IMSAI CP/M

SYSTEM USER'S GUIDE

Version 1.31, Rev. 2

3/21/77

This manual contains a general introduction to CP/M and a description of the console commands.

81-0000020 Rev. E

TABLE OF CONTENTS

[I] What CP/M is

[II] Copyright Information

[III] Hardware Requirements

[IV] Preface to "Features and Facilities"

[V] An Introduction to CP/M Features and Facilities, by Digital Research*

[VI] IMSAI Notes on An Introduction to CP/M Features and Facilities

[VII] Miscellaneous

A: Peripheral Port Assignments
B: CP/M Automatic Bootstrap Simulator
C: History of Changes to IMSAI CP/M

CP/M Summary

[I.] What CP/M is

CP/M is an advanced floppy disk operating system for the IMSAI 8080 Microcomputer System.

CP/M is an excellent tool for program development. It provides file storage, editing and debugging facilities in addition to an Intel-compatible Assembler and Loader.

CP/M is an excellent operating environment for applications programs requiring named-file, random-access, dynamically allocated floppy disk file storage with files up to 240,000 bytes long.

CP/M provides file storage, input/output, and editing facilities for word processing.

CP/M provides a convenient means to store and retrieve a multitude of applications programs. Just type your program's name and it is loaded and executed.

CP/M has resident drivers for common peripherals and a logical-physical device assignment capability, simplifying the coding of programs.

IMSAI CP/M is supplied on diskette, ready to run on an IMSAI 8080 with 16K of memory, floppy disks and a CRT terminal or teletype. Instructions are included for modifying CP/M to run in other hardware environments and for adding special commands and device drivers.

II. Copyright Information

CP/M is distributed under the condition that the user agrees to the conditions of a license agreement from Digital Research supplied with the CP/M diskette.

READ THIS AGREEMENT BEFORE USING CP/M.

You must return the registration form supplied with this agreement before using CP/M. Those who have returned the form will receive updates and corrections to CP/M.

The exact legal obligations you have under the agreement include putting the following label on any diskette to which you copy CP/M or any of the programs supplied with CP/M.

IMSAI CP/M

VERSION NO. _____

SERIAL NO. _____

III. Hardware Requirements

IMSAI 16K CP/M requires an IMSAI 8080 Microcomputer System configured as follows:

A. 16K of RAM at addresses 0 through 3FFF hex.
B. IMSAI Dual Floppy Disk System on I/O port FD.
C. Terminal on port 2 or 4 (i.e., jumper the IMSAI SIO board for address 0 and use either of its channels).

Optional devices are supported as summarized in Appendix A.

Once the system is up, those familiar with Assembly Language programming can easily add or alter I/O drivers.

The system will run with a single disk drive but no software is supplied for transferring files from one diskette to another.

As distributed, the system best supports a teletype-like terminal on port 2 or a CRT terminal on port 4. The minor differences between ports 2 and 4 are documented in Section VII-F. If both types of terminals are available, connect them both to increase the system's versatility.

If you wish to use additional RAM with CP/M, configure it at consecutive addresses above 3FFF Hex. Get the 16K system up, then use the procedure described in Section VI. 6.97 to relocate the CP/M resident code.

[IV.] Preface to "Features and Facilities"

The Digital Research Manual An Introduction to CP/M Features and Facilities is reproduced starting on the next page. Following the Digital Research Manual are IMSAI's notes on it. These describe a number of differences and enhancements in the IMSAI CP/M system.

DIGITAL RESEARCH
Post Office Box 579
Pacific Grove, California 93950

[V.] AN INTRODUCTION TO CP/M
FEATURES AND FACILITIES

Table of Contents

Section		Page
1.	GENERAL	1
2.	FUNCTIONAL DESCRIPTION OF CP/M	2
	2.1. General Command Structure	3
	2.2. File References	3
3.	SWITCHING DISKS	5
4.	THE FORM OF BUILT-IN COMMANDS	5
	4.1. ERA afn cr	6
	4.2. DIR afn cr	6
	4.3. REN ufn1=ufn2 cr	6
	4.4. SAVE n ufn cr	7
	4.5. TYPE ufn cr	7
5.	LINE EDITING	7
6.	TRANSIENT COMMANDS	7
	6.1. STAT cr	8
	6.2. ASM ufn cr	8
	6.3. LOAD ufn cr	9
	6.4. PIP cr	10
	6.5. ED ufn cr	12
	6.6. SYSGEN cr	13
	6.7. SUBMIT ufn parm#1 ... parm#n cr	14
	6.8. DUMP ufn cr	15
7.	OPERATION OF CP/M ON THE MDS	15

notes at [VI]

An Introduction to CP/M Features and Facilities

1. GENERAL

CP/M is a monitor control program for microcomputer system development which uses IBM-compatible flexible disks for back-up storage. Using a computer mainframe based upon Intel's 8080 microcomputer, CP/M provides a general environment for program construction, storage, and editing, along with assembly and program check-out facilities.

The CP/M monitor provides rapid access to programs through a comprehensive file management package. The file subsystem supports a named file structure, allowing dynamic allocation of file space as well as sequential and random file access. Using this file system, a large number of distinct programs can be stored in both source and machine-executable form.

CP/M also supports a powerful context editor, Intel compatible assembler, and debugger subsystems. When coupled with CP/M's console command processor, the resulting facilities equal or excel similar large computer facilities.

CP/M is logically divided into several distinct parts:

BIOS - the basic I/O system

BDOS - the basic disk operating system

CCP - the console command processor

TPA - the transient program area

The BIOS provides the primitive operations necessary to interface standard peripherals (teletype, CRT, Paper Tape Reader/Punch, and user-defined peripherals), and can be tailored by the user for any particular hardware environment by "patching" this portion of CP/M. The BDOS provides disk management by controlling one or more disk drives containing independent file directories. The BDOS implements disk allocation strategies which provide fully dynamic file construction while minimizing head movement across the disk during access. Any particular file may contain any number of records, not exceeding the size of any single disk (240 records of 128 bytes each). In a standard CP/M system, each disk can contain up to 64 distinct files. The BDOS has entry points which include the following primitive operations:

SEARCH	look for a particular disk file by name
OPEN	open a file for further operations
CLOSE	close a file after processing
RENAME	change the name of a particular file

READ	read a record from a particular file
WRITE	write a record onto the disk
SELECT	select a particular disk drive for further operations

The CCP provides symbolic interface between the user's console and the remainder of the CP/M system. The CCP reads the console device and processes commands which include listing the file directory, printing the contents of files, and controlling the operation of assemblers, editors, and debuggers. The standard commands which are available in the CCP are listed in a following section.

The last segment of CP/M is the area called the TPA. The transient program area holds programs which are loaded from the disk under command by the CCP. During program editing, for example, the TPA holds the CP/M text editor machine code and data areas. Similarly, programs created under CP/M can be checked-out by loading and executing these programs in the TPA.

It should be mentioned that any or all of the CP/M component subsystems can be "overlayed" by an executing program. That is, once a user's program is loaded into the TPA, the CCP, BDOS and BIOS areas can be used as the program's data area. A "bootstrap" loader is programmatically accessible at all times; thus, the user program need only branch to the bootstrap loader at the end of execution, and the complete CP/M monitor is reloaded from disk.

It should be reiterated that the CP/M operating system is partitioned into distinct modules, including the BIOS portion which defines the hardware environment in which CP/M is executing. Thus, the standard system can be easily modified to any nonstandard environment by changing the peripheral drivers to handle the custom system. The standard system is provided with I/O drivers for Intel's MDS microcomputer development system, along with a general discussion of the modification technique.

2. FUNCTIONAL DESCRIPTION OF CP/M.

The user interacts with CP/M primarily through the CCP which reads and interprets commands input through the console. In general, the CCP addresses one of several disks which are online (the standard system addresses up to two different disk drives). These drives are labelled disk "A", "B", and so-forth. A disk is "logged in" if the CCP is currently addressing the disk. In order to clearly indicate which disk is the currently logged disk, the CCP always prompts the operator with the disk name, followed by the symbol ">" indicating that the CCP is ready for another command. Upon initial start up, the CP/M system is brought in from disk A, and the CCP displays the message

xxK CP/M VER m.m

where xx is the memory size (in kilobytes) which this CP/M system manages, and m.m is the CP/M version number. The CCP then automatically logs-in disk A, and prompts the user with the symbol "A>" (indicating that CP/M is currently addressing disk "A") and waits for a command. The commands are implemented at two levels: built-in commands and transient commands.

2.1. GENERAL COMMAND STRUCTURE.

Built-in commands are a part of the CCP program itself, while transient commands are loaded into the TPA from disk and executed. The built-in commands are:

ERA	remove files from the logged disk
DIR	list names of the files on the logged disk
REN	rename the specified file on the logged disk
SAVE	save the specified file on the logged disk
TYPE	type the contents of a file on the logged disk

Nearly all of the commands reference a particular file or group of files. Thus, the form of a file reference is specified below.

2.2. FILE REFERENCES.

A file reference identifies a particular file or group of files on a particular disk attached to CP/M. These file references can be either "unambiguous" or "ambiguous". An unambiguous file reference uniquely identifies a single file, while an ambiguous file reference may be satisfied by a number of different files.

File references consist of two parts: the primary name and the secondary name. Although the secondary name is optional, it usually is generic; that is, the secondary name "ASM" for example, is used to denote that the file is an assembly language source file, while the primary name distinguishes each particular source file. The two names are separated by a "." as shown below:

pppppppp.sss

Where pppppppp represents the primary name of eight characters or less, and sss is the secondary name of no more than three characters. As mentioned above, the name

pppppppp

is also allowed and is equivalent to a secondary name consisting of three

blanks. The characters used in specifying an unambiguous file reference cannot contain any of the special characters

```
. , ; : =  ? *
```

while all alphanumerics and remaining special characters are allowed.

An ambiguous file reference is used for directory search and pattern matching. The form of an ambiguous file reference is similar to an unambiguous reference, except the symbol "?" may be interspersed throughout the primary and secondary names. In various commands throughout CP/M, the "?" symbol indicates that any file name satisfies a match if it matches exactly in all character positions where "?" appears. Thus, the ambiguous reference

```
X?Z.C?M
```

is satisfied by the unambiguous file names

```
XYZ.COM
          and
X3Z.CAM
```

Note that the ambiguous reference

```
*.*
```

is equivalent to the ambiguous file reference

```
????????.???
```

while

```
pppppppp.*
          and
*.sss
```

are abbreviations for

```
pppppppp.???
              and
????????.sss
```

respectively. As an example,

```
DIR *.*
```

is interpreted by the CCP as a command to list the names of all disk files in the directory, while

```
DIR X.Y
```

searches only for a file by the name X.Y Similarly, the command

```
DIR X?Y.C?M
```

causes a search for all (unambiguous) file names on the disk which satisfy this ambiguous reference.

The following file names are valid unambiguous file references:

X	XYZ	GAMMA
X.Y	XYZ.COM	GAMMA.1

3. SWITCHING DISKS.

The operator can switch the currently logged-in disk by typing the disk drive name (A, B, ...) followed by a colon (:) when the CCP is waiting for console input. Thus, the sequence of prompts and commands shown below might occur after the CP/M system is loaded from disk A:

```
16K CP/M VER 1.Ø
A>DIR *.*                 list all files on disk A
SAMPLE   ASM
SAMPLE   PRN
A>B:                      switch to disk B
B>DIR *.ASM               list all "ASM" files on B
DUMP     ASM
FILES    ASM
B>A:                      switch back to A
```

4. THE FORM OF BUILT-IN COMMANDS.

The file and device reference forms described above can now be used to fully specify the structure of the built-in commands. In the description below, assume the following abbreviations:

ufn	-	unambiguous file reference
afn	-	ambiguous file reference
cr	-	carriage return

Further, note that the CCP always translates lower case characters to upper case characters internally. Thus, lower case alphabetics are treated as if they are upper case in command names file references.

4.1 ERA afn cr

The ERA (erase) command removes files from the currently logged-in disk (i.e., the disk name currently prompted by CP/M preceding the ">"). The files which are erased are those which satisfy the ambiguous file reference afn. The following examples illustrate the use of ERA:

ERA X.Y	the file named X.Y on the currently logged disk is removed from the disk directory, and the space is returned
ERA X.*	all files with primary name X are removed from the current disk
ERA *.ASM	all files with secondary name ASM are removed from the current disk
ERA X?Y.C?M	all files on the current disk which satisfy the ambiguous reference X?Y.C?M are deleted

4.2. DIR afn cr

The DIR (directory) command causes the names of all files which satisfy the ambiguous file name afn to be listed at the console device. The command

```
DIR *.*
```

for example, lists the files on the currently logged disk.

Valid DIR commands are:

```
DIR X.Y

DIR X?Z.C?M

DIR ??.Y
```

4.3. REN ufn1=ufn2 cr

The REN (rename) command allows the user to change the names of files on disk. The file satisfying ufn2 is changed to ufn1. The currently logged disk is assumed to contain the file to rename. The CCP also allows the user to type a left-oriented arrow instead of the equal sign, if the user's console supports this graphic character. Examples of the REN command are:

```
REN X.Y=Q.R          The file Q.R is changed to X.Y
```

```
REN XYZ.COM=XYZ.XXX      The file XYZ.XXX is changed to XYZ.COM
```

4.4. SAVE n ufn cr

The SAVE command places n pages (256 byte blocks) onto disk from the TPA and names this file ufn. The machine code file can be subsequently loaded and executed. Examples are:

```
SAVE  3  X.COM

SAVE  40  Q

SAVE  4  X.Y
```

(Note that n is a decimal value).

4.5. TYPE ufn cr

The TYPE command displays the contents of the ASCII source file ufn on the currently logged disk at the console device. Valid TYPE commands are

```
TYPE  X.Y

TYPE  X.C

TYPE  XXX
```

The TYPE command expands tabs (clt-I characters), assumming tab positions are set at every eighth column.

5. LINE EDITING.

The CCP allows certain line editing functions while typing the command.

rubout	delete and echo the last character typed at the console
ctl-U	delete the entire line typed at the console
ctl-E	physical end of line, carriage is returned, but line is not sent until the carriage return key is depressed
ctl-C	CP/M system reboot (warm start)
ctl-Z	end-of-input from the console (used in PIP and ED)

Note that the ctl-x sequence shown above denotes that the control key and the key x are depressed simultaneously.

6. TRANSIENT COMMANDS.

Transient commands are loaded from the system disk and executed in the TPA. The transient commands defined with the CCP are:

STAT	List the number of bytes of storage remaining on the currently logged disk
ASM	load the CP/M macro assembler and assemble the specified program from disk
LOAD	load the file in Intel "hex" machine code format and produce a file in machine executable form which can be loaded into the TPA
DDT	load the CP/M debugger into the TPA and start execution
PIP	load the Peripheral Interchange Program for subsequent media conversion operations
ED	load and execute the CP/M text editor program
SYSGEN	Create a new CP/M system diskette
SUBMIT	Submit a file of commands for batch processing
DUMP	Dump the contents of a file in hex

Transient commands are specified in the same manner as built-in commands, and additional commands can be easily defined by the user. The basic transient commands are listed in detail below.

6.1. STAT cr

The STAT transient command examines the storage map for the currenly logged diskette and prints a message in the format

xxxK BYTES REMAINING

where xxx is the number of kilobytes of storage available

6.2. ASM ufn cr

the ASM command loads and executes the CP/M 8080 assembler. The ufn specified a source file containing assembly language statements where the secondary name is assumed to be ASM, and thus is not specified. The following ASM commands are valid:

ASM X

ASM GAMMA

The two pass assembler is automatically executed. If assembly errors occur during pass 2, the errors are printed at the console.

The assembler produces a file

x.PRN

where x is the primary name specified in the ASM command. The PRN file contains a listing of the source program (with imbedded tab characters if present in the source program), along with the machine code generated for each statement and diagnostic error messages, if any. The PRN file can be listed at the console using the TYPE command, or sent to a peripheral device using PIP (see the PIP command structure below). The file

x.HEX

is also produced which contains 8080 machine language in Intel "hex" format suitable for subsequent loading and execution (see the LOAD command). For complete details of CP/M assembly language program, see the "CP/M Assembler Language (ASM) User's Guide."

6.3. LOAD ufn cr

The LOAD command reads the file ufn, which is assumed to contain "hex" format machine code, and produces a memory image file which can be subsequently executed. The file name ufn is assumed to be of the form

x.HEX

and thus only the name x need be specified in the command. If a file x.HEX does not exist, the LOAD command reads the current RDR: device instead of a disk file. The LOAD command creates a file named

x.COM

which marks it as a machine executable code. The file is actually loaded into memory and executed when the user types the name x immediately after the prompting character ">" printed by the CCP.

In general, the CCP reads the name x following the prompting character and looks for a built-in function name. If no function name is found, the CCP searches the system disk directory for a file by the name

x.COM

If found, the machine code is loaded into the TPA, and the program executes. Thus, the user need only LOAD a hex file once; it can be subsequently executed any number of times by simply typing the primary name. In this way,

the user can "invent" new commands in the CCP. In fact, initialized disks have the transient commands as COM files, and thus can be deleted at the users option.

6.4. PIP cr

PIP is the CP/M Peripheral Interchange Program which implements the basic media conversion operations necessary to load, print, punch, copy, and combine disk files. The PIP program is initiated by typing one of the following forms

```
PIP cr
PIP command-line cr
```

In both cases, PIP is loaded into the TPA and executed. In the first case, PIP reads command lines directly from the console, prompted with the character "*" until an empty command line is typed (i.e., a single carriage return is issued by the operator). Each successive command line causes some media conversion to take place according to the rules shown below. The second form of the PIP command is equivalent to the first, except that the single command line given with the PIP command is automatically executed, and PIP terminates immediately with no further prompting of the console for input command lines. The form of each command line is

```
destination = source#1, source#2, ... , source#n cr
```

where "destination" is the file or peripheral device to receive the data, and "source#1, ..., source#n" represents a series of one or more files or devices which are copied from left to right to the destination. A CP/M end of file mark (ctl-Z) is inserted as the last character if the destination is an ASCII file (all files except ".COM" files are treated as ASCII files in the current CP/M implementation). The equal symbol (=) can be replaced by a left-oriented arrow if your console supports this ASCII character, to improve readability. Lower case ASCII alphabetics are internally translated to upper case to be consistent with CP/M file and device name conventions. Finally, the total command line length cannot exceed 255 characters (the ctl-E control can be used to force a physical carriage return for lines which exceed the console width).

The destination and source elements can be unambiguous references to CP/M source files, with or without a preceding disk drive name. That is, any file can be referenced with a preceding drive name (A:, B:, C:, ...) which defines the particular drive to fetch or store the file. When the drive name is not included, the currently logged disk is assumed. Further, the destination file can also appear as one or more of the source files, in which case the source file is not altered until the entire concatenation is complete. If the destination file already exists, it is removed if the command line is properly formed (it is not removed if an error condition arises). The following command lines (with explanations to the right) are valid as input to PIP

x = y cr	copy to file x from file y, y remains unchanged
x = y,z cr	concatenate files y and z and copy to file x, with y and z unchanged
X.ASM=Y.ASM,Z.ASM,FIN.ASM cr	create the file X.ASM from the concatenation of the Y, Z, and FIN files with type ASM
NEW.ZOT = B:OLD.ZAP cr	move a copy of OLD.ZAP from drive B to the currently logged disk, and name the file NEW.ZOT
B:A.U = B:B.V,A:C.W,D.X cr	concatenate file B.V from drive B with C.W from drive A and D.X from the logged disk, and create the file A.U on drive B

PIP also allows reference to physical and logical devices which are attached to the CP/M system. The device names are three character identifiers, followed by the colon (:) symbol. The device names are

RDR:	Paper tape reader
LST:	Listing device (printer)
PUN:	Paper tape punch
TTY:	Teletype device
CRT:	Cathode ray tube display
ARD:	Addmaster paper tape reader
IRD:	Intel or Icom paper tape reader
PRN:	Tally printer device
CON:	Currently defined console device

The RDR, LST, PUN, and CON devices are all defined within the BIOS portion of CP/M, and thus are easily altered to any particular I/O system. The TTY and CRT devices are present to support the Intel "iobyte" function which allow a simple logical to physical device mapping (see the CP/M Interface Guide for a discussion of the iobyte function). ARD, IRD, and PRN are three popular peripheral devices which have dedicated input/output ports in the CP/M environment. Tab characters (ctl-I) are expanded when the destination device is not the punch. The allowable destination devices are

LST PUN TTY CRT PRN CON

while the allowable source devices are

RDR TTY CRT ARD IRD CON

When devices are used as input, the end of file is indicated by a ctl-Z (the CP/M end of file standard) or, in the case of the ARD and IRD devices, a sequence of 255 rubout characters which is obtained by running the reader with no paper tape.

It should also be noted that PIP performs a special function if the destination is a disk file with type "HEX" (an Intel hex formatted machine code file), and the source is an external peripheral device, such as a paper tape reader. In this case, the PIP program checks to ensure that the source file contains a properly formed hex file, with legal hexadecimal values and checksum records. When an invalid input record is found, PIP reports an error message at the console and waits for corrective action. It is usually sufficient to open the reader and rerun a section of the tape (pull the tape back about 20 inches). When the tape is ready for the re-read, type a single carriage return at the console, and PIP will attempt another read. If the tape position cannot be properly read, simply continue the read (by typing a return following the error message), and enter the record manually with the ED program after the disk file is constructed.

Valid PIP commands are shown below

pip lst: = x.prn cr	copy x.prn to the LST device and terminate the PIP program
pip cr	start PIP for a sequence of commands (PIP prompts with "*")
*con:=x.asm,y.asm,z.asm cr	concatenate three ASM files and copy to the CON device
*x.hex=con:,y.hex,ard: cr	create a HEX file by reading the CON (until a ctl-Z is typed), followed by data from y.hex, followed by data from ARD until a ctl-Z or 255 rubouts are encountered.
*cr	Single carriage return stops PIP

6.5. ED ufn cr

The ED program is the CP/M system context editor, which allows creation and alteration of ASCII files in the CP/M environment. Complete details of operation are given the ED user's manual "ED: a Context Editor for the CP/M Disk System." In general, ED allows the operator to create and operate upon source files which are organized as a sequence of ASCII characters, separated by end of line characters (a carriage return line feed sequence). There is no practical restriction on line length (no single line can exceed the size of the working memory) but is instead defined by the number of characters typed between cr's. The ED program has a number of commands for character string searching, replacement, and insertion, which are useful in the creation and correction of programs or text files under CP/M. Although the CP/M has a limited memory work space area (approximately 6000 characters in a 16K CP/M system), the file size which can be edited is not limited, since data is easily "paged" through this work area.

Upon initiation, ED creates the specified source file if it does not exist, and opens the file for access. The programmer then "appends" data from the source file into the work area, if the source file already exists (see the

-mand) for editing. The appended data can then be displayed, altered, and written from the work area back to the disk (see the W command). Particular points in the program can be automatically paged and located by context (see the N command) allowing easy access to particular portions of a large file.

Given that the operator has typed

ED X.ASM cr

the ED program creates an intermediate work file with the name

X.$$$

to hold the edited data during the ED run. Upon completion of ED, the X.ASM file (original file) is renamed to X.BAK, and the edited work file is renamed to X.ASM. Thus, the X.BAK file contains the original (unedited) file, and the X.ASM file contains the newly edited file. The operator can always return to the previous version of a file by removing the most recent version, and renaming the previous version. Suppose, for example, that the current X.ASM file was improperly edited, the sequence of CCP command shown below would reclaim the backup file

DIR X.*	check to see that BAK file is available
ERA X.ASM	erase most recent version
REN X.ASM=X.BAK	rename the BAK file to ASM

Note that the operator can abort the edit at any point (reboot, power failure, ctl-C, or Q command) without destroying the original file. In this case, the BAK file is not created, although the original file is always intact.

The ED user's manual should be consulted for complete operating details.

6.6. SYSGEN cr

The SYSGEN transient command allows generation of an initialized diskette containing the CP/M operating system. The SYSGEN program prompts the console for commands, with interaction as shown below

SYSGEN cr	initiate the SYSGEN program
*SYSGEN VERSION m.m	SYSGEN signon message
GET SYSTEM? (Y/N)	If a memory image of the CP/M is not present (see CP/M interface guide) type N, otherwise type Y. Normally type Y.
SOURCE ON B THEN TYPE RETURN	Place a diskette containing the CP/M operating system on drive B (it's ok to remove the one that you are using on drive A) and follow with a return when

	ready.
FUNCTION COMPLETE	System is copied to memory
PUT SYSTEM? (Y/N)	If a new diskette is being built, type Y; otherwise type N. Normally type Y.
DESTINATION ON B THEN TYPE RETURN	Place new diskette into drive B, type return when ready.
FUNCTION COMPLETE	New diskette is initialized in drive B

The SYSGEN program then reboots the system from drive A. Upon completion of a successful system generation, the new diskette contains the operating system, and only the built-in commands are available. A factory-fresh IBM-compatible diskette appears to CP/M as a diskette with an empty directory, and thus the operator must copy the appropriate COM files from an existing CP/M diskette to the newly constructed diskette using the PIP transient.

It should be noted that a SYSGEN does not destroy the files which already exist on a diskette; it results only in construction of a new operating system. Further, if a diskette is being used only on drive B, and will never be the source of a bootstrap operation on drive A, the SYSGEN need not take place, and, in fact, a new diskette needs absolutely no initialization to be used with CP/M.

6.7. SUBMIT ufn parm#1 parm#2 ... parm#n cr

The SUBMIT command allows CP/M commands to be batched together for automatic processing. The ufn given in the SUBMIT command must be the filename of a file which exists on the currently logged disk, with an assumed file type of "SUB." The SUB file contains CP/M prototype commands, with possible parameter substitution. The actual parameters parm#1 ... parm#n are substituted into the prototype commands, and, if no errors occur, the file of substituted commands are processed sequentially by CP/M.

The prototype command file is created using the ED program, with interspersed "$" parameters of the form

$1 $2 $3 ... $n

corresponding to the number of actual parameters which will be included when the file is submitted for execution. When the SUBMIT transient is executed, the actual parameters parm#1 ... parm#n are paired with the formal parameters $1 ... $n in the prototype commands. If the number of formal and actual parameters does not correspond, then the submit function is aborted with an error message at the console. The SUBMIT function creates a file of substituted commands with the name

$$$.SUB

on the logged disk. When the system reboots (at the termination of the SUBMIT), this command file is read by the CCP as a source of input, rather than the console. If the SUBMIT function is performed on any disk other than drive A, the commands are not processed until the disk is inserted into drive A, and the system reboots. Further, the user can abort command processing at any time by typing a rubout when the command is read and echoed. In this case, the $$$.SUB file is removed, and the subsequent commands come from the console. Command processing is also aborted if the CCP detects an error in any of the commands. Programs which execute under CP/M can abort processing of command files when error conditions occur by simply erasing any existing $$$.SUB file.

The last command in a SUB file can initiate another SUB file, thus allowing chained batch commands.

Suppose the file ASMBL.SUB exists on disk, and contains the prototype commands

```
ASM $1
DIR $1.*
ERA *.BAK
PIP $2:=$1.PRN
ERA $1.PRN
```

and the command

```
SUBMIT ASMBL X PRN cr
```

is issued by the operator. The SUBMIT program reads the ASMBL.SUB file, and substitutes "X" for all occurrences of $1, and "PRN" for all occurrences of $2, resulting in a $$$.SUB file containing

```
ASM X
DIR X.*
ERA *.BAK
PIP PRN:=X.PRN
ERA X.PRN
```

which are executed in sequence by the CCP.

6.8. DUMP ufn cr

The DUMP program types the contents of the disk file given by ufn at the console in hexadecimal form. The file contents is listed sixteen bytes at a time, with the absolute byte address listed to the left of each line in hexadecimal. Long typeouts can be aborted by pushing the rubout key during printout. (The source listing of the DUMP program is given in the CP/M interface guide, as an example of program written for the CP/M environment.)

7. OPERATION OF CP/M ON THE MDS.

This section gives operating procedures for using CP/M on the Intel MDS microcomputer development system. A basic knowledge of the MDS hardware and software systems is assumed.

CP/M is initiated in easentially the same manner as Intel's ISIS operating system. The disk drives are labelled 0 and 1 on the MDS, corresponding to CP/M's drive A and B, respectively. The CP/M system diskette is inserted into drive 0, and the BOOT and RESET switches are depressed in sequence. The interrupt 2 light should go on at this point. The space bar is then depressed on the device which is to be taken as the system console, and the light should go out (if it does not, then check connections and baud rates). The BOOT switch is then turned off, and the CP/M signon message should appear at the selected console device, followed by the "A>" system prompt. The user can then issue the various resident and transient commands

The CP/M system can be restarted (warm start) at any time by pushing the INT 0 switch on the front panel. The built-in Intel ROM monitor can be initiated by pushing the INT 7 switch, except when operating under DDT, in which case the DDT program gets control instead.

Diskettes can be removed from the drives at any time, and the system can be shut down during operation without affecting data integrity. Note, however, that the user must not remove a diskette and replace it with another without rebooting the system (cold or warm start) unless the inserted diskette is read-only.

Due to hardware hang-ups or malfunctions, CP/M may type the message

PERM ERR DISK x

where x is the drive name which has the permanent error. This error may occur when drive doors are opened and closed randomly, followed by disk operations, or may be due to a diskette, drive, or controller failure. The user can optionally elect to ignore the error by typing a single return at the console. The error may produce a bad data record, requiring re-initialization of up to 128 bytes of data. The operator can reboot the CP/M system and try the operation again.

Termination of a CP/M session requires no special action, although it is best to remove the diskettes before turning the power off, to avoid random transients which could make their way to the drive electronics.

[VI.] IMSAI Notes on An Introduction to CP/M Features and Facilities

This section is intended to be read concurrently with the Digital Research "Features and Facilities" manual. The section numbers represent Digital Research's numbers or insertions between them.

1. General

The BIOS supplied on the distribution diskette has already been modified to support IMSAI peripherals as described above.

A diskette has room for 1898 records, not 240 records as stated by Digital Research.

Programs run in the TPA which begins at 100 hex and ends at 2800 hex (in the current 16K system) for code loaded by the CCP. The area available for data ends at 2800 hex to the top of memory, depending on how much of CP/M it is desired to overlay.

See the CP/M Interface Guide for further information on memory usage.

2.2 File References

Since the Digital Research User's Guides were written, an optional third part has been added to file references in console commands. This is the disk name, "A" or "B". It must precede the rest of the file reference and be separated from it by a colon. Examples:

```
A:X.Y
B:LIST.PRN
A:*.ASM
```

If no disk name is given, the file is searched for or written on the currently logged disk.

Examples of the commands described in sections 4 and 6 using the enhanced file references:

```
ERA B:FOO.*
DIR B:ABC??.COM
SAVE 3 A:X.COM
TYPE B:FOO.PRN
LOAD B:PROG
LIST A:PROG.PRN
```

In other Digital Research Guides, the "primary file name" is referred to as the "file name" and the "secondary file name" is referred to as the "file type".

In general, a file reference can refer to a disk file or files only. The device names CON:, RDR:, etc., are acceptable only in PIP.

2.3 Peripheral Device References

IMSAI CP/M has four logical names which are used in referencing devices other than disk:

CON:	Console (terminal)
RDR:	System (paper tape) reader
PUN:	System (paper tape) punch
LST:	System List Device

Each can be assigned to one of as many as four physical devices as shown in Table VI-A. The initial assignments come from the switches at cold start (they may be changed with the ASSIGN command, as described below). The CON: device is used to input CP/M commands and output responses to them. The LST: device is used by the LIST command, as described below. All can be accessed by PIP and by suitably coded applications programs.

The device names cannot be used in place of disk file names except in PIP.

In addition, the IMSAI PTR-30A Diablo Printer is supported by a driver contained in the DIABLO command. Additional devices can be supported by adding drivers to the basic I/O system (see CP/M System Alteration Guide) or by coding an application program to drive them.

3. Switching Disks

After changing the disk in either of the drives, always reboot (type CTRL C) if you are going to write onto the new disk. Penalty: The old files on the disk may turn out to have their contents replaced by bits and pieces of data from newly-written files if you don't reboot. The only exceptions are the first time a disk is inserted in the unlogged drive since a boot.

4.4 SAVE ufn cr

The area of memory saved runs consecutively upward from the base of the TPA, 100 hex.

TABLE VI-A: IMSAI CP/M I/O DEVICES

Logical Name	Switch and IOBYE Value	Physical Name	Comments
CON:	XXXXXX00	TTY:	Teletype or similar terminal
(CONSOLE)	XXXXXX01	CRT:	Teletype-compatible Cathode Ray Tube Terminal
	XXXXXX10	BATCH:	Batch Mode-Use logical RDR: for input; logical LST: for output**
	XXXXXX11	CON3:	For user-added device*
RDR:	XXXX00XX	TTY:	As above
(READER)	XXXX01XX	RDR:	High Speed Reader*
	XXXX10XX	RDR2:	*
	XXXX11XX	RDR3:	*
PUN:	XX00XXXX	TTY:	As above
(PUNCH)	XX01XXXX	PUN:	High Speed Punch*
	XX10XXXX	PUN2:	*
	XX11XXXX	PUN3:	*
LST:	00XXXXXX	TTY:	As above
(LIST)	01XXXXXX	CRT:	As above
	10XXXXXX	LPT: or PRN:	IMSAI PTR-300 High Speed Line Printer
	11XXXXXX	LST3:	*

* No driver implemented in distribution system. Attempt to input gives control Z's, output goes to the CRT.

** Do not confuse this Batch Mode with the form of batch operation invoked by the SUBMIT command.

4.5 TYPE ufn cr

TYPE can be terminated by typing rubout or any character on the console.

TYPE can be "held" by typing CTRL S, freezing the screen of the CRT. Another CTRL S will continue the output.

5.1 Output Control Characters

The following, when input to the console, modify console output:

CTRL S	Stop output. "Freezes" text on CRT screen for study. Another CTRL S or any character resumes output.
CTRL P	Echo console output to LIST device. Another CTRL P turns this feature off. Handy for getting hard copy of directory listings, etc. when operating on a CRT terminal.

The above characters work for all console I/O, not just CCP commands. Thus, an accidentally typed CTRL S could seem to "freeze" the system. If in doubt, type any character.

6.3 LOAD

The word "load" is used two ways in the CP/M documentation. First, the LOAD command creates a COM file from a HEX file or reader input. It does not actually load RAM. Second, a program is loaded into RAM from a COM file when the file name is typed to the CCP without a preceeding command word, i.e., a program that has been LOADed can be loaded and executed by using its name as a command.

When the name of a COM file is used as a command, the CCP reads the file into the TPA consecutively upward from 100 hex, then begins execution at 100 hex.

COM files are created either with LOAD or with SAVE. The latter case is common after using DDT. Note that if you wish to combine multiple HEX files into one COM file, or to load with offset, you must use DDT rather than LOAD.

6.4 PIP

The following device names correspond to the logical devices described in Table VI-A:

CON: RDR: PUN: LST:

The others are physical devices and are not supported directly by PIP in the IMSAI initial release. Those for which BIOS contains drivers, namely

CRT: TTY: PRN:

can be accessed by ASSIGNing them to a logical device then using the logical device name in PIP.

The current version of PIP supplies line numbers when the destination is LST:.

6.6 SYSGEN

IMSAI SYSGEN gets the system from drive A. Its operation is identical to Digital Research's except the prompt "SOURCE ON B" is replaced by "SOURCE ON A".

6.7 SUBMIT

The SUB file containing the commands to be SUBMITted can be created using ED or with PIP name.SUB=CON:. In the latter case, no correction of typing errors is possible and you must enter line feeds after carriage returns.

The file GEN.SUB on the distribution diskette contains a set of commands for initializing a new diskette.

6.9 Additional Transient Commands in IMSAI CP/M

6.91 ASSIGN cr and
ASSIGN logical device = physical device cr

ASSIGN is used to change the physical device assigned to the CP/M logical devices. See Section VI. 2.3 for a description of the device assignment scheme.

ASSIGN cr updates the assignments from the switches. Each logical device's assignment is controlled by two switches as shown in Table VI-A.

ASSIGN logical device = physical device changes the assignment of one logical device by name. The device names are given in Table VI-A.

Examples:

```
ASSIGN cr
ASSIGN LST:=PRN: cr
ASSIGN CON:=CRT: cr
ASSIGN CON:=TTY: cr
```

The last two examples change the console device used for the next and following CP/M commands. Thus, if you have both a teletype and a CRT, you may switch between them at will, depending on whether you want fast output or hard copy. The teletype can also function as READER, PUNCH and/or LIST devices regardless of the device in use for CONSOLE.

The source code for the ASSIGN transient is included on the distribution diskette. it is table-driven and can easily have names added to reflect drivers which have been added to the system. The file should be assembled, listed and comments read for further information.

6.92 LIST ufn (heading) cr

The LIST command copies a disk file to the current LST: device. A heading including the file name, any text typed after the file name and the page number is printed at the top of each page. LIST feeds to the next page whenever a form feed (CTRL L) character* is encountered in the file or after 60 lines have been printed.

Examples:

```
LIST X.ASM
LIST LIST.PRN 10/23/76
```

The latter lists the assembly listing produced by assembling the file LSIT.ASM and includes the given date at the top of each page.

A LIST in progress can be interrupted by typing rubout or any character on the console.

* LIST will also treat a CTRL K character as a form feed. This permits you to control the page breaks in files prepared with ED, which will not put CTRL L's into the file. Put a ";" before the CTRL K if the file is to be ASMbled.

6.93 DIABLO ufn cr

DIABLO copies the given file to an IMSAI PRT-30 Diablo Printer. In other respects, DIABLO is identical to LIST, as described above.

Because of the amount of code involved, the Diablo Printer driver is not incorporated in the resident system but is built into the DIABLO transient. Those who want the driver for inclusion in another program can extract it from DIABLO.ASM on the distribution diskette.

6.94 READ ufn cr

READ inputs paper tape from the currently assigned RDR: device to the named disk file. READ was implemented because the current version of PIP sometimes looses characters when inputting paper tape from a teletype.

End-of-file on the paper tape is indicated by either a CTRL-Z character (the CP/M end-of-file standard), a CTRL-A character (as used on tapes generated by IMSAI's Self-Contained System), or by a sequence of 255 or more rubout characters. If the tape being input had none of the above terminators, there are two other ways to signal READ that the end-of-file has been reached. First, unless the CON: and RDR: logical devices are both assigned to TTY:, type a rubout at the console. (If the paper tape has run out or stopped, back it up and input one more character.) READ will then close the disk file, using all characters input before the rubout was typed. Second, when READing from a teletype, stopping the tape and typing a CTRL-Z will terminate the READ operation.

READ is designed to work with ASCII source files and HEX format object files. Punching and READing .COM files is not recommended as the data in the file could look like and end-of-file indicator to READ.

6.95 FORMAT cr

FORMAT is used to write an IBM-compatible format on a diskette. FORMATting of brand new, approved diskettes is generally not necessary. Use FORMAT if the format has been damaged (i.e., by powering down with a disk in the drive, opening the drive door at the wrong time, etc.) or if there are any unusual problems reading or writing on a diskette.

Since FORMAT destroys all information on a diskette, FORMAT makes an inquiry to verify that the user really does want to re-format. If so, FORMAT asks for the drive name (A orB) and final confirmation. The interaction is self-explanatory.

6.96 SHAKDOWN cr

SHAKDOWN is a floppy disk and memory testing program. In addition to detecting problems in the disk drives and interface, SHAKDOWN has shown itself to be very good at detecting memory defects.

If your system is functioning well, SHAKDOWN will periodically type a message of the form

Ø ERRORS IN nnn OPERATIONS

For detailed information on SHAKDOWN and its error messages, see the SHAKDOWN User Guide.

SHAKDOWN does not test RAM below approximately 17ØØ hex; to test all of your boards, reconfigure your memory and run SHAKDOWN again.

6.97 CPM n *

Typing CPM n *, where n is a decimal number between 16 and 64, creates a CPM system relocated to run in nK of RAM. The new system image is left in the TPA, ready for writing onto a diskette with SYSGEN.

For example, to create a 24K CP/M system, the interaction would be as follows (user typing underlined, plus carriage returns at the end of each command).

```
A>CPM 24 *                          Command given by user

CONSTRUCTING 24K CP/M VERS 1.31
READY FOR "SYSGEN OR                Messages typed by CPM
"SAVE 32 CPM24.COM"

A>SYSGEN                            Command typed by user

GET SYSTEM (Y-N) N
PUT SYSTEM (Y-N) Y                  Tell SYSGEN to write
DESTINATION OF B, TYPE RETURN_      system
FUNCTION COMPLETE

A>
```

At this point, the diskette in drive B has the new system. If not done previously, the COM files that implement the transient commands must be put on the diskette. If you do this with SUBMIT GEN as described in Section VI-E, answer "N" to SYSGEN's questions.

To run the new system, move the diskette to drive A and type control-C.

If you type CPM * * instead of CPM n *, the CPM command will determine how much RAM is in your machine and generate a system of appropriate size.

If the * as the second argument is omitted, the relocated system is placed at the memory addresses at which it will run.

6.98 BASIC-E ufn cr
RUN-E ufn cr

These respectively compile and execute a program written in the BASIC-E language. See the BASIC-E User's Guide for details.

7. CP/M Operation on the IMSAI 8080

Before beginning work with your new CP/M, we recommend you initialize one or more additional diskettes from your distribution diskette, and work with one of the new diskettes.

A. Cold Start*

1. Turn on System power.
2. Insert CP/M diskette in Drive 0, close door, wait for READY light.
3. Set address switches to zero, hit STOP, RESET, EXAMINE.
4. Set PROGRAMMED INPUT switches to I/O device assignments, in particular
   switches 0 and 1 down for terminal on port 2
   switch 0 down, switch 1 up for terminal on port 4
5. Hit RUN.
6. System should type sign on message and "A>"

* NOTE: If your floppy disk interface still contains firmware PROM's of revision 2 or older, refer to Appendix B for information simulating the automatic bootstrap.

B. Restart

1. If the system is running, typing CTRL-C to the console command processor, ED, DDT or PIP will reboot the system and enter the console command processor.
2. If the system or program is hung up, follow step A-1 through A-5 above. If the RUN light does not go out when STOP is depressed, raise RESET and depress STOP simultaneously.

C. Terminating a Session

Remove the diskette(s) from the drives and store them safely. NOTE: Powering down the mainframe and/or the disk drives while the diskettes are in place usually destroys the information on the diskette.

[VII.] Miscellaneous

A. Initializing a Diskette for use with CP/M

A virgin diskette that is correctly formatted may be inserted in Drive 1 and used for file storage with no further preparation. However, it is generally desired to keep at least some of the transient command programs (COM files) on each diskette. Note that there must be a diskette containing a system image in Drive Ø whenever ED, ASM, DDT and other programs terminate.

The system image may be copied from one diskette to another with SYSGEN; the COM files may be individually copied with PIP. If the system image and all of the COM files are desired, the process may be automated with the command SUBMIT GEN.

B. The Distribution Diskette

The files on the diskette include:

PIP, COM, ED.COM, ASM.COM, DDT.COM, LOAD.COM, LIST.COM, STAT.COM, etc.	The COM files are the programs which implement the transient commands.
BIOS.ASM BOOT.ASM	Source code of IMSAI-supplied portions of CP/M system. Listings are given in the CP/M System Alteration Guide.
LIST.ASM DIABLO.ASM FORMAT.ASM	These contain the assembly language source code for some of the IMSAI-supplied commands. You may list them for your information and modify programs to suit your purposes. Also, they are valuable as examples of code written to interface with CP/M.
GEN.SUB	File to SUBMIT to initialize a new CP/M diskette (does not copy .ASM files).

Use the DIR command to obtain a complete list of files on the diskette.

APPENDIX A: Peripheral Port Assignments

Device	Interface	I/O Port
Teletype or similar terminal (Note 1)	SIO or MIO	2 Status 3
Cathode Ray Tube terminal (Note 1)	SIO	4 Status 5
IMSAI Dual Floppy Disk Drive	FIF	FD
IMSAI PTR-300 Line Printer	LIF	F6
Diablo Printer IMSAI PTR-30*	PIO	FA Jumpers
Priority Interrupt Control Board**	PIC-8	F7

Note 1: Drivers for these ports are identical except for minor differences described in the CP/M System Alteration Guide; thus, for example, a CRT terminal may be used on port 2.

* Connect printer with IMSAI cable as specified in documentation supplied with printer. Printer is supported only by Diablo transient.

** Initialized for user's convenience but not used.

APPENDIX B

CP/M Automatic Bootstrap Simulators for
Use with Old Floppy Disk Interfaces

CP/M as distributed is intended to work with the automatic bootstrap feature which is being added to the IMSAI Floppy Disk interface. Arrangements are being made to distribute revised firmware including this feature at approximately the same time as CP/M is distributed.

If your new FIF PROM's (rev. 3 or greater) are installed by the time you wish to bring CP/M up, use the operating procedures given in the IMSAI CP/M User's Guide and disregard the rest of this document.

If you wish to use CP/M before your new firmware is in use, you will need some way to simulate the automatic bootstrap, i.e., to read drive Ø, track Ø sector 1 into RAM at Ø, then transfer control to it.

Listings of two programs for this purpose are attached. The first, 26 bytes long, is suggested if you must toggle it in. The second, slightly longer, retries the read if an error occurs and is coded in such a way that it will operate in PROM as well as RAM. This version is suggested if you have any way of making it resident.

Operation is most convenient with the program in PROM. The next best choice is RAM above 4ØØØ hex. If you have no extra RAM, relocate the program to any location at or above 8Ø hex and enter it each time you need to cold-start CP/M.

Once running, CP/M can reboot itself without using this program. NOTE: These programs are distinct from the "Bootstrap" which resides on the first two disk sectors of any CP/M disk. The function of these programs is to read in and start the bootstrap or the disk.

```
                ;EZBOOTS.ASM  VERSION 1.0  10/27/76 JRB

                ;AUTOMATIC BOOTSTRAP SIMULATOR FOR STARTING CP/M
                ; IN SYSTEMS IN WHICH AUTOMATIC BOOTSTRAP DISK
                ; INTERFACE FIRMWARE HAS NOT YET BEEN INSTALLED.

                ;NOT NEEDED ONCE NEW PROM'S ARE INSTALLED IN FIF.

                ;THIS IS THE SHORT VERSION, SUITABLE FOR TOGGLING IN.
                ;A SEPERATE VERSION, WITH ERROR RETRIES, IS
                ; RECOMMENDED IF YOU HAVE A WAY OF KEEPING IT RESIDENT.

4000 =          BBASE   EQU  4000H      ;YOU MAY CHANGE THIS TO ANY VALUE
                                        ;GREATER THAT 7FH. IF BELOW 4000H
                                        ;PROGRAM MUST BE RELOADED EACH
                                        ;TIME YOU WANT TO BOOT CP/M.
0000 =          EXIT    EQU  0          ;WHERE THIS PROGRAM EXITS TO
00FD =          DISK    EQU  0FDH       ;FLOPPY DISK COMMAND OUTPUT PORT

4000                    ORG  BBASE      ;THIS IS ALSO START ADDRESS
                ;SET DISK INTERFACE STRING POINTER
4000 3E10               MVI  A,10H      ;"SET STRING POINTER 0" COMMAND
4002 D3FD               OUT  DISK       ;SEND COMMAND TO DISK INTERFACE
4004 3E1F               MVI  A,BCMD AND 0FFH    ;LO ORDER OF STRING LOC
4006 D3FD               OUT  DISK       ;SEND TO DISK
4008 3E40               MVI  A,BCMD SHR 8 AND 0FFH ;HI ORDER HALF OF SAME
400A D3FD               OUT  DISK
                ;READ SECTOR. ONE TRY ONLY.
400C 212040             LXI  H,BSTAT    ;POINT AT STATUS BYTE OF STRING
400F AF                 XRA  A          ;GET ZERO IN A
4010 77                 MOV  M,A        ;ZERO STATUS BYTE
                        ;N. B. A=0 IS ALSO DISK COMMAND TO DO STRING 0
4011 D3FD               OUT  DISK       ;DO IT!
4013 86         WAIT:   ADD  M          ;LOOK FOR NON-0 STATUS
4014 CA1340             JZ   WAIT       ;KEEP LOOKING TILL IT COMES
                ;ALL THAT IS ABSOLUTELY ESSENTIAL AT THIS POINT IS TO JMP 0.
4017 FE01       HANG:   CPI  1          ;THIS CODE CAUSES A HANG HERE IF
4019 C21740             JNZ   HANG      ;..DISK READ FAILED.
401C C30000             JMP  EXIT       ;READ WAS OK, GO TO ROUTINE READ IN.
                ;PRE-INITIALIZED COMMAND STRING FOR DISK (MUST BE IN RAM)
401F 21         BCMD:   DB   21H        ;COMMAND BYTE: READ, UNIT 1
4020 00         BSTAT:  DB   0          ;STATUS BYTE, SET BY DISK WHEN DONE
4021 0000               DW   0          ;TRACK 0 (2 BYTES)
4023 01                 DB   1          ;SECTOR 1
4024 0000               DW   0          ;BBUFFER ADDRESS: READ TO LOCATION 0

4026                    END  BBASE
```

```
                ;ABOOTSIM.ASM   VERSION 1.0  10/27/76  JRB

                ; IMSAI CP/M SIMULATOR FOR AUTOMATIC BOOTSTRAP

                ;THIS PROGRAM IS ONLY NEEDED TO START CP/M COLD
                ; IF YOU DO NOT YET HAVE THE NEW AUTOMATIC BOOTSTRAP
                ; FIRMWARE INSTALLED IN YOUR DISC INTERFACE.

                ;THE FUNCTION OF THIS PROGRAM IS TO READ DRIVE 0,
                ; TRACK 0, SECTOR 1 INTO RAM LOCATIONS
                ; 0 TO 7FH THEN JMP TO 0.

                ;THIS IS THE FANCY VERSION,
                ; SUITABLE FOR USE IN ROM AND WITH ERROR RECOVERY.

4000 =          BBASE   EQU  4000H      ;ORIGIN AND STARTING ADDRESS

0000 =          EXIT    EQU  0          ;WHERE THIS PROGRAM EXITS TO

00FD =          DISK    EQU  0FDH       ;COMMAND PORT FOR FLOPPY DISC

0080 =          BCMD    EQU  80H        ;BEGINNING OF DISC COMMAND STRING
                                        ;AND COMMAND BYTE THEREOF
0081 =          BSTAT   EQU  BCMD+1     ;STATUS BYTE...
0082 =          BTRK    EQU  BCMD+2     ;TRACK (2 BYTES)
0084 =          BSECT   EQU  BCMD+4     ;SECTOR
0085 =          BBUFAD  EQU  BCMD+5     ;BUFFER ADDRESS

4000                    ORG  BBASE
                ;
                ; START HERE
                ;
                ; SET FLOPPY DISC INTERFACE STRING POINTER
4000 3E10               MVI  A,10H      ;COMMAND TO SET STRING POINTER 0
4002 D3FD               OUT  DISK       ;SEND IT
4004 218000             LXI  H,BCMD     ;POINT H AT COMMAND STRING
4007 7D                 MOV  A,L        ;LO ORDER STRING ADDRESS
4008 D3FD               OUT  DISK
400A 7C                 MOV  A,H        ;HI ORDER STRING ADDRESS
400B D3FD               OUT DISK
```

```
                ; ABOOTSIM.ASM  CONTINUED

                ;SET UP STRING IN RAM
400D 3621               MVI   M,21H       ;COMMAND TO READ SECTOR, UNIT 0
400F 23                 INX   H           ;POINT AT STATUS BYTE
                              ;('STATUS BYTE WILL BE ZEROED LATER)
4010 23                 INX   H           ;POINT AT HI ORDER TRACK
4011 AF                 XRA   A           ;GET 0 IN A
4012 77                 MOV   M,A         ;ZERO HIGH ORDER TRACK
4013 23                 INX   H
4014 77                 MOV   M,A         ;ZERO LO ORDER TRACK
4015 23                 INX   H
4016 3601               MVI   M,1         ;SECTOR 1
4018 23                 INX   H
4019 77                 MOV   M,A         ;ZERO LO ORDER BUFFER ADDRESS
401A 23                 INX   H
401B 77                 MOV   M,A         ;ZERO HIGH ORDER BUF ADDRESS
                ;
                ; INITIALIZATION COMPLETE. NOW READ SECTOR.
                ;
401C 218100     REPEAT: LXI   H,BSTAT
401F AF                 XRA   A
4020 77                 MOV   M,A         ;STATUS MUST BE 0 BEFORE COMMAND
                        ;N. B. A=0 IS DISK COMMAND TO EXECUTE STRING 0
4021 D3FD               OUT   DISK        ;TELL DISK TO GO!
4023 86         WAIT:   ADD   M           ;LOOK FOR NON-0 STATUS
4024 CA2340             JZ    WAIT        ;KEEP LOOKING TILL IT COMES
4027 FE01               CPI   1           ;ONLY CORRECT RETURN VALUE IS 1
4029 CA0000             JZ    EXIT        ;ON SUCCESS, TRANSFER CONTROL
                              ;CP/M BOOTSTRAP ROUTINE READ IN FROM DISK
                ;
                ; DISK ERROR HAS OCCURED.
                ; DISPLAY ERROR CODE IN LIGHTS AND KEEP TRYING.
                ; THUS IF DISK IS NOT READY,
                ; CP/M WILL BOOT AS SOON AS IT COMES READY.
                ;
402C 2F                 CMA               ;LIGHTS DISPLAY. COMPLEMENT
402D D3FF               OUT   0FFH        ;OUTPUT A TO "PROG OUTPUT" LIGHTS
402F 3E21               MVI   A,21H       ;RESTORE DRIVE 1 - IT HELPS!
4031 D3FD               OUT   DISK        ;SEND RESTORE COMMAND TO DISK
4033 C31C40             JMP   REPEAT      ;TRY AGAIN (AND AGAIN, AND ...)

4036                    END   BBASE
```

SUMMARY

Basic Commands:

DIR (afn) cr	list file names that match afn
ERA afn cr	erase files that match afn
REN ufn1=ufn2	rename file2 as file1
SAVE n ufn cr	write n_{10} 256 byte blocks in file
TYPE ufn cr	output file to console
A:, B:	change active disk
ufn cr	load .COM file and excute

Transient Commands: (Standard Utility Programs)

STAT cr	Type bytes free storage on disk
ED ufn cr	Initiate EDITOR program with file
ASM ufn cr	Assemble .ASM file to .PRN and .HEX
LOAD ufn cr	Create .COM file from .HEX
PIP ufn1=ufn2,ufn3,...	Move and/or concatenate files
SUBMIT ufn1 parm1,... parmN	Accept commands/parameters from .SUB file
DDT (ufn) cr	Initiate debugger program
DUMP ufn	Type file in hex on console
SYSGEN cr	Copy CP/M system to another dikdette
CPM n * cr	Relocate system to use n_{10}K memory
ASSIGN log. device= phy. device cr	Change physical I/O devices
FORMAT cr	Re-initialize diskette (IBM format)
SHAKDOWN cr	Initiate memory and disk test program
LIST ufn cr	Output file to system list device
DIABLO ufn cr	Output file to PTR-30 (Diablo Hy-Type)
BASIC-E ufn cr	Compile .BAS file to .INT
RUN-E ufn cr	Run .INT file
READ ufn cr	Input paper tape from system reader to disk file

Console Input Special Characters:

rubout	delete last character and echo
CTRL-U	delete last line
CTRL-E	physical end of line
CTRL-C	reboot
CTRL-Z	end of console input (ED)
CTRL-P	send console output to LST:
CTRL-S	freeze screen (stop output)
cr	(carriage return) terminates command

Miscellaneous

>	CP/M prompt
=	replaceable with ←
ufn	unambiguous file name: [<disk unit>:]<file name>[.<file type>]
afn	ambiguous file name
	? always matches character
	* always matches name or type

disk unit	A:, B:
file name	up to 8 alphanumeric characters
file type	up to 3 alphanumeric characters

Standard file types:

.COM	Command file - object for 100H
.HEX	Object file - Intel Hex format
.ASM	Assembly source file
.PRN	Assembly listing file
.BAK	Back-up source file
.$$$	Temporary work file
.SUB	Submit file
.BAS	Basic source file
.INT	Compiled BASIC-E file

Logical Devices:

CON:	Console
RDR:	System reader (paper tape)
PUN:	System punch (paper tape)
LST:	System list device

Physical Devices:

TTY:	Teletype (port 2)
CRT:	Video terminal - ADM-3 (port 4)
LPT:	IMSAI PTR-300 line printer

APPENDIX C: History of Changes to IMSAI CP/M

Version 1.3Ø, Revision Ø; 11/1/76

Original Release

Version 1.31, Revision Ø; 12/15/76

CPM	Command to relocate resident system
BIOS	Enlarged by 1ØØ bytes; "CRASH" error message added
SYSGEN 1.31	Bases system in TPA at 7ØØ hex; Transfers two more sectors
DDT 1.3	Relocates self for size of running system
MEMTEST 1.3	More switch options; Bug relating to more than 32K of memory fixed
READ Ø.91	Bug that occurred when COM: and RDR: were both assigned to TTY: fixed

Version 1.31, Revision 1; 1/27/77

BASIC-E	Files BASIC-E.COM, RUN-E.COM added to distribution diskette; Documentation added to manual
GEN.SUB	Updated to include above files
LIST 1.3	Bug fix: set bit 7 of characters to Ø before sending to printer

Version 1.31, Revision 2; 3/21/77

SHAKDOWN	SHAKDOWN.COM file added to distribution diskette SHAKDOWN User's Guide added to documentation
MEMTEST	Removed from documentation and from distribution diskette
BASIC-E	Bug fixes: VAL function; compiling files over 16K long
Documentation	Reorganized into separately-bound manuals