COMPUTER MART OF NEW JERSEY, INC.
501 ROUTE 27 • ISELIN, NEW JERSEY 08830 • PHONE 201/283-0600

ICOM - CP/M SYSTEM UPGRADE

By Bruce Ratoff
and
Computer Mart of New Jersey's
Software Division

SOFTWARE AGREEMENT

Introduction

The programming effort which results in a complete CP/M system comes from two sources: Digital Research (the inventors of CP/M) and your computer store. CP/M was designed to operate with any brand of floppy disc drives utilizing what is commonly referred to as 'IBM format'. Therefore, the specific routines necessary to control a particular brand of disc hardware are isolated from the rest of the operating system. Provisions have been made in the design of CP/M for the addition of these routines. Your System Upgrade release disc contains a copy of CP/M to which have been added routines that control the iCOM Frugal Floppy, resulting in a complete and very powerful disc operating system. These routines are known in CP/M parlance as the Customized Basic Input/Output System, henceforth known as CBIOS.

CBIOS also contains, in addition to the disc routines, routines to handle a terminal, a reader, a punch, and a listing device. On the disc supplied, where possible these have been set to be the terminal I/O routines contained in the PROM on your iCOM floppy disc interface card. The remaining routines are contained within CBIOS itself. For example, one routine tests whether a character has been struck on your console keyboard, but does not input the character. This is commonly known as a Console Status test, and is clearly marked in the CBIOS source, in case it must be altered for your system. All other routine addresses are contained in a list of EQU statements, which may be easily changed to interface to any existing I/O package you may be using.

Besides CBIOS, several other programs are necessary to load and start CP/M. This operation is commonly known as bootstrapping, and for this system is accomplished in three steps or levels.

The Level 0 Bootstrap is a very short program whose sole purpose is to load and start the Level 1 Bootstrap. The Level 0 Bootstrap is located in the System Upgrade PROM, at address p+3CC hex, where 'p' is the starting address of the PROM in your system.

The Level 1 Bootstrap is a program which loads CP/M and the Level 2 Bootstrap. It is stored in track 0, sector 1 of a CP/M diskette. The Level 0 Bootstrap loads this program at address 0 in memory, and then jumps to it. The Level 1 Bootstrap loads CP/M into the top 6K of the configured system. It loads the Level 2 Bootstrap into the page of memory just below CP/M, and jumps to the Level 2 Bootstrap. Note that the Level 1 Bootstrap must be aware of the size of the configured system, in order to know

where to load CP/M. This means that the Level 1 Bootstrap is altered whenever a different sized CP/M system is generated.

The Level 2 Bootstrap implements the automatic program execution feature of the ICOM/CPM System Upgrade. It is stored in track 0, sector 2 of the CP/M diskette. This program gets control of the system from the Level 1 Bootstrap. It initializes CP/M, and then uses it to load a program into the Transient Program Area of the system. The T.P.A. starts at address 100 hex, and is the area in which all programs are normally executed under CP/M. Before passing control to the loaded program, page 0 of memory is set up to contain the vectors which normally link a transient program to the operating system. The program will therefore execute in an environment similar to that which would exist if the program were invoked from CP/M in the normal manner. Note that since the location of the Level 2 Bootstrap depends on the memory size of the configured system, it too will be re-located when a different sized system is generated. The name of the program which is loaded on startup is currently set to INITIAL.COM. If this program is not found, no error is generated. The Level 2 Bootstrap simply jumps to CP/M, which then signs on and prompts for a command.

Please note that in the instructions which follow, those portions of a line which you are supposed to type will be underlined for clarity.

Installing the FDOS/CPM System Upgrade

The first step in bringing up CP/M is to install the new interface PROM which has been provided in this package. This PROM contains, in addition to all the FDOS support routines, a simple bootstrap routine for CP/M. The PROM is installed as follows:

Either work on a metallic, conductive surface or cover your work surface with aluminum foil or a metal tray. This will decrease the likelihood of static electricity, which can damage the PROM when it is handled.

Remove the iCOM interface card from your computer system and lay it, component side up, on your work surface. Lay the new PROM, still in its protective black foam, next to the interface card.

Locate the existing type 2708 PROM on the interface card. It should be the 24-pin IC on the lower right corner of the interface card. Observe the orientation of the PROM in its socket, making note of which direction the marked corner of the PROM is pointing in. This marking will either consist of a dot, a notch, or a numeral '1'. Carefully unplug the PROM from its socket, and place it in aluminum foil or on black anti-static foam for safekeeping.

Following the same orientation as the PROM you just removed, carefully remove the new PROM from its anti-static foam and insert it into the socket on the interface card.

No further hardware modifications are required. You may now return the iCOM interface card to your computer, reconnect the disc drive cable, and power up your system.

Booting up CP/M

The procedure for booting CP/M is very similar to the procedure for booting up FDOS. With your computer, terminal, and discs powered up, execute at address C3E7 hex (BBE7 for SOLOS users). You should observe the usual iCOM prompt character on your terminal:

>

Now place your CP/M system diskette in drive 0. Wait a few seconds for the drive to come ready, and type:

>GC3CC (BBCC for SOLOS users)

The disc heads should load, and after a few seconds you should see CP/M sign on:

```
16K CP/M VERSION 1.4
A>
```

That's all it takes - you're now running CP/M !

At this point, try the various commands in the CP/M manual to satisfy yourself that everything works correctly. If you have more than 16K of contiguous memory, you may now use your working 16K system to generate a new CP/M system configured to make use of all of your contiguous memory.

Second-Level System Generation

The CBIOS and bootstrap routines have been supplied for a 16K system. This matches the CP/M release disc, which is also initially a 16K system. It is always necessary to operate a 16K system when CP/M is first received. This system is then used to generate larger systems.

Modifying Your System for Larger Memory Sizes

The program 'MOVCPM' which is described in the Digital Research documentation will generate all parts of a new CP/M system for any memory size. If you need to make any special patches, you should use MOVCPM to generate a standard system of the right memory size, and then use DDT to modify that system. The patched system can then be written into the system area of a disc by using SYSGEN. The complete process is described below:

The main idea in creating a new CP/M system is to lay out the exact contents of disc tracks 0 and 1 of the new system disc in the transient program area. By convention, this layout starts at address 900 hex, and extends for 6K, to address 20FF hex. This leaves enough room in the TPA for the program SYSGEN, which will eventually be used to write the new system onto tracks 0 and 1 of a diskette, to load under the new system image. The following list relates the important memory addresses and disc locations:

Address	Track/Sector	Program
0900	0/1	L1BOOT
0980	0/2	L2BOOT
0A00	0/3 - 1/18	CP/M (less CBIOS)
1F00	1/19 - 1/22	CBIOS

Note that these addresses have nothing to do with the actual RAM addresses at which these programs execute, but simply establish the relative locations at which they are stored on the disc. Also note that some of these addresses are 80 hex bytes higher than those described in the Digital Research documentation. This is because of L2BOOT, which did not exist in Digital Research's original system.

Start the generation procedure by using the program MOVCPM.COM to relocate a fresh copy of CP/M. To do this, bring up your original system and type:

A>MOVCPM xx *

Where 'xx' is the desired new system size in decimal K. The following messages should appear:

```
CONSTRUCTING xxK CP/M VERS 1.4
READY FOR "SYSGEN" OR
"SAVE 32 CPMxx.COM"
```

At this point, save the partially generated system by typing:

```
A>SAVE 32 CPMxx.COM
```

Where, once again, 'xx' is the new CP/M system size.

Any necessary patches are added by using DDT to read in CPMxx.COM and modify it, and then re-saving the modified memory image. The only 'catch' is that in each case, an offset or bias must be used to locate each patch in its correct place in the map described above, rather than inserting it at its eventual execution address. Depending on the module involved, these offsets may be positive or negative, and a method of determining them will be described in each case. Note that you should only run DDT if you need to make patches to the standard system. Otherwise, you may skip the next few steps, and go directly to the SYSGEN program.

Start by invoking DDT with the newly relocated copy of CP/M by typing:

```
A>DDT CPMxx.COM
-
```

The offset for patches to L1BOOT is 900 hex. This offset is fixed, since L1BOOT will always have an origin of 0.

The offset for CBIOS may be calculated using the hex calculation feature of DDT, and is given by the formula:

$$(i + s) - m,$$

where,

i	is the address in the new system image (1F00)
s	is the size of CBIOS (0200)
m	is the number of bytes of memory in your new system.

For example, for a 24K system (6000 hex bytes of memory), type:

```
-H2100 6000
8100 C100
-
```

The second number (C100), is the result of (1F00+0200)-6000, and is the correct load offset for a 24K CBIOS.

The Level 2 Boot offset is the same as the offset for CBIOS.

Using these offsets, enter your patches and return to CP/M command mode by typing control-C. The complete new system image should be saved for future use:

```
A>SAVE 32 CPMxxC.COM     ('xx' is memory size, as before)
A>
```

The following procedure may now be used to call the system image back into memory and write it onto the system area of a disc:

```
A>SYSGEN CPMxxC.COM
SYSGEN VERSION 1.4
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)B
```

Mount a disc on your second drive, and type a carriage return. The new system will be written from the TPA into the system area of the disc. Note that any files on the disc will remain intact, since SYSGEN does not affect anything but the bootstrap area (system area) of the disc. No data files are ever lost by doing a SYSGEN. After writing the new system out, SYSGEN will again ask:

```
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)
```

A carriage return at this point will reboot the system from drive A. To execute the newly-created system, mount the newly-created system disc on your first drive and execute the Level 0 Bootstrap, as described previously under the heading "Booting Up CP/M". If the system fails to come up as expected, go back and carefully check all the preceding steps. It is usually a good idea to retain a working copy of your existing system until you are certain that your new one works correctly.

The final step is to use PIP to copy those utility programs which you expect to need onto the new disc. You may find it useful to write a SUBMIT file which does this for you. This is explained in the Digital Research manuals.

Bringing Up CP/M Using ZAPPLE and/or the TDL SMB

There are advantages to using ZAPPLE with CP/M. These are found mainly in the device-handling capabilities of ZAPPLE. There is, however an additional consideration in generating your CP/M system. ZAPPLE requires that the top page of your lowest contiguous block of memory be untouched by all other programs in your system. This area contains a routine used by the 'G' command, as well as storage for all registers and ZAPPLE's stack. CP/M normally resides in the top 6K of its nominal memory size. If a CP/M system as large as your lowest contiguous block of memory were created, it would overlap the page of memory required by ZAPPLE. Thus, if ZAPPLE and CP/M are to be used in the same system, the largest possible CP/M system is decreased by 1K. For example, if you have RAM in the first 32K of your machine (not counting the RAM used for ZAPPLE, if any) then the largest CP/M system you can generate is 31K.

If you specified ZAPPLE or a TDL SMB when you ordered your ICOM/CPM, then all non-disc I/O will be performed by calling the generic routines contained within ZAPPLE. You may use the 'A' command within ZAPPLE to change you device assignments at any time, and CP/M will use the newly-assigned devices.

You may enter ZAPPLE from CP/M at any time by typing:

```
A>DDT
-GF01E
```

You may return from ZAPPLE to CP/M by typing:

```
>G0
```

Bringing Up CP/M Using SOLOS

A version of the ICOM/CPM upgrade is available which uses SOLOS for console status and input, as would be required on a Processor Technology SOL computer. A different version of the System Upgrade PROM is provided, which is addressed at B800 instead of the usual C000, so that it does not conflict with the SOLOS PROMs at address C000. Console input and status is handled by calling the SOLOS routine 'SINP', the 'standard input' routine. All other non-disc I/O is vectored thru the iCOM jump table, as usual. This version was provided if you specified SOLOS when you ordered your CP/M system.

Additional CP/M Utilities

XDIR

This program produces an expanded directory listing of a CP/M diskette. To save space, filenames are listed three across on a line, along with the number of sectors occupied by the file. For files longer than 16K bytes, each 16K segment is shown separately, with its segment (extent) number.

FDOS-CPM and FDOS-DIR

These two utilities are used to transfer programs from an iCOM FDOS-II or FDOS-III disc to a CP/M disc. FDOS-DIR will display the directory of the FDOS disc on drive B: by typing:

```
A>FDOS-DIR B:
```

Having used FDOS-DIR to identify the files you wish to transfer to CP/M, you may copy them onto the CP/M disc in drive A: from the FDOS disc on drive B: by typing:

```
A>FDOS-CPM destname.typ B:srnam
```

Where 'destname.typ' is the CP/M destination filename and 'srnam' is the FDOS source filename.

DCOPY

This is a whole-disc copying program which serves the same purpose as the 'COPY' command in FDOS. The user has the option of copying the entire diskette, just the system tracks, or just the data tracks. The commands are DCOPY ALL, DCOPY SYSTEM, and DCOPY DATA, respectively. The copy is always from disc A: to disc B:. A sample dialogue follows:

```
A>DCOPY ALL
+SOURCE ON A
+OBJECT ON B
+THEN TYPE RETURN (mount desired discs and type return)
+FUNCTION COMPLETE
+SOURCE ON A
+OBJECT ON B
+THEN TYPE RETURN (type control-C to break this loop)
+REBOOTING, TYPE RETURN (mount a system disc and type return)
A>
```

BASIC-E and RUN-E

This is a fairly powerful BASIC compiler/interpreter combination with disc data file handling capability. The complete users' manual for it is available from several sources, including Computer Mart of New Jersey and Digital Research, for a nominal charge. Please note that the BASIC-E compiler requires at least a 24K CP/M system, and therefore will produce the CP/M message 'LOAD ERROR' on the 16K starter system provided. This error will vanish when a larger system (24K or greater) has been generated and installed.

Any questions, comments, or suggestions relating to the use of this software should be directed to:

Bruce R. Ratoff
Box 28-D
1 Marineview Plaza
Hoboken, NJ 07030
201-963-0848