Disk Directory. Also, the name word (which contains the name code computed from the file name) is changed in each block of the file. If the file in question is Queued on a disk logical unit, the name in the LUD is not changed.

VI.E.23 REWIND

Purpose: To position a unit at the start of the first file.

Format: RE UNIT
Where UNIT = a physical or functional unit

Comments: If UNIT is a cassette deck, a rewind of the specified deck is initiated, and control returns to the user without waiting for the rewind to be completed.

If UNIT is a disk logical unit, the effect of the command is modification of the in-core logical unit parameter table (LUPT) to indicate that the unit is positioned at the start of file 1 of that unit.

If UNIT is not specified, DI is assumed.

VI.E.24 RUN

<u>Purpose</u>: To load a specified file from drive D and transfer control.

<u>Format</u>: RU NAME,S.A.,parameters

Where NAME = name of file.
S.A. = (octal) start address if standard not desired.
parameters - format determined by program being called.

<u>Comments</u>: Same as RD, except drive D is assumed.

VI.E.25 RUN FROM SPECIFIED DISK

Purpose: To load a named file from a specified disk and transfer control.

Format:
```
RD DISK,NAME,S.A., parameters
Where DISK = D,E,F or G (floppy disk drive)
      NAME = name of the file
      S.A. = address to which to transfer control if
             standard not desired.
      parameters - format determined by program being run.
```

Comments: When this command is given, the Disk Directory is read from DISK. This directory contains the names and starting sector addresses of the files on that disk. If the name is not found, a message to that effect is printed. Otherwise, the parameters are used to set up PARAM through PARAM+7 of the Communication Region, S.A. is used to set STRTA in the Communication Region, and a transfer is made to the System Loader ELOAD entry with the appropriate drive and sector address.

VI.E.26 SAVE

Purpose: To write an absolute binary format image of the area of memory from FWA to LWA inclusively, capable of being loaded into memory locations ADDR thru ADDR+(LWA-FWA) at a later time.

Format:
```
SA UNIT,FWA,LWA,ADDR
Where UNIT = 1,2,3 (cassette decks)
             D1,D2,D3,D4 (logical units of disk drive D)
             E1,E2,E3,E4 (logical units of disk drive E)
             F1,F2,F3,F4 (logical units of disk drive F)
             G1,G2,G3,G4 (logical units of disk drive G)
```

Comments: The portion of memory lying between the two addresses FWA and LWA (inclusive) is written on the specified unit as a series of 128-word absolute binary format records. That is,

```
    Ø764ØØ                          (#data words, i.e., 125, in high
                                     order)

    FWA + 125(n-1)                  (starting address in memory of
                                     data of the n-th record)

    contents of first location
    contents of second location
             .
             .
             .
    contents of 125-th location

    checksum                        (sum of the address and the data
                                     words)
```

If the number of words written is not a multiple of 125, the n-th record is shorter, but of the same format as above.

If UNIT is not specified, DP is used.
If FWA is not specified, then 2B is used.
If LWA is not specified, then FWA (one word) is used.

If ADDR is not specified, FWA is used; ADDR specifies the intended load address. Thus, the words written from memory locations FWA thru LWA will be loaded back into ADDR thru ADDR+(LWA-FWA).

N.B. No file mark is written after the data records, so multiple saves can be used to build a file. When the desired portions of memory have been saved, a file mark MUST be written (to close the file) using the WE command.

(See example under COPY command.)

VI.E.27 SKIP

Purpose: To position a unit at the start of the N-th following file.

Format: SK UNIT,#

```
        Where UNIT = a physical or functional unit.
              #    = number of files to skip.
```

Comments: If UNIT is a cassette deck, successive "search" commands are issued, and control returns to the user when the drive becomes "ready".

If UNIT is a disk logical unit, the in-core logical unit parameter table (LUPT) is modified to indicate position at the start of the #-th file following the current one.

If UNIT is not specified, DI is assumed.
If # is not specified, 1 is assumed.

VI.E.28 SAVE START ADDRESS

Purpose: To write an absolute binary format record of locations 2B and3B as the standard start procedure, using the specified address as the start address (location 3).

Format: SS UNIT,ADDR

```
Where UNIT = a physical or functional unit.
      ADDR = (octal) start address desired.
```

Comments: An absolute binary format record (to be loaded into locations 2B and 3B) is written on the specified unit, without distrubing locations 2B or 3B.

This is usually used in conjunction with, and usually prior to, the Save command when building a file.

N.B. No file mark is written after this record.

If UNIT is not specified, the DP is assumed.
ADDR must be specified.

VI.E.29 TEMPORARY QUEUE

Purpose: To temporarily attach the specified file to the specified logical unit.

Format:

```
TQ UNIT,NAME
UNIT = disk logical unit
NAME = name of file to be attached.
```

Comments: The in-core LUPT is set up to the start of the file called NAME. The in-core file number is backed up by one, so that when EOF is detected in NAME, the LUPT is set up as it was before the TQ.

The LUD on disk is not modified, so TQ can be used with a write-protected disk.

VI.E.30 VERIFY

Purpose: To compare one or more files.

Format: VE SRCE,DEST,#,MODE

```
        Where SRCE = a physical or functional unit
              DEST = a physical or functional unit
              #    = number of files to compare
              MODE = A(ASCII), BA(absolute binary), B(relocatable
                     binary)
```

Comments: The remainder of the current file and the #-1 subsequent files of SRCE are compared with the corresponding files of DEST.

If Ø is specified for #, 1 is used, but the process terminates with the detection of a file mark on SRCE. (Normally, a final read of DEST would then be made and an error reported if no file mark were detected there.)

If SRCE is not specified,DI is assumed.
If DEST is not specified,DP is assumed.
If # is not specified, 1 is assumed.
If MODE is not specified, A is assumed.

(See the example under COPY.)

VI.E.31 WRITE

Purpose: To allow ASCII data to be output directly into a file on a unit.

Format:
```
WR DEST,SRCE
(line of text)
(line of text)
     .
     .
(CTRL/D)
```
Where DEST = a cassette deck, a disk logical unit, T,L, or P

Comments: As each line is input from SRCE (normally a keyboard), it is output to DEST. When the CTRL/D (followed immediately by Carriage Return then Line Feed) is input, a file mark is written on DEST and control returns to the user.

For example, if the user wanted to put a file of Batch commands at the start of logical unit E1, the following sequence could be used:

```
RE E1;WR E1
RE 2;SK E1;CO 2,E1;RE 2;RE E1;SK E1;VE 2,E1 [illegible]
RU ASMB,,,2;BX
(CTRL D)
```

If E1 is subsequently used as the batch input unit and DI, the above sequence will cause a source program to be copied into the second file of E1 and assembled, after which the system will leave the Batch mode.

If DEST is not specified, DP is used.
If SRCE is not specified, CK is used.

VI.E.32 WRITE END-OF-FILE MARK

Purpose: To terminate the current output file on a cassette or disk logical unit.

Format: WE UNIT
Where UNIT = a physical or functional unit.

Comments: If UNIT is a cassette deck a 3/4" gap of blank tape is written, followed by a record consisting of the single character 4B (control D, or EOT).

If UNIT is a disk logical unit, an end-of-file indication is written in the current block of the current file, any remaining blocks of the file are released, and the in-core logical unit parameter table (LUPT) is modified to indicate that the unit is positioned at the start of the next file.

N.B. The integrity of the data written into a file is not assured unless the file is properly terminated.

If UNIT is not specified, DP is assumed.

(See comments under the DELETE command describing how the remaining blocks of the file are released)

VI.E.33 ZERO SYSTEM

Purpose: To clear all SIO disk buffers of any holdover sectors before changing disk cartridges in a drive.

Format: ZS

Comments: The hueristics used in the SIO Driver to minimize disk sector reading and writing cause sectors to reside in core. If a disk is changed in a drive, and a sector is accessed that resides in core from the previous disk, SIO will use the old sector rather than read the new one (usually resulting in a Format Error).

The ZS is used to clear out all SIO buffers so that there is no hold-over, and all new sectors are read.

VI.F FDOS EXECUTIVE MESSAGES

1. The FDOS Executive prints messages when certain conditions arise. Those which apply generally to the various commands are as follows:

ILC
The last command processed by the Executive is defective in some way, most probably because an illegal character appeared or because one of the parameters does not fall within the expected set of values. The Executive discards the remainder of the command line and waits for the operator to type another.

NOT READY (unit)
When the Executive attempts to perform an operation on a unit, the driver, DIO, checks to see if the unit is ready before issuing the command. If the unit is not ready, the driver returns the "not ready" indication, and the Executive prints the "NOT READY (unit)" message.

WRITE LOCK (unit)
This message is printed if DIO returns the "write locked" indication in response to an attempt to write on a write-protected disk or cassette deck.

EOF (unit)
This message is printed if DIO returns the "end of file on read" indication in response to an attempt to read data from a unit. This is usually not an error condition during executive operations. It is printed mainly to help the operator keep track of the progress of multifile operations.

END OF TAPE (unit)
This message is printed if DIO returns the "end of tape" indication on an attempted read/write operation. This results in the termination of the command and the loss of the remainder of the command line. The Executive waits for the operator to type another command.

WRITE ERROR (unit)
This message is printed if DIO returns the "rate error" indication in an attempt to write on a disk.

READ ERROR (unit)
This message is printed if DIO returns a "rate error" or "CRC" (parity) error" indication in an attempt to read from a disk, or if DIO returns a "parity error" indication in an attempt to read from cassette.

FORMAT ERROR (unit)
This message is printed if, while accessing a named disk file, the name code in the first word of each sector does not verify with the file name (chaining error).

UNKNOWN DEVICE (unit)
No such peripheral device has been defined (at sys-gen time) in the system.

IMPROPER REQUEST (unit)
This message is printed if unit specified cannot perform the function requested.

TYPE 'GO' WHEN READY

This message is printed after "NOT READY (unit)" and "WRITE LOCK (unit)" messages and at other times. When the operator types "GO", the Executive retries the operation or goes on to its next task. If the operator types "NO", the Executive aborts the command.

2. The "fixed address files" reserved for the basic FDOS software system are named SYSØ, SYS1, SYS2, SYS3, and SYSM. To protect these, and to allow for other protected files, any file whose name begins with the letters "SY" is considered a "system" file. When such names are input to the Executive the following is typed for each reference:

```
SYxx?!
ARE YOU QUITE SURE?
```

Only the response of "JA" will cause the Executive to continue processing the requested command. Any other response will cause the "ILC" message, and that command (plus the remainder of the command line) will be aborted.

To avoid inconvenience when intentionally doing work with these files, the Executive has two additional commands:

SE	System name work Enable
SD	System name work Disable

In the Enabled mode (SE), the above discussed verification message is not typed, nor is a response expected. The "SD" command returns the Executive to the normal mode (ie., the verification is performed on each "SY" reference). In addition, the execution of a RE, WE, FD, DL, RN, AQ, or QU command causes the Enable mode to be exited at the end of the current command line; an illegal command or system reload causes an immediate exit from the Enabled mode.

VI.G. PAPER TAPE EQUIPMENT CONSIDERATIONS

Since paper tape equipment does not have file-manipulation capabilities, special considerations apply when it is specified in a command.

1. Ten nulls (feed frames) define an End-of-File on input from paper tape. Thus, when mounting a paper tape on the low-speed (TTY) reader, valid data for the desired file must be placed within ten frames of the read station; if this is not done, an "EOF LS TAPE" message will occur upon subsequent input from the low-speed reader.

2. On input from the high-speed reader, nulls are ignored until a valid record (of the specified type) is encountered; thus the tape can be mounted at any point on the leader.

3. If an operation with paper tape is aborted before completion (such as a compare error when verifying from paper tape to some other device) the driver retains the fact that the paper tape is in the middle of the file. Thus, if the operator positions the paper tape back to the beginning, the driver must be informed of this manual intervention. This can be accomplished by rewinding the appropriate reader (RE HT or RE LT). A HLT 6ØB will occur, thus reminding the operator to reposition the paper tape; when complete, press RUN.

4. If an ASR33 punch is used, two halts apply:

HLT 56B -	printing complete; turn on low-speed punch and press RUN.
HLT 55B -	punching complete; turn off low-speed punch and press RUN.

VII. SYSTEM PROCESSORS

The FDOS system processors provided are (primarily) modified versions of standard Hewlett-Packard programs. The modifications allow the various processors to be called by (and return to) the FDOS Executive, thus acheiving automatic, "hands-off" operation. The modifications also allow the processors to position the I/O media between passes, providing the media is disk or cassette. If the I/O unit defined is paper tape, certain halts will occur during processor operation to allow manual intervention; upon completion, RUN is depressed to continue the processor. These halts are:

HLT 6ØB -	end of paper tape; rewind tape, mount in reader, and press RUN.
HLT 57B -	end of source section; mount next paper tape and press RUN.
HLT 56B -	(ASR 33 only) printing complete; turn on low-speed punch and press RUN.
HLT 55B -	(ASR 33 only) punching complete; turn off low-speed punch and press RUN.

VII.A. EDITOR

This program is a modified version of the Hewlett-Packard Editor 20100B. Its operation is very similar to that described in the manual HP 02116-9016, with the primary difference that the "/D" device selection (instead of "/M") refers to input and output via FDOS functional units.

The following brief summary of the editors's operation can be supplemented by reference to the above-mentioned H-P manual:

'II.A.1. Entry of the edit file.

The program asks for the edit file device.

The following answers are accepted:

/T Edit file is to be typed
/A Edit file is to be read from AK.[illegible] cassette

The edit file consists of combinations of the following commands:

/F,n Open the n-th file. That is, copy files from the Input unit to the Output unit until the beginning of the n-th file is reached. /F,1 is assumed if any edit command other then /L or /L,n appears as the first line of the edit file.

/I,n Insert one or more lines after the n-th line of the currently opened file. The lines inserted are all those lines of the edit file lying between this control statement and the next control statement.

/D,n,m Delete line n through m of the currently opened file. /D,n is used to delete the n-th line.

/R,n,m Replace lines n through m of the currently opened file. /R,n is used to replace the n-th line. /R,n,m has the same effect as /I,n-1 followed by /D,n,m.

/CI,n,m Insert text after the m-th character of the n-th line of the currently opened file. The text to be inserted appears on the line following this control statement.

/CD,n,m,k Delete characters m through k of the n-th line of the currently opened file. /CD,n,m deletes the m-th character.

/CR,n,m,k Replace characters m through k of the n-th line of the currently opened file. This is equivalent to /CI,n,m-1 followed by /CD,n,m,k.

/L,n List the n-th file. /L is equivalent to /L,1. The only other command allowed in an edit file containing this command is /E.

/↑ Delete the last previous line of the edit file.

/E Close the currently opened file (if any), i.e., copy the remainder of it to the Output unit and terminate the edit. Note that if a /L,n command appears as the only other line in the edit file, no file is considered to be open, so no copying to the Output unit occurs.

If an end-of-file is encountered before a /E command is found, the editor says "END OF TAPE".

To read the remainder of the edit file from another tape, position the new tape in the Input unit and type "GO".

To terminate the edit file, type /C followed by /E.

VII.A.2. Specification of the symbolic file device.

The program asks for the symbolic file source device and the symbolic file destination device. The possible replies are:

/D	Use DI or DP.
/A	Use AI or AP.

If an end-of-file or end of punched tape is encountered during symbolic file input, the editor says "END OF TAPE". If further symbolic input is not required, the edit process is completed and a return to the Executive is made via a transfer to the System Loader (X754ØB). If further symbolic input is required (e.g., in the /F,n command), the Editor proceeds to input the next file from the input device. If paper tape is being used, the operator must mount the next tape in the reader.

VII.B ASSEMBLER

This program is a modified version of the Hewlett-Packard Assembler 24031B. It is supplied in three versions: Floating Point (ASMBF), EAU (ASMBE), and Non-EAU (ASMBN). It uses DI for source input, DP for binary output, and offers a choice of list output to HL or PL. Listing uses PL if the P option is specified during the call of the assembler (e.g., RU ASMBF RU ASMBF,,,P).

The assembler recognizes I/O options specified by the contents of certain locations in the parameter area of the Communication Region. If the assembler is loaded using the Executive program, these can be set by a call of the form:

```
RU ASMB,,P1,P2,P3
```

NOTE: The calls for all three assemblers (ASMBF, ASMBE, and ASMBN) are the same; the name of the version desired would be used instead of ASMB. It may be desirable to delete the unused assemblers from the working disk, and rename the desired one to ASMB (see Software Installation section).

If the Executive is not used, P1 through P3 must be set into cells PARAM +1 (X7525B) through PARAM +3 (X7527B). Numbers must be entered as their actual values, while letters are represented by their 7-bit ASCII codes. Missing parameters are set to zero. The parameters are interpreted as follows:

P1:K = accept control statement from PK;
   I = ignore first statement of the source program (assumed to be the control statement) and accept control statement from PK;
   If P1 is any character other than I or K, the control statement is expected to be the first statement of the source program.

P2: The number, counting from one, of the file on DI to be assembled; if not specified, defaults to assembling the first file.

P3:P = Write listing and symbol table (if requested in control statement) to PL; otherwise use HL. NOTE: If HL used and it is a disk logical unit, a file <u>must</u> be queued on HL even though no listing or symbol table is requested; this is because HL must be capable of receiving any error messages, etc., that are generated.

The assembly control statement consists of "ASMB" followed by one or more of the following:

,A Absolute assembly

,R Relocatable assembly

,B Produce a binary output

,L Produce a list output

,T Write the symbol table at the end of the list output (if any).

,C Follow the assembly by a call to the Cross-Reference Table Generator.

NOTE: If the listing is directed to HL, the normal error summary is also printed on CL at the end of each pass for the operator's convenience.

Use of the Assembly Control Statement C option causes the code for CROS to be set in LERR (X7537B) before the transfer to the System Loader (X754ØB) which occurs at the end of the assembly. The program first clears PARAM+1 (X7525B) and then loads the Cross-Reference Table Generator.

If the Assembler detects an End-of-Source Section condition (End-of-File on disk/cassette or End-of-Tape on paper tape prior to detecting an END statement), then the following applies:

1. If DI is a paper tape device, a HLT 57B is executed; this allows the operator to mount the next paper tape of the source program. Press RUN to continue the assembly.

2. If DI is cassette, the same HLT 57B considerations apply. However, upon detection of the END statement (interpass rewind) the Assembler will initiate a rewind, wait for rewind completion, then issue a HLT 57B,C (103057B instead of 102057B); this informs the operator to turn the cassette over and position it to the begining of the source program. An Assembler Control Statement C option (call for CROS) is treated the same way (interpass rewind).

3. If DI is a disk logical unit, no halts occur; the Assembler assumes that the various Source Section files have been Queued on DI prior to calling the Assembler.

The control statement errors "CS" and "R?" are printed on PL (in addition to HL if it was chosen in the assembler call) and a new control statement is accepted from PK. If the original statement was from the source program (DI), then the "I" option of the assembly call is forced (see P1 parameter description).

The Floating Point instructions FIX and FLT can be used even though the Floating Point hardware is not available. The Non-EAU and EAU Assemblers generate calls to library functions IFIX and FLOAT, respectively, and automatically provide for the external statement linkages (as they do for FDV, FMP, FAD, and FSB). Except for speed, the differences between the Non-EAU/EAU Assemblers and the Floating Point Assembler are transparent to the user.

In order to use the EAU shift-rotate instructions (ASR, ASL, LSL, RRR, RRL, and SWP) the EAU hardware must be available; the Non-EAU Assembler does not recognize these instructions.

The RAM instruction (not mentioned in most HP documentation), which is used for executing user's microprograms, is implemented in all versions of FDOS assemblers.

VII.C. CROSS-REFERENCE TABLE GENERATOR

This program is a modified version of the Hewlett-Packard Cross-Reference Symbol Table Generator program 24109B. It uses DI for input of the source program and offers a choice of list output to HL or PL. Listing uses PL if the P option is specified during the call.

The Cross-Reference Table Generator recognizes several options specified by the contents of certain cells in the parameter area of the communication region. If the program is loaded using the Executive program, these can be set by a call of the form:

```
RU CROS,,P1,P2,P3
```

If the Executive is not used, P1 through P3 must be set into cells PARAM+1 (X7525B) through PARAM+3 (X7527B). Numbers must be entered as their actual values, while letters are represented by their 7-bit ASCII codes. Missing parameters are set to zero. The parameters are interpreted as follows:

P1:K =	accept a range of characters from PK, and do a cross reference table of all symbols whose initial characters fall into the character range included between the two characters typed. Otherwise, do a table of all symbols.
P2:	The number, counting from 1, of the file on DI to be processed.
P3:P =	write output to PL. Otherwise, use HL.

If the Cross-Reference Table Generator program is to be called automatically when the C Assembly Control Statement option is specified, it must be on the system disk (if the Executive is used) with the name CROS. In the case of an automatic call from the assembler, P1 is cleared, and P2 and P3 are as they were specified for the assembler.

VII.D. DEBUGGER

The interactive debugger program (DEBUG) supplied by Dicom is used to interface with machine language, SIO-oriented, user routines as an aid in debugging. DEBUG provides for an active breakpoint within the user's routines. The accumulators (A and B registers) and memory can be examined and modified from the console device TTY (either the standard HP-interfaced teleprinter or "deck zero" Teletype/CRT) after a breakpoint has occured.

When the breakpoint (that has been inserted in the user's program) is encountered, DEBUG is entered via a jump through location 4B. Upon entering DEBUG via the breakpoint, the contents of the A, B, Overflow, and Extend Registers are saved; thus these registers can be restored by DEBUG when returning to the user's routine via the Proceed or Run Commands of DEBUG. When entering DEBUG via the breakpoint, the breakpoint address and the values of the A and B registers are printed on the console device; at this point DEBUG can be used to examine memory locations, dynamically "patch" the user's program, establish a new breakpoint, and re-enter the user's program.

DEBUG commands are single keystrokes (usually letters) preceeded by an argument when applicable. The argument defines a memory address. The commands recognized by DEBUG are described below.

II.D.1. DEBUG COMMANDS

N/	Print contents of location N. Location N is then open, so an octal number followed by RETURN or LINEFEED will cause that number to be stored into location N. If no number is typed before the RETURN or LINEFEED, location N is not modified.
N;	Print contents of location (N+offset). This location is now open, as above.
/	Reopens location N.
;	Reopens location (N+offset).
LINE FEED	Close current location (N or N+offset), changing the contents if a number was typed before the LINEFEED, and open next by typing (N+1)/ XXXXXX or (N+1); XXXXXX

RETURN	Same as LINEFEED, except that the next location is not opened.
↑	Same as LINEFEED, except that the previous location, rather than the next, is opened.
Ø/	Examine and open the A-register location. This location is set to the value of the A-register on each entry into DEBUG, and it is used to set the A-register before transferring control in response to P or R.
1/	Same as Ø/, except for the B-register.
B	Examine and open breakpoint. A value of zero indicates that there is no breakpoint. A non-zero breakpoint causes a JSB 4B,I to be placed in the location specified just before the DEBUG transfers control in response to P or R. (N.B. Location 4B is used as a pointer). A breakpoint can not be used on multiple-word instructions.
E	Examine and open the E and O location. This location is set to the values of the Extend and Overflow registers on each entry into DEBUG, and it is used to set them before transferring control in response to P or R. The format of the word is: E=bit 15; O=bit Ø.
I	Examine and open the location specified in bits 14-Ø of the contents of the last location opened.
J	Examine and open break pointer location.
NL	Relocate DEBUG by a multiple of 2ØØØB, so that it resides on the memory page containing location N.
M	Examine memory referenced by last contents.
P	Proceed from a breakpoint. When a breakpoint has been inserted into a program at location XXXXX and the program reaches that location, the debugger prints XXXXXB aaaaaa bbbbbb Where aaaaaa and bbbbbb are the contents of the A- and B- registers. The operator can now use DEBUG

commands to modify memory locations (including XXXXX), move the breakpoint, change the values of A,B,E and O, etc. If it is then desired to continue execution of the program, the P command is used.

If the breakpoint has not been moved, then P cannot be used if XXXXX contains a JSB to a subroutine which expects an argument in a location relative to the location of the JSB, or which has a return to any location other than one of the two locations following the JSB.

NP	Proceed from the breakpoint N times.
NR	Run from location N. A,B,E and O are set and control transfers to location N.
S	Proceed from the breakpoint, but suppress break message.
NS	Proceed from the breakpoint N times, but suppress all but last break message.
NT	Trace - move breakpoint to location N and proceed from the breakpoint.
X	Examine and open the offset register. This is the offset used in connection with the semicolon.

II.D.2. STARTING DEBUG

When DEBUG is called via the Executive, it is loaded into locations Ø42ØØB through approximately Ø54ØØB, and is automatically executed; it assumes that the console device is on channel 12B (since DEBUG is a "stand-alone" program that does not use the SIO driver package). If a different channel is desired, halt the computer, set bits 5 through Ø of the location 42Ø1B to the desired channel (with bit 15 = 1 if "deck zero" is used), and restart the computer at location Ø42ØØB. If it is desired to write this modified version of DEBUG to the disk, relocate DEBUG (use Ø2ØØL), reload the Executive (use X7544R), remove the write protect tab from the system disk, and use the following command string to write DEBUG to disk:

```
TQ D1,DEBUG;SA D1,2ØØB,14ØØB;WE D1
```

VII.E. FDOS FORTRAN COMPILER

The FDOS FORTRAN package consists of modified versions of the Hewlett-Packard FORTRAN pass one and pass two programs (HP 20548A). Pass two is supplied in two versions; Floating Point/EAU(FOR2E) and Non-EAU(FOR2N). The package uses DI for source input, SC for intermediate-tape storage, DP for binary output, and offers a choice of list output (if requested) to HL or PL.

FORTRAN recognizes options specified by the contents of certain locations in the parameter area of the Communications Region. If FORTRAN is loaded using the Executive program, these can be specified by a call of the form:

```
RU FORT,,P1,P2,P3,P4
```

If the Executive is not used, parameters P1 through P4 must be set into cells PARAM+1 (X7525B) through PARAM+4 (X753ØB). Numbers must be entered as their actual values, while letters are represented by their 7-bit ASCII codes. Missing parameters are set to zero. The parameters are interpreted as follows:

P1:K = accept control statement from PK;
   I = ignore first statement of the source program (assumed to be the control statement) and accept control statement from PK;

If P1 is any character other than I or K, the control statement is expected to be the first statement of the source program.

P2: the number, counting from one, of the file on DI to be compiled; if not specified, defaults to compiling the first file.

P3:P = write all listings and symbol table (if requested in control statement) to PL; otherwise use HL.
NOTE: If HL used and it is a disk logical unit, a file _must_ be queued on HL even though no listing or symbol table is requested; this is because HL must be capable of receiving any error messages, etc., that are generated.

P4:E = use Floating Point/EAU version of pass two (FOR2E).

   X = use FOR2X or FOR2F version of pass two; these names
or F have been included in order to accomodate user-supplied (or future) versions of pass two.

If P4 is any character other than E, F, or X, the Non-EAU version of pass two (FOR2N) will be used.

The compiler control statements consist of "FTN" followed by one or more of the following:

,A Produce Assembly level listing (Pass two)

,B Produce Relocatable Binary

,L Produce source listing (Pass one)

,T Produce Assembly level symbol table only (Pass two).

The FDOS FORTRAN Compiler called is stored on disk in two files: pass one (FORT) and pass two. FORT performs an initial pass over the source program, converting the FORTRAN statements into an intermediate binary form. This intermediate binary file is written onto a scratch file (the first file on SC is used), which is read by pass two and translated into standard relocatable binary format.

To operate FORTRAN, the user must Queue the FORTRAN source program in DI, Queue a file for the relocatable binary as the next file on DP, Queue a file on SC for the intermediate-tape, and Queue a file on HL (if required). If HL is specified for list and HL is a disk logical unit, a file must be available (even though no listing is requested; see P3 description); if this is not done, the compiler will halt with an "End-of-tape in write"(HLT 66B, see Section IV.C.1).

A scratch file must be Queued as the first file on the scratch unit (SC). This scratch file can be left on the disk, where it will be used each time FORTRAN is run, or it may be deleted after FORTRAN is run. FORTRAN will run faster if the scratch file already exists on the disk, so it is wise to simply leave the temporary file on the disk.

After the files have been Queued appropriately, the user types:

```
RU FORT,,P1,P2,P3,P4
```

When the program comes into memory it positions DI and ejects a page on the list unit (just a "thunk" on PL if the Form Feed option is in effect). Then it waits for the user to type the control statement if the P1 = K or I. The compilation is then performed, and the program returns to the FDOS Executive by transferring to the System Loader (X754ØB).

VII.F DISK UTILITY PACKAGE

The FDOS Disk Utility Package (DUP) was designed primarily to duplicate disks. It is structured to copy and/or verify on a track for track basis (instead of being sector oriented as is the rest of FDOS). This approach results in a significant saving of time; it takes approximately forty-five seconds to reporduce (copy and verify) an entire 131K word disk using DUP, vs. two and one-half minutes using a sector for sector basis.

DUP is not part of the Executive, since DUP requires 4K of memory (two tracks) for buffers.

When DUP is called, command information is passed via the standard parameter scheme. If DUP is called using the Executive program, these can be specified by a call of the form:

```
RU DUP,,P1,P2,P3,P4,P5,P6
```

If the Executive is not used, parameters P1 thru P6 must be set into cells PARAM+1 (X7525B) through PARAM+6 (X7532B). Numbers must be entered as their actual values, while letters are represented by their 7-bit ASCII codes. Missing parameters are set to zero. The parameters are interpreted as follows:

P1: command; C (copy), V (verify), or R (reproduce-copy and verify).

P2: source disk drive (SDISK); D,E,F,G, or S.

P3: destination disk drive (DDISK); D,E,F,G, or S.

P4: initial disk address (FSECT).

P5: final disk address (LSECT).

P6: target disk address (TSECT; initial sector on destination disk drive).

NOTE: Parameters entered for sector addresses are treated as decimal values unless followed by "B".

Default values for the parameters are:

```
P1 (CMD)   = R
P2 (SDISK) = S
P3 (DDISK) = E
P4 (FSECT) = Ø
P5 (LSECT) = FSECT if FSECT is specified,
             else use 1777B.
P6 (TSECT) = FSECT
```

If "D" is specified for either SDISK or DDISK, then when DUP is loaded it types

```
DISKS READY?
```

to allow the operator to place the desired disks in the drives. A response of "Y" initiates the specified command. A response of "X" causes an Executive reload via a transfer to the System Loader. Defining SDISK as "S" (meaning a "system" disk) causes drive "D" to be used, but DUP skips this question.

When DUP has completed processing the specified command it types

```
DONE
SYSTEM READY?
```

to allow the operator to exchange disks. A response of "A" (again) causes DUP to repeat the specified command; a response of "Y" causes the Executive to be reloaded via a transfer to the System Loader (NOTE: a "system" disk must be installed in drive D).

If DUP is called using defaults for all parameters (RU DUP) the disk in drive D is duplicated (copied and verified) onto drive E. DUP communicates using the Console List and Console Keyboard (CL nad CK) units.

ERRORS:

If the parameters entered are in error, DUP types

```
            ILLEGAL REQUEST
```

and reloads the Executive via a transfer to the System Loader.

If an error is detected during verification, DUP types

```
            COMPARE ERROR
```

and proceeds to the "SYSTEM READY" question.

If bit 15 is set between transfers, then DUP types

```
            ABORTED
```

and proceeds to the "SYSTEM READY" question.

All device errors cause SIO/XIO halts; see Section IV.C.l.

VIII. FDOS BCS PACKAGE

In addition to the Basic Control System Drivers provided, other programs are needed to allow the Hewlett Packard computer user to take full advantage of the FDS in a BCS environment. The FDOS BCS package provides disk I/O facilities during the Prepare Control System process. It also provides console device interaction, I/O capabilities, and an interface with the SIO Driver package (and, consequently, with the entire FDOS system if desired) during the loading process.

The FDOS BCS package contains the following modules:
*Disk Prepare Control System (DPCS), absolute binary
*Input/Output Control (.IOC), relocatable binary
*Teleprinter BCS Driver (D.ØØ), relocatable binary
*Cassette System BCS Driver (D.35), relocatable binary
*Floppy Disk System BCS Driver (D.36), relocatable binary
*FDOS Relocating Loader (RLOA), relocatable binary
*Subroutine Library, relocatable binary
In addition, a configured BCS file (ABCS, Absolute Basic Control System) is provided; this file was built using the above modules and hardware configuration in Section V.A.

III.A. DISK PREPARE CONTROL SYSTEM

The Disk Prepare Control System (DPCS) is a modified version of the standard Hewlett Packard Prepare Control System (PCS). DPCS processes relocatable modules of the Basic Control System and produces an Absolute Basic Control System file (ABCS) that is configured to work with a specific hardware configuration. It creates operating units of the Input/Output Control subroutine (.IOC), the equipment driver subroutines (BCS Drivers), and the Relocating Loader (RLOA). It also establishes the contents of certain locations used in interrupt handling. Options are available to define the equipment driver modules and other BCS system subroutines as relocatable programs to be loaded with the user's object program.

DPCS is an absolute program which runs in the SIO environment using the configured SIO Driver (see Section V). The order in which the BCS modules are loaded and processed by DPCS is not significant, except that the disk BCS Driver must be the first module loaded and the FDOS version of the Relocating Loader must be the last module loaded. Two modules, the Input/Output Control subroutine and the Relocating Loader, require that the parameters be entered via the keyboard after being loaded.

The EQT input of IOC has been modified in two ways to handle the D.36 driver: first, the Unit Number can be one or two digit octal value in the range Ø to 37; secondly, another input following the Unit Number has been added which defines the offset between the disk data channel and the disk control channel as a "C" followed by a one or two digit octal value in the range 1 to 17. Thus, the Equipment Table Statement is of the form:

```
nn,D.ee(,D)(,Uu(u))(,Co(o))
```

The first word of the EQT has been changed to assign bits 1Ø thru Ø6 to the Unit Number (instead of bits Ø8 thru Ø6), and to assign bits 14 thru 11 to the disk control channel offset. These added bits (14 thru Ø9) were previously unused.

In addition to the above changes, all loader diagnostics and completion of DPCS cause a reload of the Executive (via a transfer to the System Loader, X7544B). [illegible] a [illegible] routine [illegible] included [illegible]

DPCS is an absolute program which runs in the SIO environment; it uses the SIO Driver to load the various BCS modules from DI and to write the configured BCS file (ABCS) onto DP. [illegible] onto the system [illegible]

Since the SIO Driver has been configured for the I/O aspects (see Section V.A), most of the initialization phase of HP's PCS is not required in DPCS. It should be noted that DPCS can be used to prepare a BCS system for a hardware configuration different from the one that DPCS is being run on. This is possible because the ABCS file is not built in core, but on the disk.

To operate DPCS, proceed as follows:

1. Since the BCS Drivers to support a hardware configuration defined in Section V.A are Queued on D3 of the "skeleton" disk shipped from the factory, it is only necessary to reassign DI to this unit prior to calling DPCS. This, along with obtaining a list of the modules that will subsequently be processed by DPCS, is accomplished by typing

```
AS DI,D3;LD DI
```

The Executive responds by typing

```
D.36
D.36X
D.ØØ
D.35
.IOC.
RLOA
```

If additional BCS Drivers are to be incorporated, they should be copied to disk and Queued to DI preceeding RLOA.

2. Open a file on DP to receive the ABCS file generated by DPCS, and call DPCS by typing

```
QU DP,ABCS;RU DPCS
```

NOTE: Following this section is a sample dialog of DPCS.

3. DPCS will be loaded, and will type

```
FWA MEM?
```

requesting the first word of available memory, i.e., the first word in the base page following the locations required for interrupt processing; this word defines the start of the BCS system linkage area. Here, as with all subsequent addresses requested by DPCS, reply with an <u>octal</u> number.

4. DPCS will type

```
LWA MEM?
```

requesting the last word of available memory. This word is usually the location prior to the protected area (e.g., X7677B), or the location prior to the FDOS System Loader (X7533B) so that the BCS system may interact with the System Loader (and therefore, the entire FDOS if desired). Reply appropriately.

5. DPCS will type

```
SYS RST?
```

requesting the system restart address to be used by BCS when returning to the SIO environment via the System Loader. Use X7544 (Cold start) since SIO and BCS use different in-core buffers and tables.

6. DPCS will type

```
          *LOAD
```

informing the operator that it is ready to process a BCS module from the system Input unit (DI). The modules may include .IOC, the various BCS drivers, and the Relocating Loader (RLOA). To cause a module to be loaded, strike any key on the keyboard followed by Return-Line Feed. The "*LOAD" message is repeated after each module is loaded until RLOA has been processed. Diagnostics are printed if certain error conditions occur during the loading.

The absolute lower and upper bounds of each module within BCS are listed after the module is loaded. The format is as follows:

```
           MODULE NAME
           11111 uuuuuu
```

NOTE: For details of Input/Output Control (.IOC) and the Relocating Loader (RLOA) operation, the user should refer to the Hewlett Packard BASIC CONTROL SYSTEM manual, P/N 02116-9017; to A POCKET GUIDE TO THE 2100 COMPUTER, or to A POCKET GUIDE TO HEWLETT-PACKARD COMPUTERS; or to A POCKET GUIDE TO INTERFACING HP COMPUTERS.

7. When .IOC has been loaded, DPCS requests the information needed to construct the Equipment Table (EQT), the Standard Equipment Table (SQT), and to define the Direct Memory Access statement. Refer to the dialog following this section for examples of these.

8. When RLOA has been loaded, DPCS requests the parameters needed to set the interrupt linkages for input/output processing; for examples, refer to the dialog following this section. When these have been supplied, DPCS goes on to Processing Completion; the final step in DPCS processing is the writing of the Absolute Basic Control System file (ABCS) on the system Output unit (DP). When this is completed, control is returned to the FDOS Executive via a transfer to the System Loader.

VIII.B. FLOPPY DISK SYSTEM BCS DRIVER, D.36

The FDS BCS Driver (D.36) is a relocatable driver operating in the interrupt environment which accesses the disk data in the same format as does the SIO Driver. D.36 accepts commands and returns status in the standard form expected by the Input/Output control program (.IOC). This driver is not designed to be used with the Hewlett Packard Buffered IOC, since that program cannot assure the sixty microsecond interrupt response time required by the FDS.

A seperate BCS module, D.36X, is also provided. This module contains the Logical Unit Parameter Tables and Buffers for D.36, which D.36 refers to with externals to determine how many buffers are available and how many drives are to be supported. The source (SD36X) of this module is also provided so that the user can edit the module to any unique requirements.'

The commands on the following page are accepted by the driver.

Bits 15	14	13	12	11	10	9	8	7	6	Operation
Ø	Ø	Ø	Ø	X	X	X	X	X	X	CLEAR REQUEST
Ø	Ø	Ø	1	X	X	X	X	X	Ø	READ ASCII RECORD
Ø	Ø	Ø	1	X	X	X	X	X	1	READ BINARY RECORD
Ø	Ø	Ø	1	X	X	X	X	1	1	READ VARIABLE BINARY RECORD
Ø	Ø	1	Ø	X	X	X	X	X	Ø	WRITE ASCII RECORD
Ø	Ø	1	Ø	X	X	X	X	X	1	WRITE BINARY RECORD
Ø	Ø	1	1	X	X	X	Ø	Ø	1	WRITE END-OF-FILE
Ø	Ø	1	1	X	X	X	Ø	1	1	SKIP ONE ASCII RECORD
Ø	Ø	1	1	X	X	X	1	Ø	Ø	REWIND
Ø	Ø	1	1	X	X	X	1	1	Ø	SKIP TO END-OF-FILE
Ø	Ø	1	1	X	X	X	1	1	1	SKIP ONE VARIABLE-BINARY RECORD

NOTE: A read record command with a buffer address of zero will produce a skip-one-record of the designated type (i.e., ASCII, binary, or variable binary). All other commands are rejected by the driver.

The fol[illegible]

The following status returns are generated by the driver:

Bits 7	6	5	4	3	2	1	Ø	Meaning
1	X	X	X	X	X	X	X	END-OF-FILE DETECTED
X	1	X	X	X	X	X	X	REWIND JUST EXECUTED (AT BOT)
X	X	1	X	X	X	X	X	END-OF-TAPE ON READ, DISK FULL ON WRITE
X	X	X	1	X	X	X	X	WRITE ERROR (OTHER THAN WRITE PROTECTED)
X	X	X	X	1	X	X	X	BROKEN TAPE (SECTOR CHAINING IN ERROR)
X	X	X	X	X	1	X	X	WRITE PROTECTED
X	X	X	X	X	X	1	X	READ ERROR
X	X	X	X	X	X	X	1	DEVICE BUSY OR NOT READY OR INOPERABLE

Bit 14 of EQT word 2 is set for read errors and write errors other than write protected.

The Driver returns to .IOC. with the following conditions:

A REGISTERS	B	Meaning
Ø Ø Ø Ø Ø Ø	Ø Ø Ø Ø Ø Ø	OPERATION INITIATED, IN PROGRESS
1 Ø Ø Ø Ø Ø	Ø Ø Ø Ø Ø Ø	OPERATION EXECUTED AND COMPLETED
Ø Ø Ø Ø Ø 1	Ø Ø Ø Ø Ø Ø	ILLEGAL FUNCTION WAS REQUESTED
Ø Ø Ø Ø Ø 1	1 Ø Ø Ø Ø Ø	DRIVER OR DEVICE IS BUSY OR NOT READY OR A "BROKEN TAPE" CONDITION EXISTS

If the EQT is incorrectly configured, i.e., illegal Logical Unit number or the DMA state from EQT does not match DMA state of the driver, then the ***FATAL*** halt at "IOERR" (in.IOC) will occur, with A = 3 (for misconfiguration) and B = absolute address of the "JSB .IOC".

VIII.C. DISK DATA I/O DURING BCS OPERATION

Since the disk is a file-oriented device, and since D.36 has file-manipulation capabilities, the disk can be thought of as a mag tape. Thus, a user program can store and retrieve data on disk by using the FORTRAN Auxilary Input/Output Statements of REWIND and ENDFILE as an example. However, prior to doing disk I/O, the file or files must be "opened" on the appropriate disk logical unit (the Unit Reference Numbers defined in the EQT during DPCS) and the Logical Unit Directory set up; this is anagolous to mounting the mag tape.

This "opening" of files and LUD set up can be accomplished via the Queue command of the FDOS Executive prior to calling the user program. It may also be appropriate to use the Reserve File command to reduce the amount of head slewing (see QU and RF command descriptions in Section VI).

During some application programs it may be desirable to have the same file Queued on both the Output Unit and the Input Unit. Thus, the user program could output an array, "close" the file (ENDFILE), reposition the file (REWIND), then input and process the array.

A note of caution is appropriate here. The .IAR. routine of the standard HP formatter reads data in fixed blocks of 120 characters (60 words) during an Unformatted Read. Thus, calls must be in multiples of 60 words, or input data will be lost (i.e., if 50 was requested, words 51 thru 60 would go to the "bit bucket"). Also, during Unformatted Read, the total number of words requested must be a multiple of 60.

VIII.D. THE CASSETTE SYSTEM BCS DRIVER, D.35

D.35 is a relocatable driver for the cassette system. It accepts commands and returns status in the standard form expected by the Input/output Control program (IOC). This driver is not designed for use with Hewlett-Packard Buffered IOC, since that program cannot assure the 2 ms interrupt response time required by the cassette system.

The following commands are accepted by the driver:

Bits 15 - 12	Operation:
0	Clear
1	Read a record Bit 6 = 1 for binary, Ø for ASCII
2	Write a record Bit 6 = 1 for binary, Ø for ASCII
3	Control function:

Bits 8 - 6	Function:
1	Write End-of-File
3	Skip one record
4	Rewind
6	Skip to end of file
7	Skip one record

Any other commands are rejected by the driver.

The following status returns are generated by the driver:

Bit	Meaning:
7	End-of-file detected during read
6	Leader detected after rewind
5	Leader detected after operation other than rewind
4	Write command rejected, but not for write protected cassette
3	Not Used

2	Write protected cassette
1	Error detected during read
∅	Device busy (not ready)

End of tape status (Bit 5) can only be cleared by issuing a rewind command. Read error (Bit 2) and write commands rejected for reasons other than write protected cassette (Bit 4) cause Bit 14 of EQT word 2 to be set.

VIII.E. THE ABSOLUTE BASIC CONTROL SYSTEM FILE

The Absolute Basic Control System file (ABCS) was built with DPCS and the hardware configuration defined in Section V.A. The following differences exist between the Relocating Loader module (RLOA) of this file and the standard Hewlett Packard Relocating Loader:

*Interaction is via the console device rather than the Switch Register
*All loader diagnostics cause a return via the System Restart Address (see Section VIII.A.5).
*The subroutine library is a file named LIBN, LIBE, LIBF, OR LIBX Queued on the library unit D4.
*After writing an absolute binary file, or in response to a HALT call, ABCS executes a JMP to the System Restart Address (see Section VIII.A.5).

When ABCS (of which RLOA is a module) is loaded, the message "*OPT" is typed on the console device. The user must respond with one or two characters as follows:

"P" requests writing an absolute binary file on the system Output unit (Unit-Reference no. 4, normally E2), followed by a file (on the system Output unit) containing the entry point list; if a second file is not Queued on the unit, the entry point list is lost.

"L" requests printing a bounds list on the system list device (Unit-Reference no. 6, normally E3). All other characters have no effect, except that they are counted (e.g., "Q" signifies to the program that no options are desired).

Whenever an end of tape condition (e.g., a File Mark) is encountered, the program types "*LOAD". At this point the user must type one character (D,F,I,N,T,E or X), and may type "L" if desired.

These have the following meanings:

I = Load another file from the system Input device (as defined in SQT).

D = Proceed to the end-of-loading phase.

T = Terminate loading.

N = Load from LIBN (Non-EAU Library).

E = Load from the LIBE (EAU Library).

F = Load from LIBF (Floating Point Library).

X = Load from LIBX (User-supplied library).

L = (with any of the above) Produce a bounds list on the system list unit (Unit-Reference no. 6, normally E3); if a file is not Queued on the list unit, the bounds list is lost.

At the end-of-loading, the program types ""*LST". The user must type one character:

L = Produce an entry point list on the console list device (Unit-Reference no. 2, normally the teletype)

Any other character = Do not produce an entry point list on the console device

Upon completion, ABCS prints the "*END" message, and reloads the Executive via a transfer to the System Restart Address (normally the System Loader "cold start", X7544B).

VIII.F. BCS DEBUGGING SYSTEM

The debugging routine provided by Hewlett Packard as part of the standard BCS package is fully compatible with the FDOS BCS package. The user is refered to the Hewlett Packard BASIC CONTROL SYSTEM manual, P/N 02116-9017, for integration and operation instructions.

VIII.G. SAMPLE DIALOG FOR DPCS

```
*AS DI,D3
*LD DI
D3:
D.36
D.36X
D.00
D.35
.IOC.
RLOA
```

Logical Unit D3 is Assigned to DI. A Logical Unit Directory for DI is obtained to show which modules will be processed by DPCS.

```
*OU DP,ABCS
*RU DPCS
FWA MEM?
25
LWA MEM?
27533
SYS RST?
27544
```

A file is opened on DP to accept ABCS. DPCS is called. The First Word of Available memory, the Last Word of Available memory, and the System Restart address is defined.

```
* LOAD
Z
```

DPCS is ready to process modules; "Z" (or any character) directs it to load.

```
D.36
 25365 27533
```

The FDS BCS Driver is loaded and its bounds listed.

```
* LOAD
Z

D.36X
 24716 25364
```

The FDS BCS Driver's "externals" module is loaded and its bounds listed.

```
* LOAD
Z

D.00
 24162 24715
```

The teleprinter BCS Driver is loaded and its bounds listed.

```
* LOAD
Z

D.35
 23320 24161
```

The cassette system BCS Driver is loaded and its bounds listed.

```
* LOAD
Z

IOC
 23101 23317
```

IOC is loaded and its bounds listed.

```
* TABLE ENTRY
                         Unit-
                         Reference
EQT?                     Number
10,D.36,U0,C1              7
10,D.36,U1,C1             1Ø
10,D.36,U2,C1             11
10,D.36,U3,C1             12
10,D.36,U4,C1             13
10,D.36,U5,C1             14
10,D.36,U6,C1             15
10,D.36,U7,C1             16
12,D.ØØ                   17
14,D.35,U0                2Ø
14,D.35,U1                21
14,D.35,U2                22
14,D.35,U3                23
/E
```

The Equipment Table is defined; Logical Unit D4 becomes EQT entry 7, D1 becomes 1Ø, D2 becomes 11, D3 becomes 12, E4 becomes 13, E1 becomes 14, E2 becomes 15, E3 becomes 16, cassette deck zero becomes 2Ø, deck 1 becomes 21, deck 2 becomes 22, deck 3 becomes 23, and the teleprinter becomes EQT entry 17.

```
SQT?
-KYBD?
17
-TTY?
17
-LIB?
7
-PUNCH?
15
-INPUT?
14
-LIST?
16
```

Unit-Reference numbers are assigned to the units in the Standard Equipment Table. The Library will be D4, system Output (PUNCH) will be E2, system Input will be E1, and High-speed List (LIST) will be E3.

```
DMA?
Ø
```

No Direct Memory Access channels are available.

```
* LOAD
Z

LOADR
 20345 23ØØ5
```

The Relocating Loader is loaded and its bounds listed.

```
INTERRUPT LINKAGE?

10,20,I.36
11,21,C.36
12,22,I.ØØ
13,1Ø7Ø13
14,24,I.35
/E
```

The Interrupt Linkages are defined for channels 1Ø, 11, 12, and 14; since no BCS Driver was included for the device in channel 13, a halt will be executed if an interrupt occurs on channel 13.

```
.SQT.   23006
.FQT.   23014
C.36    26405
D.36    25646
DICOM   25365
I.36    26431
RETRY   26345
.BUFS   25322
.LUPT   25324
.STBL   24717
IOERR   23275
NBUFS   24716
NLOGU   25323
D.00    24162
I.00    24336
.BUFR   23247
D.35    23320
I.35    23757
.IOC.   23101
DMAC1   23316
DMAC2   23317
XSQT    23314
XEQT    23315
LST     20376
.LDR.   22001
.MEM.   22777
HALT    22774
```

A list of the entry points is printed.

```
*SYSTEM LINK
 00025 00301

*ZS;RS
*
```

The bounds of the System Linkage area are printed. Upon completion of the writing of the ABCS file to DP, the Executive is automatically reloaded. Note that ABCS is built on disk, not in core. After returning to the Executive, we zero out the system and restore the I/O assignments.

FDOS COMMAND SUMMARY

COMMAND	FORM	DEFAULT CONDITIONS
Add to LUD.	AQ DUNT,NAM1,NAM2,...	None.
Assign.	AS FUNT,PUNT,LCTR,PGCTR	LCTR & PGCTR as before.
Batch.	BA NAME	NAME = next file on BC.
Batch Exit.	BX NAME	NAME = next file on BC.
Comments.	** Text	N. A.
Copy.	CO SRCE,DEST,#,MODE	SRCE = DI, DEST = DP, # = 1, MODE = ASCII.
Directory.	DI DISK,DEST	DEST = CL.
Directory with Addresses.	DA DISK,DEST	DEST = CL.
Delete.	DL DISK,NAME	None.
Dump Disk.	DD DEST,DISK,LO,HI	DEST = CL, LO = 2, HI = LO.
Dump Memory.	DM DEST,LO,HI	DEST = CL, LO = 2, HI = LO.
Dump Records.	DU SRCE,DEST,LO,HI,MODE	SRCE = DI, DEST = CL, LO = 1, HI = EOF, MODE = ASCII.
Format Disk.	FD DISK	None.
Go To.	GO ADDR	None.
If Disk.	ID UNIT	None.
List.	LI SRCE,DEST,LO,HI	SRCE = DI, DEST = CL, LO = 1, HI = EOF.
List All Units.	LA UNIT	UNIT = CL.
Logical Unit Directory.	LD DUNT	None.
Queue.	QU DUNT,NAM1,NAM2,...	None.
Reserve File.	RF DISK,NAME,#	None.
Restore.	RS	N. A.
Rename.	RN DISK,OLDN,NEWN	None.
Rewind.	RE UNIT	UNIT = DI.
Run.	RU NAME,SA,parameters	None.
Run from specified Disk.	RD DISK,NAME,SA,parameters	None.
Save.	SA UNIT,FWA,LWA	UNIT = DP, FWA = 2, LWA = FWA.
Skip.	SK UNIT,#	UNIT = DI, # = 1.
Save Start address.	SS UNIT,ADDR	UNIT = DP.
Temp. Queue.	TQ UNIT,NAME	None.
Verify.	VE SRCE,DEST,#,MODE	(same as Copy).
Write.	WR DEST,SRCE	DEST = DP, SRCE = CK.
Write EOF.	WE UNIT	UNIT = DP.
Zero System.	ZS	N. A.

The following parameters are used with FDOS Executive commands:

ADDR	= octal address
#	= number of files or sectors to process
DEST	= any functional or physical unit
DISK	= D, E, F, or G (floppy disk drive)
DUNT	= D1, D2, D3, D4, E1,...,G4 (logical units of DISK)
FUNT	= functional unit
FWA	= first word address
HI	= last record or address to be processed
LCTR	= page-spacing line counter
LO	= first record or address to be processed
LWA	= last word address
MODE	= mode of data; A (ASCII), BA (absolute binary), or B (relocatable binary)
NAME	= name of a disk file
PGCTR	= page-spacing character
PUNT	= physical unit
SRCE	= any functional or physical unit
UNIT	= any functional or physical unit

NOTES: Any number input which defaults to octal (disk and core addresses, etc.) can be followed contiguously by "I", which sets bit 15 before further processing. This is useful with the GO command; for example, to transfer indirect to the (core resident) DEBUG breakpoint entry use "GO 4I".
DISK or DUNT values can be referenced implicitly or explicitly via Functional Units, Physical (disk logical) Units, or disk drive letters.

```
00001          SWR    EQU 1
00000          A      EQU 0
00001          B      EQU 1
00200          WSECT  EQU 128
00010          D.D    EQU 10B
00012          D.C    EQU 12B
00014          S.SK   EQU 14B
07600                 ORG CORE-200B
07600 063666   CONFG  LDA .DATL      SET CONFIGURATION POINTER
07601 073664          STA IOPTR
07602 102074          HLT 74B        ENTER DATA CHANNEL
07603 017637          JSB SWCON
07604 102075          HLT 75B        ENTER COMMAND CHANNEL
07605 017637          JSB SWCON
07606 102076          HLT 76B        ENABLE LOADER & SET X7700 IN S
07607 102501          LIA SWR
07610 013655          AND XPAGE      KEEP PAGE
07611 070001          STA B
07612 063656          LDA .STRT      GET CURRENT BOOTLOADER ORIGIN
07613 073657          STA FETCH      SET SOURCE PTR
07614 013655          AND XPAGE      COMPUTE OFFSET
07615 003004          CMA,INA
07616 040001          ADA B
07617 043656          ADA .STRT
07620 073660          STA STORE      SET DESTINATION PTR
07621 002004          INA              AND START
07622 002004          INA
07623 073664          STA IOPTR
07624 067661          LDB LBOOT      USE B AS LOOP INDEX
07625 163657   LOOP   LDA FETCH,I
07626 037657          ISZ FETCH
07627 173660          STA STORE,I
07630 037660          ISZ STORE
07631 006006          INB,SZB        DONE?
07632 027625          JMP LOOP         NO
07633 102077          HLT 77B
07634 063665          LDA SYS0
07635 102601          OTA SWR
07636 127664          JMP IOPTR,I
```

```
07637 000000  SWCON NOP
07640 102501        LIA SWR       GET CHANNEL
07641 013662        AND CHANM
07642 073663        STA CHAN
07643 167664  CONLP LDB IOPTR,I   GET NEXT ADR
07644 037664        ISZ IOPTR
07645 006003        SZB,RSS       DONE?
07646 127637        JMP SWCON,I     YES
07647 063662        LDA CHANM
07650 003000        CMA
07651 110001        AND B,I       CLEAR CHANNEL BITS
07652 033663        IOR CHAN      PUT IN DESIRED CHAN
07653 170001        STA B,I       PUT IT BACK
07654 027643        JMP CONLP
*
07655 076000  XPAGE OCT 76000
07656 007700  .STRT DEF START
07657 000000  FETCH NOP
07660 000000  STORE NOP
07661 177700  LBOOT OCT -100
07662 000077  CHANM OCT 77
07663 000000  CHAN  NOP
07664 000000  IOPTR NOP
07665 000014  SYSO  ABS S.SK
07666 007667  .DATL DEF DATLS
*
07667 007756  DATLS DEF DD.7
07670 007740        DEF DD.8
07671 007742        DEF DD.9
07672 007743        DEF DD.10
07673 000000        OCT 0         TERMINATE
*
07674 007753  CMDLS DEF DC.10     MUST FOLLOW DATLS
07675 007757        DEF DC.14
07676 007760        DEF DC.15
07677 000000        OCT 0         TERMINATE
```

```
  07700                 ORG CORE-100B
* LOAD FILE STARTING AT SECTOR ZERO
  07700 002401  START CLA,RSS
*LOAD FILE STARTING AT BLOCK SPECIFIED IN SWITCH REG
  07701 102501  STAR1 LIA SWR
* LOAD FILE STARTING AT BLOCK SPECIFIED IN A-REGISTER
  07702 007400  STAR2 CCB
  07703 077776        STB BLKNT    INITIALIZE TO READ SECTOR
  07704 063772        IOR DRDCM
  07705 073777        STA NXSEC
  07706 017745  RWCNT JSB RCWRD     READ WORD COUNT
  07707 027735        JMP BOOTE
  07710 002003        SZA,RSS       WORD COUNT = ZERO?
  07711 027706        JMP RWCNT       YES - IGNORE IT
  07712 001727        ALF,ALF       SET COUNTER
  07713 003004        CMA,INA
  07714 073774        STA WCNT
  07715 017745        JSB RCWRD     READ CORE ADDRESS
  07716 102014        HLT 14B       UNEXPECTED EOF
  07717 070001        STA B         START CHECKSUM
  07720 073775        STA PTX       SET POINTER
  07721 017745  NXDAT JSB RCWRD     READ DATA WORD
  07722 102014        HLT 14B       UNEXPECTED EOF
  07723 044000        ADB A         UPDATE CHECKSUM
  07724 173775        STA PTX,I     STORE IT
  07725 037775        ISZ PTX       BUMP POINTER
  07726 037774        ISZ WCNT      READ ALL DATA WORDS?
  07727 027721        JMP NXDAT       NO - READ MORE
  07730 017745        JSB RCWRD       YES - NOW CHECKSUM
  07731 102014        HLT 14B       UNEXPECTED EOF
  07732 050001        CPA B         CHECKSUM ZERO?
  07733 027706        JMP RWCNT       YES - GO ROUND AGAIN
  07734 102012        HLT 12B      CHECKSUM ERROR
  07735 067773  BOOTE LDB HLT77
  07736 024001        JMP B
*WAS
*SUBROUTINE TO INITIATE A SECTOR READ OPERATION
*   (A) = DISK ADDRESS OF SECTOR
*     JSB SECRD
*     ROUTINE PERFORMS THE FOLLOWING SEQUENCE
*      1. WAIT FOR READY
*      2. ISSUE READ COMMAND
*      3. SET BLOCK COUNTER TO SECTOR LENGTH IN WORDS
*      4. READ FILE "NAME" (FIRST WORD OF SECTOR)
*      5. READ NEXT SECTOR ADR. & STORE IT
*      6. DECREMENT BLOCK COUNTER FOR EACH READ
*NOW IN-LINE IN 'RCWRD','DC.10' TO 'REWRD'-1
```

```
*SUBROUTINE TO READ A WORD
*  JSB RWORD
*    1. DECREMENT BLOCK COUNTER
*    2. INPUT WORD
*    3. CHECK STATUS & HALT ON ERROR
*    4. RETURN WITH (A) = WORD. (B) UNCHANGED
  07737 000000  RWORD NOP
  07740 102310  DD.8  SFS D.D         WAIT FOR DATA FLAG
  07741 027740        JMP *-1
  07742 102510  DD.9  LIA D.D         INPUT AND
  07743 103710  DD.10 STC D.D,C         ACKNOWLEDGE DATA
  07744 127737        JMP RWORD,I
*
*SUBROUTINE TO READ A WORD AND CHECK FOR SECTOR END
  07745 000000  RCWRD NOP
  07746 037776        ISZ BLKNT
  07747 027766        JMP REWRD
  07750 063777        LDA NXSEC
  07751 002003        SZA,RSS
  07752 127745        JMP RCWRD,I
  07753 102312  DC.10 SFS D.C        WAIT FOR READY
  07754 027753        JMP *-1
  07755 033772        IOR DRDCM       CREATE READ COMMAND
  07756 103110  DD.7  CLF D.D
  07757 102612  DC.14 OTA D.C
  07760 103712  DC.15 STC D.C,C
  07761 063771        LDA SCLTH
  07762 073776        STA BLKNT       SET UP BLOCK COUNT
  07763 017737        JSB RWORD       READ FILE "NAME"
  07764 017737        JSB RWORD       READ NEXT SECTOR ADDRESS
  07765 073777        STA NXSEC       SAVE SECTOR ADR
  07766 037745  REWRD ISZ RCWRD
  07767 017737        JSB RWORD
  07770 127745        JMP RCWRD,I
  07771 177602  SCLTH ABS -WSECT+2  -(DATA WORDS/SECTOR)
  07772 020000  DRDCM OCT 20000     DISK READ COMMAND
  07773 102077  HLT77 HLT 77B
  07774 000000  WCNT  NOP
  07775 000000  PTX   NOP
  07776 000000  BLKNT NOP
  07777 000000  NXSEC NOP
                      END
** NO ERRORS*
```

```
1071  00000            ORG.2 EQU COMRG-COMSZ-*     IF PAGE TWO TOO LARGE
1072  17512                  ORG COMRG-COMSZ
1073******************** COMMUNICATIONS REGION ************************
1074*
1075*LOCATIONS LOADED WITH DRIVER
1076  17512            COMBG EQU *
1077  17512 016161  P.CIO DEF XIO.0     POINTER FOR TAILORED "SIO"
1078  17513 000000  NLOGU NOP           SET BY CONFIGURATOR
1079  17514 000000  NBUFS NOP           SET BY CONFIGURATOR (FDOS FLAG)
1080  17515 000000  GSFLG NOP
1081  17516 000000  AFLG  NOP
1082  17517 016014  P.XIO DEF XIO
1083  17520 013730  P.PDT DEF T.PDT
1084  17521 013740  P.FDT DEF T.FDT
1085  17522 016017  P.DIO DEF DIO
1086  17523 027547  ELOAD JMP LOAD
1087  17523            COMND EQU *-1
1088*
1089*LOCATIONS NOT LOADED WITH SYSTEM OR LOADER
1090  17524            PARBG EQU *
1091                         REP 8
1092  17524 000000           NOP
1092  17525 000000           NOP
1092  17526 000000           NOP
1092  17527 000000           NOP
1092  17530 000000           NOP
1092  17531 000000           NOP
1092  17532 000000           NOP
1092  17533 000000           NOP
1093  17533            PARND EQU *-1
1094*
1095*LOCATIONS LOADED WITH LOADER
1096  17534            LDRBG EQU *
1097  17534 027544  JSYST JMP SYST
1098  17535 000000  STRTA BSS 1
1099  17536 000000  BTCHF NOP
1100  17537 000000  LERR  NOP
1101*
1102  00000            ORG.3 EQU COMRG-*  IF 'COMSZ' TOO SMALL
1103  00000            ORG.4 EQU *-COMRG  IF 'COMSZ' TOO LARGE
```

```
************************ SYSTEM LOADER ************************
*
* EXEC - LOAD C/FDOS EXECUTIVE
* SYSW - LOAD C/FDOS DRIVERS
* SYST - LOAD C/FDOS SYSTEM
* LOAD - LOAD PROGRAM STARTING AT DISK ADDRESS IN A REGISTER
*  STRTA SPECIFIES STARTING ADDRESS
*  FIRST LOAD ADDRESS IS STORED IN 3 BEFORE LOAD BEGINS
*
 17540 063664  EXEC  LDA EXADR
 17541 002001        RSS
 17542 063665  SYSW  LDA SIADR
 17543 002001        RSS
 17544 063663  SYST  LDA SYADR
 17545 006400        CLB
 17546 077535        STB STRTA    CLEAR START ADDRESS
 17547 073676  LOAD  STA SLCAD
 17550 013667        AND DSMSK
 17551 073671        STA DSKBT
 17552 007400        CCB
 17553 077672        STB STFLG
 17554 077675        STB BLKNT     INITIALIZE WORD READ, NO WORDS
 17555 017625  RWCNT JSB RCWRD     READ WORD COUNT
 17556 027611        JMP STRT      EOF
 17557 002003        SZA,RSS       WORD COUNT = ZERO?
 17560 027555        JMP RWCNT       YES - IGNORE IT
 17561 001727        ALF,ALF       SET COUNR
 17562 003004        CMA,INA
 17563 073673        STA WCNT
 17564 017625        JSB RCWRD     READ CORE ADDRESS
 17565 027606        JMP STR1      UNEXPECTED EOF
 17566 070001        STA B         START CHECKSUM
 17567 037672        ISZ STFLG
 17570 002001        RSS
 17571 070003        STA 3
 17572 073674        STA PTX       SET POINTER
 17573 017625  NXDAT JSB RCWRD     READ DATA WORD
 17574 027606        JMP STR1      UNEXPECTED EOF
 17575 044000        ADB A         UPDATE CHECKSUM
 17576 173674        STA PTX,I     STORE IT
 17577 037674        ISZ PTX       BUMP POINTER
 17600 037673        ISZ WCNT      READ ALL DATA WORDS?
 17601 027573        JMP NXDAT       NO - READ MORE
 17602 017625        JSB RCWRD       YES - NOW CHECKSUM
 17603 027606        JMP STR1      UNEXPECTED EOF
 17604 050001        CPA B         CHECKSUM ZERO?
 17605 027555        JMP RWCNT       YES - GO ROUND AGAIN
 17606 003400  STR1  CCA          ERROR - SET FLAG
 17607 073537        STA LERR
 17610 027544        JMP SYST     LOAD SYSTEM COLD
*
* GOOD LOAD - START PROGRAM
 17611 067535  STRT  LDB STRTA
 17612 073535        STA STRTA    (A CLEAR FROM EOF)
 17613 107700        CLC 0,C
 17614 006002        SZB
 17615 124001        JMP B,I
```

```
17616 124003        JMP 3,I
*SUBROUTINE TO READ A WORD
*  JSB RWORD
*   1. DECREMENT BLOCK COUNTER
*   2. INPUT WORD
*   3. CHECK STATUS & HALT ON ERROR
*   4. RETURN WITH (A) = WORD. (B) UNCHANGED
17617 000000  RWORD NOP
17620 102310  DD.90 SFS D.D        WAIT FOR DATA FLAG
17621 027620        JMP *-1
17622 102510  DD.91 LIA D.D        INPUT AND
17623 103710  DD.92 STC D.D,C        ACKNOWLEDGE DATA
17624 127617        JMP RWORD,I
*
*SUBROUTINE TO READ A WORD , OR IF NEED,
*  START A NEW SECTOR READ OPERATION
17625 000000  RCWRD NOP
17626 037675        ISZ BLKNT
17627 027657        JMP REWRD
17630 063676        LDA NXSEC
17631 002003        SZA,RSS
17632 127625        JMP RCWRD,I
17633 033671        IOR DSKBT
17634 073676        STA SECAD   SAVE SECTOR ADDRESS
17635 102514  DC.90 LIA D.C
17636 013670        AND LRDYS
17637 002003        SZA,RSS
17640 027635        JMP *-3
17641 063662        LDA SCLTH
17642 073675        STA BLKNT    SET UP BLOCK COUNT
17643 063676  NRDY  LDA SECAD
17644 043666        ADA LRDCM    CREATE READ COMMAND
17645 103110  DD.93 CLF D.D
17646 102601  S.2   OTA SWR      DISPLAY COMMAND
17647 102614  DC.91 OTA D.C
17650 103714  DC.92 STC D.C,C
17651 102514  DC.93 LIA D.C      READ STATUS
17652 001210        RAL,SLA      READY?
17653 027643        JMP NRDY       NO
17654 017617        JSB RWORD    READ FILE "NAME"
17655 017617        JSB RWORD    READ NEXT SECTOR ADDRESS
17656 073676        STA NXSEC    SAVE SECTOR ADR
17657 037625  REWRD ISZ RCWRD
17660 017617        JSB RWORD
17661 127625        JMP RCWRD,I
17662 177602  SCLTH ABS -WSECT+2  -(DATA WORDS/SECTOR)
17663 000015  SYADR ABS S.CD     TRACK & SECTOR OF SYST (COLD)
17664 000017  EXADR ABS S.EX     TRACK & SECTOR OF EXEC
17665 000016  SIADR ABS S.SI     TRACK & SECTOR OF SYST (WARM)
17666 020000  LRDCM OCT 20000    DISK READ COMMAND
17666         DRDCM EQU LRDCM
17667 006000  DSMSK OCT 6000     DISK BITS
17667         MDISK EQU DSMSK
17670 040000  LRDYS OCT 40000    FORMATTER READY BIT
17670         RDYST EQU LRDYS
17670         DWRCM EQU RDYST    WRITE COMMAND
17671 000000  DSKBT BSS 1
```

```
17671         DSBUF EQU DSKBT
17672 000000  STFLG BSS 1
17672         DERCM EQU STFLG
17673 000000  WCNT  BSS 1
17673         SCCNT EQU WCNT
17674 000000  PTX   BSS 1
17674         I.BUF EQU PTX
17675 000000  BLKNT BSS 1
17675         RTRYC EQU BLKNT
17676 000000  NXSEC BSS 1
17676         SECAD EQU NXSEC
17676         SCADR EQU SECAD
17617         .I.XX EQU RWORD
*
17677 102077  HALT  HLT 77B        A SYSTEM PRE-BOOTLOADER HALT
*
17677         LDRND EQU *-1
*
00000         ORG.5 EQU CORE+7700B-*  IF LOADER TOO LARGE
00000         ORG.6 EQU *-CORE-7700B  IF LOADER TOO SMALL ('HALT')
```

INTERFACE DESCRIPTION

The model 420 Floppy Disk System (FDS) is interfaced to the Hewlett-Packard 21XX Series computers (2100, 2114, 2115, and 2116) via two identical interface cards (Dicom Part Number 520-50052-02). The interface cards can be installed in any two available I/O channel slots. The highest priority (lowest numbered) channel is used for data transfers; the other channel is used for command and status transfers. The FDS interface supports programmed I/O, interrupt environment, and DMA operation (16 bit).

Since the FDS is a block-synchronous device (data transfers must occur every sixty microseconds once initiated), a high priority should be used for the data channel when operating in the interrupt environment in order to avoid rate errors.

An OTA (or OTB) instruction is used to transfer commands and data to the FDS; an LIA (LIB, MIA, or MIB) instruction is used to transfer status and data from the FDS. After executing the I/O instruction, control must be set and the flag cleared for that channel (select code); thus the proper sequence is OTA followed by STC,C; or LIA followed by STC,C.

The POPIO line generates an initialize to the FDS; thus the FDS is initialized when the PRESET (or EXTERNAL PRESET) push-button is depressed.

Following is a definition of the command and status bit assignments for the FDS.

1. COMMANDS (true when set in A/B before OTA/B)

BIT	NAME	FUNCTION
15	RESET	Initilizes FDS; positions to track zero.
14	WRITE	Select write operation
13	READ	Select read operation
		NOTE: If bits 13 and 14 are both zero upon receipt of STC, no read or write operation will be attempted; this allows selection of a unit for status testing.
12	- - -	Not used

11 and 10	UNIT	Specifies unit number (unit D = ØØ, unit E = Ø1, unit F = 1Ø, unit G = 11)
09 thru 04	TRACK	Specifies track number
03 thru 00	SECTOR	Specifies sector number

STATUS (true when set in A/B after LIA/B)

BIT	NAME	FUNCTION
15	NOT READY (DRIVE)	The selected disk unit is not ready. The unit is not ready if the power is off, if no disk is installed, or if the disk is not up to speed.
14	FORMATTER NOT BUSY (FORMATTER READY)	The FDS is not currently executing a read or write operation, nor is it seeking a new track. The FDS will ignore any commands given when this bit is not set.
13	DATA ERROR	An error (parity error) was detected by the FDS when reading a sector of data. This bit is reset by a new read or write command.
12	WRITE PROTECT	The disk in the selected unit is write protected except when a read operation is in progress (Read and Formatter Busy).
11	RATE ERROR	A data request has not been serviced within the allowable sixty micro-second window.
10	TRACK ZERO	The selected unit is positioned to track zero and the formatter is not busy.
09 thru 00		not used.

INSTALLATION OF HARDWARE

The Model 420 Floppy Disk System is interfaced to the Hewlett Packard 21XX series computers via two identical interface cards (P/N 520-50052-02) and a "Y" cable which interconnects between the interface cards and the baseplane wiring in the FDS enclosure containing drives D and E. In the "standard" software configuration, channel 10 is used for data and channel 14 is used for control/status. Installation of the hardware consists simply of plugging in the interface cards and installing the cable:

1. Unpack the FDS from its shipping box. If shipping damage is noted, notify the carrier (see note in the shipping box).

2.
```
* * * * * * * * C A U T I O N * * * * * * * *
*                                           *
* OPEN THE LOADING DOOR OF EACH DRIVE AND *
* REMOVE THE SHIPPING INSERT BEFORE POWER *
* IS APPLIED.  IF THIS PRECAUTION IS NOT  *
* TAKEN, DAMAGE TO THE DRIVES MAY RESULT! *
*                                           *
* * * * * * * * * * * * * * * * * * * * * * *
```

3. With the computer power off, install the 520-50052-02 interface cards in slots 10 and 11. Install the cable connector labeled "DJ2" onto the card in slot 10; install the cable connector labeled "CJ2" onto the card in slot 11.

4. If a cassette system is to be used with the "standard" FDOS software, install its interface card in slot 14 and cable it up to the cassette system. If a cassette system is not used, install a card in slot 14 to complete the interrupt chain.

5. If the HP-supplied BUF'R'D TTY REG (teletype interface card) is to be used, install it in slot 12; connect the TTY cable to the card. If it is not used, install a card in slot 12 to complete the interrupt chain.

6. If the photoreader is to be supported under the "standard" FDOS software, install its interface card in slot 13 and cable it up. If it is not used, install a card in slot 13 to complete the interrupt chain.

7. Apply power to the computer, TTY, and FDS.

8. Place blank disks into the drives and Bootload the Disk Exerciser tape supplied. Configure, define the parameters, and start the Exerciser per the instructions in Appendix F. The Exerciser should produce a regular incrementing pattern in the Switch Register lights as it writes, reads, and compares; bits 11 through ØØ indicate the last disk address that was accessed (see the INTERFACE DESCRIPTION Appendix of this manual). Setting bit 15 of the Switch Register causes an error summary to be printed on the console device.

9. After the hardware has run successfully for a while, install the software.

```
*INSTRUCTIONS
*  LOAD PROGRAM & START AT 2. PROGRAM COMES TO A SERIES OF HALTS
*  WHICH WAIT FOR SWITCH REGISTER INPUT. SET S APPROPRIATELY & RUN.
*  BEFORE EACH HALT AFTER THE FIRST, A PROMPTER MESSAGE
*  APPEARS ON THE CONSOLE OUTPUT DEVICE TO REMIND
*  THE OPERATOR WHICH PARAMETER IS SOUGHT.
*     HLT 72B:  ENTER CHANNEL OF CONSOLE OUTPUT DEVICE. SET
*               BIT 15 IF INTERFACED THRU A DICOM 344 DECK ZERO.
*     (NOTE: TO USE SOME OTHER OUTPUT DEVICE, SET THE ADDRESS
*            OF THE SIO DRIVER IN 102B
*
*     HLT 73B:  ENTER DISK DATA CHANNEL.
*
*     HLT 74B:  ENTER DISK CONTROL CHANNEL.
*
*     HLT 75B:  ENTER COMPUTER TYPE (FOR TIMING) :
*                0B   2100A
*               14B   2114A , 2115
*               16B   2116
*
*     HLT 76B:  ENTER DRIVES TO TEST/ & PERMANENT OPTIONS
*               BIT     DRIVE
*               0       D
*               1       E
*               2       F
*               3       G
*               SET BIT 15 TO SUPPRESS TESTING OF INTERRUPT
*               SET BIT 14 TO REMAIN IN WRITE PORTION OF CYCLE
*               SET BIT 13 TO REMAIN IN READ PORTION OF CYCLE
*               SET BIT 12 TO REMAIN IN INTERRUPT PORTION OF CYCLE.
*               SET BIT 11 TO RUN THRU THE VARIOUS COMBINATIONS OF
*               HEAD FLOPPING & RANDOM ADDRESSING AUTOMATICALLY.
*
*     HLT 77B:  ENTERING OF PARAMETERS IS COMPLETE. PLACE SCRATCH
*               DISKS INTO THE SPECIFIED DRIVES, SET SWITCHES
*               (SEE NEXT PAGE) AND RUN.
*     (NOTE: TO RESTART WITHOUT CHANGING PARAMETERS, START AT 2000B.
*            TO RESTART AT HLT 76B, START AT 2001B.)
*
*     *** N.B.: EVERY SECTOR OF THE SCRATCH DISKS WILL BE WRITTEN ON
*               DO NOT LEAVE GOOD DISKS IN THE DRIVES.
*  DURING THE COURSE OF OPERATION, SWITH REGISTER BITS 11-0 DISPLAY
*  THE DISK ADDRESS, AS FOLLOWS:
*       BITS 11-10:  DRIVE (CODED AS ABOVE UNDER HLT 76B)
*       BITS  9-4 :  TRACK
*       BITS  3-0 :  SECTOR
*
*  BITS 15-12 ARE USED FOR CONTROL PARAMETERS:
*       BIT 15:  PRINTOUT CONTROL.
*                IF PRINTOUT IS NOT GOING ON, SETTING BIT 15
*                WILL CAUSE AN ERROR SUMMARY TO BE PRINTED
*                (SEE BELOW).
*                IF IT IS GOING ON, SETTING BIT 15 WILL STOP IT.
*                (NOTE: BIT 15 IS AUTOMATICALLY CLEARED AFTER BEING
*                TESTED. ON MACHINES WITH TOGGLE SWITH REGISTERS,
*                THIS MUST BE DONE MANUALLY.)
*
*       BIT 14:  RANDOM ADDRESSING.
*                THIS BIT IS TESTED AT THE START OF EACH PASS. IF
*                SET, A PSEUDO-RANDOM ADDRESSING SCHEME IS USED;
*                OTHERWISE THE PROGRAM PROCEEDS SEQUENTIALLY THRU
*                EVEN SECTORS, THEN THRU ODD.
*
*       BIT 13:  FLOP WRITING.
*                THIS BIT IS TESTED BEFORE EACH WRITE OPERATION
*                IF IT IS SET, THE PROGRAM DELAYS LONG ENOUGH TO
*                CAUSE THE DISK READ/WRITE HEAD TO BE RETRACTED.
*
*       BIT 12:  FLOP READING.
*                THIS BIT IS TESTED BEFORE EACH READ OPERATION.
*                IF IT IS SET, THE PROGRAM DELAYS LONG ENOUGH TO
*                CAUSE THE DISK READ/WRITE HEAD TO BE RETRACTED.
*
*       (NOTE: THE DELAY USED IN FLOP READING OR WRITING IS USED
*              BY THE TIMING SET UP AT HLT 75B OF THE PARAMETER
*              SETTING PROCESS AND BY THE NUMBER OF REVOLUTIONS
*              (THE CONSTANT IN NREV) THE HARDWARE IS EXPECTED
*              TO WAIT BEFORE HEAD RETRACTION.
*  THE EXERCISE CYCLE PROCEEDS AS FOLLOWS:
*     1. A WRITING PASS IS MADE, IN WHICH EITHER RANDOM
*        OR SEQUENTIAL ADDRESSING IS USED TO WRITE ALL THE
*        SECTORS (RANDOM MODE) OR ALL THE EVEN OR ODD SECTORS
*        (SEQUENTIAL MODE) OF EACH OF THE DRIVES BEING EXERCISED.
*
*        THE PATTERN WRITTEN INTO A SECTOR IS COMPOSED OF
*        16 COPIES OF THE FOLLOWING 8 WORDS:
*          WORD 1:  125252B  (CONTENTS OF EXFPT)
*          WORD 2:  DISK ADDRESS (AS IN SWITCH REGISTER DISPLAY)
*          WORD 3:  HIGH ORDER WORD OF DOUBLE PRECISION PASS COUNT.
*          WORD 4:  LO ORDER WORD OF PASS COUNT.
*          WORD 5:  COMPLEMENT OF WORD 1
*          WORD 6:  COMPLEMENT OF WORD 2
*          WORD 7:  COMPLEMENT OF WORD 3
*          WORD 8:  COMPLEMENT OF WORD 4
*
*     2. A READING PASS IS MADE, IN WHICH THE BLOCKS WRITTEN
*        DURING THE WRITING PASS ARE READ AND THEIR CONTENTS COMPARE
*        WITH THE DATA SUPPOSED TO HAVE BEEN WRITTEN. A COMPARE
*        ERROR PRINTOUT OCCURRS IF THEY ARE NOT IDENTICAL.
*
*     3. A TEST OF THE INTERRUPT OPERATION IS PERFORMED. THE TEST
*        SEQUENCE PROCEEDS AS FOLLOWS:
*         A. STC,C ON DATA & CONTROL CHANNELS. THEN CLC 0 AND STF
*            ON DATA & CONTROL CHANNELS. AN INTERRUPT ON EITHER
*            PRODUCES AN ERROR MESSAGE.
*         B. STEP A IS REPEATED WITH CLC ON DATA AND CONTROL
*            CHANNELS REPLACING CLC 0.
*         C. THEN STC & STF ARE PERFORMED ON THE DATA AND CONTROL
*            CHANNELS. FAILURE OF EITHER TO INTERRUPT CAUSES AN
*            ERROR TO BE PRINTED.
*         D. A CLC 0 IS PERFORMED. THEN A READ OF TRACK ZERO, SECTOR
*            ZERO OF THE FIRST DISK UNDER TEST IS INITIATED BY MEANS
*            OF AN STC,C ON THE CONTROL CHANNEL. WITHOUT THE STC
*            ON THE DATA CHANNEL, NO FLAG INTERRUPTS SHOULD OCCUR.
*            THE PROGRAM WAITS ONE SECOND, THEN CHECKS TO SEE THAT
*            THERE WAS A CONTROL INTERRUPT, NO FLAG INTERRUPTS, AND
*            THAT RATE ERROR STATUS WAS SET.
*         E. STEP D IS REPEATED WITH INTERRUPTS ENABLED BY STC,C
*            ON THE DATA CHANNEL. THE PROGRAM WAITS ONE SECOND, THEN
*            CHECKS THAT THERE WERE 128 FLAG INTERRUPTS, A CONTROL
*            INTERRUPT AND NO ERROR STATUS.
```

```
0160*  DURING THE OPERATION OF THE EXERCISER, CERTAIN COUNTS
0161*  ARE MAINTAINED. DURING AN ERROR SUMMARY THEY ARE PRINTED
0162*  OUT:
0163*     1. PASS NUMBER. THIS IS THE NUMBER OF TIMES THE EXERCISE
0164*        CYCLE HAS BEEN STARTED.
0165*     2. NUMBERS OF SECTORS WRITTEN AND READ.
0166*
0167*     3. NUMBERS OF WRITE FLAGS AND READ FLAGS MISSED. THESE
0168*        ERRORS OCCUR WHEN AN EXPECTED READ OR WRITE DATA
0169*        FLAG DOES NOT APPEAR AFTER A SUITABLE WAIT. (FLAGS SHOULD
0170*        COME EVERY 60 MICROSECONDS.)  THESE ARE NOT REPORTED
0171*        UNLESS THE NUMBER IS NON-ZERO.
0172*
0173*     4. NUMBERS OF SOFT AND HARD READ ERRORS. AN ERROR IS
0174*        HARD IF IT PERSISTS THRU THREE TRIES. OTHERWISE IT IS
0175*        SOFT. THESE ARE NOT REPORTED UNLESS THE NUMBER IS
0176*        NON-ZERO.
0177*
0178*     5. NUMBER OF COMPARE ERRORS. THESE ARE NOT REPORTED UNLESS
0179*        THE NUMBER IS NON-ZERO.
0180*
0181*  THE ABOVE ERROR SUMMARY IS PRINTED IN RESPONSE
0182*  TO THE SETTING OF BIT 15 OR AS THE FIRST PART OF THE
0183*  PRINTOUT FOLLOWING A COMPARE ERROR (SEE BELOW).
0184*
0185*  WHENEVER A COMPARE ERROR OCCURS A MESSAGE SPECIFYING
0186*  THE DISK, TRACK AND SECTOR (THE LATTER TWO IN OCTAL) IS
0187*  PRINTED, FOLLOWED BY AN ERROR SUMMARY. THEN THE
0188*  BASIC 8 WORDS OF THE SECTOR PATTERN ARE PRINTED, FOLLOWED
0189*  BY THE ACTUAL 128 WORDS IN THE READ BUFFER.
0190*
0191*     NOTES:
0192*       1. IF FEWER THAN 128 WORDS ARE PRINTED, ALL OF THE
0193*          REMAINING WORDS ARE IDENTICAL TO THE LAST ONE
0194*          PRINTED.
0195*       2. BEFORE READING, THE READ BUFFER IS FILLED WITH
0196*          A BACKGROUND OF 146314B (THE CONTENTS OF CLRPT).
0197*          ITS APPEARANCE IN THE PRINTOUT WOULD INDICATE
0198*          FEWER THAN 128 WORDS READ.
0199*
0200*THE PSEUDO-RANDOM ADDRESSING SCHEME WORKS ACCORDING TO THE
0201*FOLLOWING ALGORITHM:
0202*  AT THE START OF THE PASS, I0 & J0 ARE SET AS FOLLOWS:
0203*     I0 = BITS 5-0 OF PASCT.  J0 = BITS 4-1 OF PASCT.
0204*  A LOOP INDEX WHICH RUNS FROM 0 TO THE NUMBER MINUS 1 OF SECTORS
0205*  CONTAINED ON THE DISKS UNDER TEST IS KEPT.  A VARIABLE "INDEX"
0206*  IS 17 TIMES THE LOOP INDEX, MODULO THE TOTAL NUMBER OF SECTORS.
0207*
0208*  A TABLE OF 64 TRACKS IN "RANDOM" ORDER (TKTBL) AND A TABLE OF 16
0209*  SECTORS IN "RANDOM" ORDER (SCTBL) ARE KEPT.
0210*     TRACK = TKTBL(I0)+TKTBL(I), WHERE I = BITS 9-4 OF INDEX.
0211*    SECTOR = SCTBL(J0)+SCTBL(J), WHERE J = BITS 3-0 OF INDEX.
0212*     DRIVE = DRTBL(K) WHERE K = BITS 11-10 OF INDEX & DRTBL IS A
0213*             LIST OF THE DRIVES UNDER TEST.
```

INSTALLATION OF SOFTWARE

The "standard" FDOS software package is supplied with a fixed configuration of I/O channel assignments and memory size (see Section V of this manual). The software package is supplied on the "master skeleton disk" containing the following files:

SYSØ	LIBE	DUP	D.35
SYS1	LIBF	DEBUG	.IOC.
SYS2	CODSK	EXER	RLOA
SYS3	EDIT	ABCS	SD36X
SYSM	ASMBN	DPCS	ASMBE
SRCE	CROS	D.36	ASMBF
FSRC	FORT	D.36X	FOR2E
LIBN	FOR2N	D.ØØ	

The SYSØ, SYS1, SYS2, SYS3, and SYSM files are discussed in Section V; the EDIT, ASMBN, ASMBE, ASMBF, CROS, DEBUG, DUP, FORT, FOR2N, and FOR2E files are discussed in Section VII; the LIBN, LIBE, LIBF, ABCS, DPCS, D.36, D.36X, D.ØØ, D.35, .IOC., RLOA, and SD36X files are discussed in Section VIII; EXER is discussed in Appendix F. FSRC is the source of a FORTRAN program to be used later in this section; CODSK will also be discussed later in this section.

SRCE is the source of an assembly language demonstration program that flashes the Switch Register lights and rings the console device bell. Upon execution, the program will type

```
COMMAND? TYPE L, R, F, C, B, OR S
```

the following responses are valid:

L (left) - turn on the LSB light of the Switch Register and rotate it left for approximately 30 seconds.

R (right) - turn on the MSB light of the Switch Register and rotate it right for approximately 30 seconds.

F (flash) - flash the Switch Register lights for approximately 30 seconds.

C (count) - do a binary count in the Switch Register for approximately 30 seconds.

B (bells, lights, etc.) - turn on the Switch Register lights one at a time; when they are all on, print an up-arrow and ring the console device bell; turn off the lights one at a time; repeat this cycle five times.

S (system) - go to the core-resident System Loader and reload the operating system.

The following narrative describes the installation of the "standard" FDOS software package, and gives examples of operational procedures. Prior to installing the software, the user should study the preceeding sections of this manual.

A. Install the FDS hardware and run the disk Exerciser (see the INSTALLATION OF HARDWARE Appendix of this manual).

B. Install the "master skeleton disk" supplied in disk drive D (the left-hand drive of the main FDS enclosure).

C. Install the FDOS Bootloader into the protected area of memory; (see Sections V.C and V.D of this manual).

D. Execute the Bootloader at X77ØØB; a Halt 77B indicates a successful load of the System Loader. A Halt 12B or 14B indicates an error was made in loading: verify that the Bootloader was entered correctly and try again; verify that the hardware is working correctly by running the disk Exerciser.

E. Execute the System Loader, thus loading the remainder of the operating system (see Section V.E); this is accomplished by merely pressing RUN. Note that the Switch Register displays the last disk command and address transmitted to the FDS; refer to the INTERFACE DESCRIPTION Appendix of this manual.

F. When the console device bell rings and an "*" is printed, FDOS is "on-the-air"; control of the computer and associated peripherals is under typed commands (refer to Section VI of this manual).

At this point, one should make a copy of the "master skeleton disk" and file the "master skeleton disk" away for back-up purposes. This can be accomplished (on a two-drive system) as follows:

A. Install a disk (to become the "master working disk" in drive E.

B. A disk that is used for the first time in FDOS must be formatted, since its sectors are not set up for use by FDOS (see Format Disk command in Section VI); formatting this disk can be accomplished by typing "FD E". It takes approximately thirty seconds to format a disk.

C. Now copy (and verify) the "master skeleton disk" by typing

```
RU DUP
```

It takes approximately forty-five seconds to copy and verify the entire disk (131K words) using DUP; see Section VII.F for a description of the Disk Utility Package.

D. Remove the "master skeleton disk" from drive D and put it away; take the disk from drive E and install it in drive D. Use the ZS command to clear the buffers.

The above exercise demonstrates calling a program from disk by its name. Remember that all of memory (except the Bootloader and System Loader areas, X7534B through X777B) can be used by the called program, since FDOS is a "disk-resident" operating system.

Now we are going to assemble the SRCE file. Place a virgin disk in drive E and format it.

The FDOS processors (ASMB, EDIT, etc.) use the units referred to in the base page pointers (1Ø1B thru 1Ø4B) for all input and output; these are analogous to DI, PL, DP, and PK (see Sections IV.C.2a, II.D, and II.E). In order for the Assembler to process SRCE, the I/O units must be prepared prior to calling it. Basically, this is done by temporarily assigning the input unit (DI), and "opening" files on the output unit (DP) and high-speed list unit (HL). One might think of this as mounting the paper tape of SRCE in the photoreader, loading the high-speed punch with tape (to accept the binary output), and loading the line printer with paper (to print the listing). Since SRCE is already Queued as the first file on D1, this I/O unit prepration can be accomplished with the following commands (we'll use the names DBIN and LIST for the binary and listing, respectively, of the demo program):

```
AS DI,D1
QU DP,DBIN;RF E,DBIN,2
QU HL,LIST;RF E,LIST,41
```

Files are reseved (RF command) for DBIN and LIST, since the assembly will go much faster if these files already "exist" on the disk; this is because the driver does not have to go back to the Free Storage List (on track zero of the disk) each time a sector is written out. Now call the Assembler by typing

```
RU ASMBN,,K
```

The EAU version (ASMBE) or Floating Point version (ASMBF) of the Assembler could have been used, but SRCE does not use these main-frame options. In the above call, we have directed the Assembler to take its Control Statement from PK. Thus, when it is loaded it will type

```
TYPE ASMB CONTROL STATMENT:
```

To direct the Assembler to generate a binary file on DP and a listing and symbol table on HL, then automatically load the Cross-Reference Table Generator and generate a cross-reference table on HL, type

```
ASMB,A,B,L,T,C
```

A two pass assembly is done automatically on SRCE, generating the binary and listing on disk; this is possible since the computer has complete control of the I/O media and can re-position to the beginning of SRCE (no operator intervention is required as with paper tape).

The absolute binary output of the assembler will be written in DBIN, and the listing, symbol table and cross-reference table will be written in LIST. When the assembly and cross-reference operation is completed, control is retuned to the user at the console device keyboard (CK) via a transfer to X754ØB (see Section V.E).

Note that this entire operation can be accomplished in approxi-mately eighty seconds (by a good typist). To illustrate how FDOS increases the productivity of the computer system, consider the following:

*On a computer system with only a teleprinter (ten character per second I/O), it would take approximately thirty-two minutes just to get a binary paper tape (no listing or cross-reference).

*On a computer system with high-speed paper tape I/O, it would take approximately five minutes just to get the binary paper tape.

*FDOS is thus twenty-four times as productive as the system with just a teleprinter, and 3.75 times as productive as the high-speed paper tape system.

*To actually print the listing, symbole table, and cross-reference table on a teleprinter, it would add approximately 18.5 minutes on all three systems (a line printer would be nice). However, on FDOS we have all this on disk, and can print it later (at night?) when there is less demand on the computer. Also, once initiated, the operation on FDOS does not require operator presence. A note of caution, however; listings (by nature) require a great deal of storage, and may clutter up the disk. One might direct them to cassette or high-speed paper tape. Then(with either media) they could be dumped to a Teletype or line printer at an off-line print station; this would further decrease the demand on the computer.

To execute the program we just assembled, type

```
RD E,DBIN
```

To reload and return to the Executive type an "S".

Now we are going to compile the FSRC file using the FORTRAN Compiler. First, prepare the I/O units (DI is still assigned to D1, since the return from DBIN was to X754ØB, not X7544B - - see Section V.E) by typing

```
QU DP,RBIN;RF E,RBIN,2
QU HL,FLIST
QU SC,TEMP;RF E,TEMP,1Ø
```

We won't reserve a file for FLIST, since only one sector (an EOF) will be written on HL (see P3 parameter description of the FORTRAN Compiler description, Section VII.E). RBIN will be the relocatable binary from pass two, FLIST will be the listing, and TEMP will be the intermediate file. Now call the Non-EAU version of FORTRAN by typing

```
RU FORT,,,2
```

Note that the compiler was directed to take the control statement from FSRC, and to compile the second file on DI. Pass one is made on FSRC, with the intermediate file written in TEMP. FOR2N (pass two) is loaded autmoatically, the intermediate file is processed, and the relocatable binary is written in the RBIN file. Upon completion, the Executive is reloaded via a transfer to the System Loader.

Now to process RBIN with the configured BCS system (ABCS, Absolute Basic Control System file); first, prepare the I/O units by typing

```
RS;QU E1,RBIN
QU E2,ABIN,ELIST;RF E,ABIN,45;RF E,ELIST,4
QU E3,BLIST;RF E,BLIST,5
```

Note that disk logical units are refered to here, rather than functional units (DI, DP, and HL), since we want to insure that we use the proper Unit-Reference numbers (see Section VIII.D). ABIN will be the absolute binary file generated by ABCS, ELIST will be the entry point list, and BLIST will be the bounds list. Call ABCS by typing

```
RU ABCS
```

When the "*OPT" message is printed, type "PL" to direct ABCS to write the absolute binary file (ABIN) followed by an entry point list (ELIST) on the system Output unit (E2); when the "*LOAD" message is printed, type "NL" to cause ABCS to load from the relocatable library (LIBN) and list the bounds (BLIST) to the system list unit (E3); when the "*LST" message is printed, type any character except "L" (see Section VIII.D for a description of ABCS operation). Upon completion, the Executive is reloaded via a transfer to the System Loader. If an entry point or bounds list is desired, these can be obtained from E2 or E3 with the List command.

Now execute the program by typing

```
RD E,ABIN
```

The program will be loaded and executed, and will request you to type the input data. After it has done the computation it will print the result, print another message, and reload the Executive via a transfer to the System Loader.

It should be noted here that FDOS made several transitions between the SIO and BCS environments in this compilation, relocation, subrouting loading, and execution process:

> *After doing the compilation (FORT) in the SIO environment, the BCS environment was entered when ABCS was called; upon completion, the Executive was reloaded (back to SIO).
>
> *When the program was executed (RD E,ABIN) the BCS environment was again entered; upon completion, the Executive was reloaded (back to SIO).

At this point the user should be reasonably familiar with the various aspects of FDOS. If a hardware configuration different from the "standard" as defined in Section V.A is desired, the following procedure can be used to build a "master working disk" (if the "standard" hardware configuration is acceptable skip to the end of this section and delete files as is appropriate). Basically, the procedure consists of using the System Generator to configure and write new "system" files (SYS1, SYS2, SYS3, and SYSM), re-defining the teleprinter channel for DEBUG, building a new ABCS, and deleting un-necessary files from the disk.

PROCEDURE FOR BUILDING A "MASTER WORKING DISK"

```
* * * * * * * * * * * * * * * * * * * * * * * * * * *
* NOTE: DO NOT REMOVE THE WRITE-PROTECT TAB FROM     *
* THE "MASTER SKELETON DISK" SHIPPED FROM THE       *
* FACTORY!                                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * *
```

1. Install the "master skeleton disk" in drive D (the left-hand drive of the main FDS enclosure).

2. Enter the Keyed-In Loader per Section V.C.

3. Load and configure the FDOS Bootloader per Section V.D.

4. Bootload the "skeleton" file (SYSØ) using the X77Ø1B entry of the Bootloader with the Switch Register set to 14B (see Section V.D.2).

5. Remove the "master skeleton disk" from drive D, and install a virgin disk in its place.

6. Halt the computer, turn off all power, configure the interface cards as desired, and turn all power back on.

7. Execute the System Generator at location 2B.

8. Proceed with the System Generation, starting at Section V.F.3; reply "YES" to the "FORMAT?" question in step 21.

9. Install the "master skeleton disk" in drive E.

10. Use the CODSK Batch file to copy all files (except SYS1, SYS2, SYS3, and SYSM) from drive E to drive D; this is accomplished by typing

```
AS BC,E2;BA CODSK
```

Note that CODSK can only be used in conjunction with the "master skeleton disk" (or a copy of it), since it assumes certain named files are on the disk in drive E. Upon entering the Batch mode, CODSK types

```
     *** ****BEGINING OF CODSK BATCH OPERATION
     *AS BM,BK
```

thus assigning the Batch Message unit to the Bit Bucket. Upon completion, CODSK exits the Batch mode (BX), clears the buffers (ZS), restores the original SIO configuration (RS), and types

```
     *** ****END OF CODSK BATCH OPERATION
     *** ****REFER TO APPENDIX G, STEP 1Ø.
```

A listing of CODSK is shown at the end of this appendix.

11. Remove the "master skeleton disk" from drive E, put it in its envelope, and store it per the directions on the envelope.

12. If the console device (teleprinter) is not on channel 12B, re-define its channel for DEBUG per Section VII.D.2.

13. Since the hardware has been re-configured, a new ABCS file must be built. This can be accomplished on the disk in drive D. First, the files that will not be used on the "working disk" (such as CODSK, SRCE, FSRC, LIBE, LIBF, SD36X, ASMBE, ASMBF, and FOR2E in a Non-EAU computer) must be deleted, since there is room for only thirty-one names in a disk directory. This is accomplished via the Delete command. Obtain a Disk Directory, and start deleting files from the bottom of the directory as appropriate.

14. At this point, the user should copy any additional BCS modules (drivers) that are to be used during DPCS onto drive D: Queue each module to a logical unit of D, and then use the Copy command to transfer them to disk.

15. Next, Queue the appropriate BCS modules to D1; D.36 and D.36X should be the first two (if they are to be included), and RLOA should be the last one queued.

16. Now Queue ABCS, assign the I/O units, and call DPCS by typing

```
                    QU D2,ABCS;AS DI,D1;AS DP,D2;RU DPCS
```

The new ABCS will replace the "standard" one; refer to Section VIII.A for details of the Disk Prepare Control System.

17. Since we are now (theoretically) done with the BCS modules, delete them from the disk.

18. Queue the library unit defined during DPCS (the standard is D4) with the appropriate library of relocatable subroutines (LIBN, LIBE, LIBF, or LIBX).

19. Install a virgin disk in drive E, format it, and copy the disk you just built (use DUP to do this).

20. Take one of the disks, write-protect it, label it "master configured disk", put it in its envelope, and store it in a secure place per the directions on the envelope. Thus, when the "working disk" gets misplaced, etc., a copy can easily be made.

21. Repeat Steps 1 thru 4 to configure the Bootloader for the new channel assignments.

If all the above operations work as indicated here, the FDOS may be placed into routine operation.

<u>CODSK Batch File Listing</u>

```
** **** BEGINING OF CODSK BATCH OPERATION
AS BM,BK
LI BC,BK,,12; ** THIS NUMBER MUST EQUAL COMMENT LINES
** BATCH FILE TO COPY ALL DISK FILES FROM
** MASTER SYSTEM DISK TO NEW SYSTEM DISK
** PROGRAMMER: D.K. LA VERNE
** CODING BEGUN: MAY 31, 1974
** PRODUCT ID: 5029 REV. B
**
** EDIT HISTORY
** 31 MAY 74  CREATE THE FILE
** 14 JUN 74  RE-ARRANGE EACH LINE
** 26 AUG 74  ADD INITIAL QUEUE OF DISK LUD'S
** 08 OCT 74  ADD ASSIGNMENT OF DI & DP
**
AS DI,E1;AS DP,D1
SE;TQ DI,SYS0;TQ DP,SYS0
SE;RF DP,SYS0,61;CO ,,,BA;TQ DI,SYS0;TQ DP,SYS0;VE ,,,BA
TQ DI,SRCE;TQ DP,SRCE
RF DP,SRCE,18;CO ,,,A;TQ DI,SRCE;TQ DP,SRCE;VE ,,,A
TQ DI,FSRC;TQ DP,FSRC
RF DP,FSRC,4;CO ,,,A;TQ DI,FSRC;TQ DP,FSRC;VE ,,,A
TQ DI,LIBN;TQ DP,LIBN
RF DP,LIBN,83;CO ,,,B;TQ DI,LIBN;TQ DP,LIBN;VE ,,,B
TQ DI,LIBE;TQ DP,LIBE
RF DP,LIBE,80;CO ,,,B;TQ DI,LIBE;TQ DP,LIBE;VE ,,,B
```

```
TQ DI,LIBF;TQ DP,LIBF
RF DP,LIBF,76;CO ,,,B;TQ DI,LIBF;TQ DP,LIBF;VE ,,,B
TQ DI,CODSK;TQ DP,CODSK
RF DP,CODSK,10;CO ,,,A;TQ DI,CODSK;TQ DP,CODSK;VE ,,,A
TQ DI,EDIT;TQ DP,EDIT
RF DP,EDIT,10;CO ,,,BA;TQ DI,EDIT;TQ DP,EDIT;VE ,,,BA
TQ DI,ASMBN;TQ DP,ASMBN
RF DP,ASMBN,32;CO ,,,BA;TQ DI,ASMBN;TQ DP,ASMBN;VE ,,,BA
TQ DI,CROS;TQ DP,CROS
RF DP,CROS,9;CO ,,,BA;TQ DI,CROS;TQ DP,CROS;VE ,,,BA
TQ DI,FORT;TQ DP,FORT
RF DP,FORT,53;CO ,,,BA;TQ DI,FORT;TQ DP,FORT;VE ,,,BA
TQ DI,FOR2N;TQ DP,FOR2N
RF DP,FOR2N,16;CO ,,,BA;TQ DI,FOR2N;TQ DP,FOR2N;VE ,,,BA
TQ DI,DUP;TQ DP,DUP
RF DP,DUP,4;CO ,,,BA;TQ DI,DUP;TQ DP,DUP;VE ,,,BA
TQ DI,DEBUG;TQ DP,DEBUG
RF DP,DEBUG,5;CO ,,,BA;TQ DI,DEBUG;TQ DP,DEBUG;VE ,,,BA
TQ DI,EXER;TQ DP,EXER
RF DP,EXER,20;CO ,,,BA;TQ DI,EXER;TQ DP,EXER;VE ,,,BA
TQ DI,ABCS;TQ DP,ABCS
RF DP,ABCS,32;CO ,,,BA;TQ DI,ABCS;TQ DP,ABCS;VE ,,,BA
TQ DI,DPCS;TQ DP,DPCS
RF DP,DPCS,14;CO ,,,BA;TQ DI,DPCS;TQ DP,DPCS;VE ,,,BA
TQ DI,D.36;TQ DP,D.36
RF DP,D.36,18;CO ,,,B;TQ DI,D.36;TQ DP,D.36;VE ,,,B
TQ DI,D.36X;TQ DP,D.36X
RF DP,D.36X,1;CO ,,,B;TQ DI,D.36X;TQ DP,D.36X;VE ,,,B
TQ DI,D.00;TQ DP,D.00
RF DP,D.00,6;CO ,,,B;TQ DI,D.00;TQ DP,D.00;VE ,,,B
TQ DI,D.35;TQ DP,D.35
RF DP,D.35,7;CO ,,,B;TQ DI,D.35;TQ DP,D.35;VE ,,,B
TQ DI,.IOC.;TQ DP,.IOC.
RF DP,.IOC.,3;CO ,,,B;TQ DI,.IOC.;TQ DP,.IOC.;VE ,,,B
TQ DI,RLOA;TQ DP,RLOA
RF DP,RLOA,21;CO ,,,B;TQ DI,RLOA;TQ DP,RLOA;VE ,,,B
TQ DI,SD36X;TQ DP,SD36X
RF DP,SD36X,4;CO ,,,A;TQ DI,SD36X;TQ DP,SD36X;VE ,,,A
TQ DI,ASMBE;TQ DP,ASMBE
RF DP,ASMBE,32;CO ,,,BA;TQ DI,ASMBE;TQ DP,ASMBE;VE ,,,BA
TQ DI,ASMBF;TQ DP,ASMBF
RF DP,ASMBF,32;CO ,,,BA;TQ DI,ASMBF;TQ DP,ASMBF;VE ,,,BA
TQ DI,FOR2E;TQ DP,FOR2E
RF DP,FOR2E,16;CO ,,,BA;TQ DI,FOR2E;TQ DP,FOR2E;VE ,,,BA
QU D1,SRCE,FSRC,SD36X;QU D2,CODSK
QU D3,D.36,D.36X,D.00,D.35,.IOC.,RLOA;QU D4,LIBN,LIBE,LIBF
LI BC,CL,,2;RE BC;BX;ZS;RS
  **** END OF CODSK BATCH OPERATION.
  **** REFER TO APPENDIX G, STEP 10.
```