Z-100 LifeLine
Since 1989

December 2004

#96

~~~~~~

CP/M Plus

Going forward, by looking back...

The Z-100 is capable of running any of a myriad of CP/M versions, including CP/M-80 (Zenith's version was CP/M-85), CP/M Plus, CP/M-86, MP/M-86, and Concurrent CP/M, just to name the most prevalent.

CP/M-80, CP/M-85, and CP/M Plus were 8-bit operating systems (CP/M-80 was a hold over from the earlier H/Z-89 computer system and updated for the Z-100 in the form of CP/M-85), while the others were 16-bit, taking full advantage of the Z-100's 8088 microprocessor.

MP/M-86 was specifically designed for multi-user capability, including password protection, while allowing users to run different programs simultaneously.

CP/M Plus is actually Version 3 of the CP/M-80 Operating System. Digital Research developed CP/M Plus to take advantage of hardware changes in the 8-bit microcomputer world since the release of CP/M-80.

The release discussed here was developed by Barry A. Watzman (copyright 1984) specifically for the Z-100 and is based upon CP/M Plus from Digital Research (copyright 1983). To my knowledge, while Barry Watzman's release of CP/M Plus to the Public Domain has been considered as recently as last year, it has not yet been released. Please continue to respect these copyrights.

Questions have come up at "*Z-100 LifeLine*" regarding the differences between CP/M-80 and CP/M Plus. I published a CP/M Primer on CP/M-85 for the Z-100 in issues #44 and #45. As I'm not a regular user of CP/M and have no experience with CP/M Plus, I thought I could eliminate the confusion by printing excerpts from the "Installation Instructions and User Notes" for Z-100 CP/M Rel. 3.0 (CP/M Plus) as published by Barry A. Watzman in 1984.

If you have any questions, please contact me here at "*Z-100 LifeLine*".

Installation Instructions and User Notes
Z-100 CP/M Rel. 3.0 (CP/M Plus)

Introduction

The Z-100 CP/M Plus Installation Kit was developed by Barry Watzman to meet the needs of those Z-100 users who were using 8-bit software on the Z-100 and who desired the many advanced features of the CP/M Release 3.0 (CP/M Plus) operating system.

The Installation Kit version of this product included the following files:

File Name	Description
CLDR100.A86	Cold start loader module source
BNKBIOS3.ASM	Source for 8-bit portion of BIOS
BNKBIOS3.SPR	Assmbld/Linked Object code of BIOS
+BIOS88.SYS	Object code for 16-bit BIOS
CPM3INIT.SYS	Object code for 16-bit init module
OSLDR3.ABS	System loader for use with WRITLDR
OSLDR3.SPC	System loader for SYSGEN/BSYSGEN
ABRTSPLR.COM	Utility to abort the print spooler
ASSIGN.COM	Utility to access hdrive partitions
CONFIG.COM	Utility to configure system
FORMAT.COM	Utility to format diskettes
WRITLDR.COM	Utility to make disks bootable
GENCPM.DAT	Input Data for GENCPM.COM
EXAMP.ASM	16-bit code in .COM example

When supplied as a complete system with CP/M Plus, the following Digital Research system files were also present:

File Name	Description
BNKBDOS3.SPR	Object code for the banked BDOS
RESBDOS3.SPR	Object code for the resident BDOS
CCP.SYS	DR's CCP.COM with name changed
CPM3.SYS	Complete generated CP/M Plus system
SCB.ASM	System control block values
SCB.REL	Assmbld Sys Control block .REL File

In addition, the CP/M Plus distribution package from Digital Research included the following example and utility programs described in the Digital Research manuals:

DATE	.COM	DIR	.COM	DUMP	.COM
ED	.COM	ERASE	.COM	GENCOM	.COM
GENCPM	.COM	GET	.COM	HELP	.COM
HELP	.HLP	HEXCOM	.COM	HIST	.UTL
INITDIR	.COM	LIB	.COM	LINK	.COM
MAC	.COM	PATCH	.COM	PIP	.COM
PUT	.COM	RMAC	.COM	RENAME	.COM
~~~~~~

```
SAVE    .COM    SET     .COM    SETDEF  .COM
SHOW    .COM    SID     .COM    SUBMIT  .COM
TRACE   .UTL    TYPE    .COM    XREF    .COM
CALLVERS.ASM    DUMP    .ASM    ECHOVERS.ASM
RANDOM  .ASM    CPM3    .LIB    Z80     .LIB
```

Preparing System Disks and Booting Up

These instructions will describe the necessary steps to go through to prepare a bootable system disk and bring up the system. If you received a bootable system disk, you can proceed directly to "Booting Up The System".

PREPARING A SYSTEM DISKETTE

A bootable CP/M Plus diskette contains the following components:

1. The CP/M+ loader, OSLDR3, on the system tracks.
2. The CP/M+ 16-bit init module, CPM3INIT.SYS, on the disk as a file.
3. The CP/M+ 16-bit BIOS module, +BIOS88.SYS on the disk as a file.
4. The CP/M+ 8-bit operating system, CPM3.SYS, on the disk as a file.
5. The CP/M+ 8-bit CCP, CCP.SYS, on the disk as a file.
6. Optionally, an alternate character font, ALTCHAR.SYS, on the disk as a file.

This section describes the procedures used to prepare these modules.

The loader, OSLDR3, is supplied ready-to-run in two forms, as OSLDR3.ABS and as OSLDR3. SPC. The ABS version is for use with the WRITLDR utility programs supplied with all Watzman operating systems (CP/M+, CP/M-86 and MPM-86) for the Z-100. The SPC version is for use with the 8-bit SYSGEN and BSYSGEN programs supplied with Zenith's CP/M-85. The loader is placed onto the system tracks as follows:

```
A>SYSGEN OSLDR3.SPC
```

or

```
A>WRITLDR OSLDR3.ABS
```

You will then be prompted for a destination disk, which may be either a floppy diskette or a hard drive partition (previously formatted and assigned), and need not be of the same device type as the source from which OSLDR3 is being obtained.

Note: If you are using SYSGEN or BSYSGEN, you may be asked whether to copy the system files also. Answer NO, since these programs will not copy the correct files anyway. For further information, see the documentation for whichever program (SYSGEN, BSYSGEN or WRITLDR) you are using.

The use of a file-name specification to replace the source drive request is an undocumented feature of SYSGEN and BSYSGEN.

The 16-bit initialization module, CPM3INIT.SYS, is supplied ready-to-run and need only be copied from the installation kit distribution diskette. This module was created from its source code using the following procedure:

```
A>ASM86 CPM3INIT
A>GENCMD CPM3INIT 8080 NHEAD
A>REN CPM3INIT.SYS=CPM3INIT.CMD
```

This process requires the Digital Research 8086 assembler and command module generator (ASM86 and GENCMD), as well as the source code to CPM3INIT.

The 16-bit BIOS module, +BIOS88.SYS, is supplied ready-to-run and need only be copied from the installation kit distribution diskette. The regeneration procedure is identical to that just described for CPM3INIT.

The 8-bit operating system, CPM3.SYS, is created in a multi-stage procedure as follows:

1. Assemble the 8-bit BIOS, BNKBIOS3.ASM, with RMAC:

```
A>RMAC BNKBIOS3
```

2. Assemble the SCB.ASM file with RMAC:

```
A>RMAC SCB
```

Note: the following items must be present in SCB.ASM:

```
public @cnwdth, @cnpgln

@CNWDTH equ scb$base+1Ah ; Console Width
@CNPGLN equ scb$base+1Ch ; Console Page Length
```

3. Link BNKBIOS3.REL and SCB.REL resulting from the previous assemblies:

```
A>LINK BNKBIOS3[B]=BNKBIOS3,SCB
```

4. Run the system generation program, GENCPM.COM (suggestion: use CONTROL-P):

```
A>GENCPM AUTO DISPLAY
```

Note: The following files are required on the default drive:

```
RESBDOS.SPR   BNKBDOS.SPR
BNKBIO3.SPR   GENCPM.DAT
```

The GENCPM program normally asks several dozen questions and uses the answers in combination with the three .SPR files to create the output

system file CPM3.SYS. Use of the AUTO option causes the GENCPM program to obtain the answers to these questions from the GENCPM.DAT file rather than from the console. Later, you may wish to alter the answers to some of these questions, but for now, use the GENCPM.DAT file provided.

The 8-bit CCP (Console Command Processor) is supplied as part of the Digital Research package containing the operating system itself. Simply rename it from CCP.COM to CCP.SYS.

The procedure for generating a bootable partition is identical to that for generating a bootable floppy disk; the partition must first be formatted (just like a floppy), and then assigned, after which it is treated in exactly the same manner as a diskette.

The ALTCHAR.SYS files are optional and were not supplied in the kit. Their format is the same as that used by CP/M-85 and both the Zenith and Watzman versions of CP/M-86 and MP/M-86.

BOOTING UP THE SYSTEM

Bootup is straight-forward and is accomplished in the same manner as with any other Z-100 operating system. During bootup, the following messages are displayed:

```
Mr. W's CP/M-80 Release 3.0 (CP/M Plus) Loader, V1.0.

All Required Files Successfully Opened.

Loading CCP     .SYS at Paragraph 0F00
Loading CPM3INIT.SYS at Paragraph 1000
Loading ALTCHAR .SYS at Paragraph 1400
Loading +BIOS88 .SYS at Paragraph 2000
Loading Resident CPM3.SYS at 1000:MMMM-NNNN
Loading Banked   CPM3.SYS at 0000:XXXX-YYYY
```

In the event of any errors (required file not present, disk I/O error reading a file, etc.), a self-explanatory error message will be displayed and the boot process will be terminated by halting the CPU (a CTRL-RESET will be required to proceed).

Z-100 CP/M Plus Use Notes

These notes are intended to explain some of the finer points of this implementation of CP/M Plus which are not covered elsewhere. Many of these items are of interest only to programmers modifying or installing the system and are of little consequence to users.

1. The bootstrap methodology:

The monitor ROM reads in the first 512 bytes or 1 sector (whichever is larger) from the system tracks, which contains CLDR100 (actually the current MTR-100 reads in the entire first track, but the design specification only requires the first 512 bytes to be read). The read and execute addresses are 0:400.

CLDR100 reads in the entire contents of the system tracks (as defined by the copy of the boot device's DPB in the disk label) at LOADSEG-200h, which contains CLDR100 itself and OSLDR3. Since CLDR100 is 512 (200h) bytes long, OSLDR3 starts at LOADSEG:0, which is 19A0:0 in the standard system as supplied. Control transfers to OSLDR3 at 19A0:0.

OSLDR3 reads in CCP.SYS, CPM3INIT.SYS, ALTCHAR .SYS (if present) and +BIOS88.SYS. It then relocates the MTR-100 ROM data area into +BIOS88 .SYS, following which it reads in CPM3.SYS.

OSLDR3 is a single program approximately 2.5K long which can read CP/M files via the directory. It is an integrated program which does both the directory decoding and physical disk I/O -- there is no separate BDOS or BIOS. Because this program is considered proprietary, source code was not supplied nor available; however the following table is present at the start of OSLDR3 and may be patched if desired:

```
Org   OSLDR3+3
;
DW   ALTSEG     ;where to load ALTCHAR.SYS
DW   0          ;LOADSEG, where OSLDR3 runs
                 (filled in by CLDR100)
DW   CCPSEG     ;where to load CCP.SYS
DW   INITSEG    ;where to load CPM3INIT.SYS
DW   B16SEG     ;where to load +BIOS88.SYS
DW   0          ;8-bit CPU startup addr
                 (filled in by OSLDR3)
```

There are major impacts to all modules of the system if any of these addresses are altered, and this information should only be used by assembly language system programmers who have access to the full system source codes and who have a complete understanding of the inter-module interactions within the CP/M Plus System.

After OSLDR3 has loaded the system, control passes to CPM3INIT, which performs all system initialization functions and then jumps to the cold start entry point of +BIOS88. +BIOS88 prints the signon message and transfers control to the 8085 micro-processor, which begins execution at address 0 of the 8-bit CPU's 64K of memory. This address contains (at this point) a jump instruction to the address specified in the CPM3 .SYS header record, which happens to be the cold start entry point in the 8-bit BIOS. From this point, operation of the system proceeds in a normal manner.

2. Disk mapping is automatic on bootup. Drive A: is the boot drive (5", 8" or hard drive).

If the boot device is a floppy, other drives of the same type found physically existing are assigned sequentially, then existing drives of the other type (5" or 8") are assigned sequentially.

If the boot device is a hard drive, the boot partition (only) is automatically assigned as drive A:, then existing 5" floppy drives are assigned sequentially, then existing 8" drives are assigned sequentially.

The system as supplied supports up to two floppy drives of each type (5" or 8"), but can be easily reconfigured to support up to four drives of each type, if desired. See Use Note number 17.

3. There is NO support for imaginary drives.

4. There was NO explicit support for 96 tpi 5" drives; it was added in update 1.2.

5. Hard drive partitions other than the boot partition are accessed via the ASSIGN utility. Up to four partitions may be assigned at any one time. Operation of an assigned partition is identical to that of a floppy. It is also substantially identical to the handling of partitions in Zenith's operating systems.

6. The operating system is able to mount and use CP/M-86 diskettes in the IBM-PC format, but this action is NOT automatic. See the ASSIGN utility documentation.

7. Disks and hard drive partitions are made bootable by writing a loader (OSLDR3.ABS) onto the system tracks with WRITLDR and putting the four required system files comprising the operating system (CPM3INIT.SYS, CCP.SYS, +BIOS88 .SYS, and CPM3.SYS) onto the disk. The only extra step for a hard drive partition is to first ASSIGN the partition before using WRITLDR. Both floppies and hard drive partitions must be formatted before they can be used or WRITLDRed.

8. All known Zenith 48tpi disk formats are supported. Due to a bug in the Z-100 monitor ROM, MTR-100, it is not possible to boot from the undocumented extended double density 5" disk format (five 1024 byte sectors per track), but disks of this type can be used without difficulty (This bug was reportedly fixed in MTR-100 version 2.5).

9. Zenith has shipped at least four disk formats with the wrong extent mask in the Disk Parameter Block. This operating system supports both the correct and the incorrect versions of these formats, but an ambiguous situation can arise with some double density 8" disks.

In addition, CP/M-85's 8" driver will force use of the wrong extent mask even if the diskette contains a valid label with the correct extent mask. Therefore, the following restrictions are mandatory when using 8" double density diskettes:

* Any 8" diskettes created under Zenith's operating systems MUST be SYSGENed under the Zenith operating system before they can be used on this operating system.

* The only 8" double density format generated by this CP/M Plus' FORMAT program which can be subsequently used under Zenith's operating systems is the 8*1024 FORMAT.

Single sided, single density 8" diskettes may be freely interchanged between the two operating systems without difficulty.

10. When using the print buffer, it should be noted that transfer of text between the applications program and the print buffer is not instantaneous, and requires about one second per thousand characters, depending on how much processing the program is doing to generate each character.

11. This implementation of CP/M Plus is very fast and approaches the limits of what it is physically possible to achieve. Because of this, the performance difference between the hard drive and the memory disk is much less than might be expected, often only 20-30% for sequential access applications.

12. This is a banked implementation of CP/M Plus. In this implementation, there is 56K of banked memory available for use by the 8-bit Digital Research portion of the CP/M Plus system. This is used approximately as follows in the standard system:

```
    .5K - BCB's
    .5K - Checksum tables
     3K - Allocation tables
    11K - Banked BDOS
    21K - Directory & data buffers
    20K - Hash tables
```

Note that this usage is dependent to a large degree on the contents of the disk definition tables in BNKBIOS3, and on the answers given to the questions asked by GENCPM. The user is therefore free to change this usage within certain limits, while the usage given above corresponds to the standard system as shipped.

13. You may notice from time to time that you perform an operation (say a DIR command on a floppy disk drive) and get a response with no

drive activity (e.g., the red LED on the front of the drive NEVER comes on). This system keeps track of the amount of time (milliseconds) which has elapsed since the last access for each removable media drive in the system. When a disk I/O request is made, if the time since the last access FOR THE SAME DRIVE is less than the threshold value (currently 1500 milliseconds), then the system assumes that the diskette could not have been changed, and that the contents of its disk buffers for this drive, including its directory buffers, are valid.

A large amount of testing and careful design thought went into this feature, and it improves floppy disk performance by nearly 50%. People intentionally trying to "beat the system" may be able to produce abnormal results, but this was considered acceptable for a 50% performance improvement in floppy disk operations.

14. This operating system uses both 8 and 16 bit code rather extensively, not only in the BIOS, but also in the utilities. In order to facilitate this, a feature was added to the operating system that allows the execution of blocks of 16-bit 8088 object code as a part of and imbedded within 8-bit .COM programs.

A decision was made to document this feature to end users who may wish to use it in their own programs. To accomplish this, a sample program using this feature, EXAMP.ASM, is included in this release. This example, though trivial, illustrates all of the important points required to use this feature and should be studied carefully by anyone who wishes to put it to use.

BE CAREFUL -- This is a VERY complex operating system with many modules inter-acting in less-than-straightforward ways. If you start messing with the BIOS, interrupts or the initialization of the 8259's, 8253, 68A21's, 6845, or 2661's, you can blow the system away.

In order to make the best use of this feature, you should obtain ASM86 and GENCMD, the CP/M-86 assembler and command module generator, either as .COM cross-programs (runs under 8-bit CP/M) or as 16-bit .CMD programs (ASM86 is MUCH faster in the 16-bit .CMD version but it means running under CP/M-86). To obtain these as .CMD programs, find the Watzman implementation of CP/M-86, or to get the .COM versions, get the full CP/M-86 package.

15. The operating system looks for a submit file named PROFILE.SUB on bootup and runs it automatically if found. This can be very useful; consider the following example:

```
ASSIGN B:=,C:=,D:=,E:=,B:=CPM85,
       C:=F5;0,D:=F8;0,E:F8;1
SETDEF *,A:,B: [T=M,NO PAGE]
```

This PROFILE.SUB file does several things:

* re-assigns all drives in the system regardless of how the default assignment during the boot process set them up,
* sets up a drive search chain so that the default drive, drive A: and drive B: are automatically searched for a .COM file by the CCP,
* assigns the system temporary drive to the memory disk (drive M:), and
* turns off console paging (which drives some people up a wall, especially when using the TYPE command).

16. The quickest way to get a listing of all files on a disk with their sizes is:

```
A>DIR[A
```

or

```
A>DIR B:[A
```

It is not necessary to close the bracket.

17. If you wish to use more than two 5" and/or 8" floppy drives, make the following changes to the system:

First, edit the BNKBIOS3.ASM file to set the drive table entries for the currently unused drives that you wish to activate to their correct values. As shipped, these entries are set to zero for the third and fourth 5" and 8" drives in the system. The correct labels to activate them are shown as comments in the source code. Now reassemble and relink the BIOS (the procedure for doing this is given in the section of this document entitled "PREPARING A SYSTEM DISKETTE").

Next, rerun GENCPM **without** the AUTO option. Since you will now need allocation tables, checksum tables and hash tables for the additional drives, there will be less memory available for disk and data buffers, and you will have to specify a smaller number of buffers. As this is written, the GENSYS.DAT file specifies 12 hard drive directory buffers, 26 hard drive data buffers, and 14 floppy disk directory buffers (refer to a GENCPM listing generated by running GENCPM under Ctrl-P with the standard distribution system).

All hard drive directory and data buffers are 512 bytes in length, while all floppy disk directory buffers are 128 bytes in length, and there are no floppy disk data buffers (because floppy disk deblocking is still done in the BIOS rather than the BDOS, and the BDOS is told that all floppy disk sectors on all floppy disk formats are physically 128 bytes long even though this is not the case). Which of these numbers you reduce and by how much is left to you.

Finally, there is a drive mask byte at an offset of 15h bytes into +BIOS88.SYS that controls which drives are automatically searched for during bootup. The current value is 33h, corresponding to the first two 5" and 8" floppy drives (the low order nibble is the 5" drives). Change this as required if you want the additional drives automatically assigned (you can patch it with SID or DDT). The system will support up to four floppy drives of each type.

18. If you have a hard drive system and there are drives in the system that you will **never** use (such as 8" floppies, or more hard drive partitions than you have), you may find it beneficial to set the corresponding drive entries in the drive table to zero and regenerate the system. This will free the allocation, checksum and hash table memory previously used by the drive you don't have so that you can assign more hard drive directory and data buffers, possibly increasing system performance (speed).

If you have a floppy-only system, do NOT alter the drive table, but do consider decreasing the number of hard drive buffers and increasing the number of floppy directory buffers. One hard drive buffer = four floppy buffers, and, as noted above, floppies do not use data buffers.

19. Some Zenith software products (in particular the Zenith supplied 8-bit Microsoft languages) will not operate under this implementation of CP/M Plus unless either the operating system or the software product is modified.

The reason is that some products, such as the 8-bit Microsoft languages, check to see whether they are operating under a Zenith operating system, which, of course, CP/M Plus is not. It is possible (in fact, rather easy) to modify either the operating system itself (one byte of the operating system serial number is an OEM ID) or the software product being run (to take out this "check"), after which everything works fine.

Note: Due to legal reasons, the detailed procedures for these modifications could not be presented here.

20. There are a few potential software incompatibilities between CP/M Release 2.2 and CP/M Release 3.0 (CP/M Plus). While these occur rarely in commercial software, they are more common in custom and user-group software. The major items of incompatibility are:

* Programs which do direct BIOS calls for disk I/O may not work (unless they were written for CP/M Plus) because disk calls to READ and WRITE transfer PHYSICAL sectors rather than 128 byte logical sectors.

Note, however, that in this implementation **floppy disk** deblocking is done in the BIOS rather than in the BDOS (unusual for CP/M Plus), and therefore calls to READ and WRITE do transfer 128 byte sectors when doing floppy disk I/O, regardless of the physical sector size.

This is <u>not</u> true for hard drive operations, where physical sectors (512 bytes) are transferred.

The use of BIOS deblocking was done for performance reasons, not for the slightly greater degree of compatibility which is a fortunate but only accidental side-effect.

* Programs which modify the BIOS jump vectors to get control themselves during BIOS operations may be in for a rude surprise. A subsequent BIOS call may occur from the BDOS, which may have done a memory bank switch between the BDOS call and the BIOS call. When the BIOS jump vector sends the CPU into what it thinks is the user-routine, it will be sending it to the correct address **in the wrong bank of memory!** The system will crash.

If you absolutely MUST write such code, the portions of the TPA below 1000h as well as between F000h and the start of the BDOS (currently at F500h) may be used for this purpose, as these portions of the TPA are not banked by MAP mode 3 in the standard Z-100 memory map PAL. The code in this segment may re-establish normal banking (e.g., select TPA bank) before transferring control to banked portions of the user program. If banking is altered, it should be restored prior to returning from the BIOS call. For similar reasons, it is important to be fully cognizant of where the stack is located if you are going to play with memory mapping in a user program.

Fortunately, these potential incompatibilities in fact impact only a very small number of programs, mostly disk edit utilities and other system-programmer utilities, and cause few problems with typical commercial applications software.

Using the Print Buffer and Memory Disk

Version 1.1 of the Watzman implementation of CP/M Plus includes a print buffer and, if sufficient memory is available, a Memory Disk.

During the initialization phase of the boot process, code within CPM3INIT sizes memory and allocates excess memory to the print buffer and possibly to the memory disk. Memory beyond that which the operating system explicitly needs for its own code and data areas (approximately 174K) is first allocated to the print buffer until the print buffer size reaches 64K.

Then all excess memory remaining is allocated to the memory disk, which is initialized and formatted by CPM3INIT.SYS.

In a 192K system, there is a print buffer of approximately 18K, and no memory disk.

In a 448K system, there is both a 64K print buffer and a memory disk with a capacity of approximately 210K.

The print buffer is transparent to all operating system functions, is always in use and cannot be bypassed. In operation, all output for the LST: logical device is sent to the print buffer, which buffers the output up to its capacity and sends data to the physical LST: device(s) (more than one physical device may be assigned to the LST: logical device) as they are able to accept the characters.

The list status check function returns the buffer status rather than the status of the physical device(s).

If the buffer becomes full, the system will not return from a list output call until the buffer has sent one or more characters and is able to accept another character (in other words, you wait, just as you would without a print buffer).

The print buffer is implemented via polling rather than interrupts. Without going into all the reasons for this, it will simply be stated that it would have been very difficult to have implemented a fully interrupt driven print buffer which supported multiple devices being assigned to the list device simultaneously with some of them driven by software ETX-ACK drivers while others use hardware handshaking.

In actual practice, use of the print buffer does not degrade system performance to a measurable degree, and the printer(s) run at full speed almost without regard to what the application program running with the printer is doing. 8-bit software which goes into totally compute bound loops with no operating system calls for long periods of time may result in operation of the printer(s) at reduced speed (roughly 60 to 120 cps) for the period of execution of the 8-bit compute bound loop. (This has been corrected in CP/M Plus Update 1.2.)

The ABRTSPLR utility may be used to terminate printing of the buffer's contents if desired.

The memory disk is implemented as drive M:, which is permanent and cannot be altered with the ASSIGN program. The memory disk is formatted by CPM3INIT and cannot be formatted separately with FORMAT or any other program. To determine the memory disk size and allocation characteristics, the following command may be used:

A>**SHOW M:[DRIVE]**

The memory disk can be used just like any other disk drive in the system; however, it must be remembered that it is volatile, that is, all of its contents are lost when power is turned off, or when the system is re-booted. It is recommended that the memory disk therefore be used for static data and overlay files only, and not for the actual data and/or text files being worked on during the computing session.

A good example of such usage would be to put WORDSTAR, the WSMSGS.OVL file, and the WSOVLY.OVL file on the memory disk while keeping the actual text file being edited on a "real" (e.g., non-volatile) disk.

User's Guide
FORMAT Program for CP/M Plus

The FORMAT program is used to format floppy diskettes and hard drive partitions. FORMAT is invoked in the usual manner without any arguments:

A>**FORMAT**

The program will print its signon message, warn you that the format operation will destroy any data currently recorded on the disk being formatted, and ask you if you wish to proceed. Respond with a "Y" if you do indeed wish to format a disk.

You will then be asked which drive is to be used for this operation. You should respond with the drive ID (one of the letters A thru P, without a colon) of a valid and assigned drive. The program then identifies the specified drive type and unit number:

Drive C: is 8" Unit 1.

Where units are numbered 0 through 3. If the requested unit is a 5" drive, you will then be asked for the number of sides:

Number of Sides (1=single, 2=double) ?

For 8" drives, determination of the number of sides is automatic and is done by the hardware and the FORMAT program.

Following side selection, you will be shown a menu of from 2 to 4 format options (depending on the type of diskette in use).

The standard option for 5" diskettes is 8*512 sectors. The higher capacity 5*1024 format may

also be selected; it is compatible with both this operating system and Zenith's operating systems, however there are two drawbacks associated with it:

* Due to a bug in the Z-100 monitor ROM, MTR-100, it is not possible to boot from this format, and
* The 5*1024 format is NOT as reliable as the 8*512 format (in particular, serious problems can occur if a diskette with this format is written on by a drive with a slightly fast rotational speed). Use this format at your own risk.

For 8" diskettes, you will have a choice of 4 single sided options and 3 double sided options (single density is not supported on double sided 8" diskettes):

* The single density format is the standard IBM 3740 single sided format often used for interchange of CP/M programs between different vendor's systems.
* The double density 26*256 format is the standard IBM 2D format except that the data fill character is E5 instead of 40 for CP/M compat-ibility. This format is identical to the ZDS format for double sided 8" diskettes on the Z-100, but the extent mask is different (see the Use Notes).
* The 8*1024 "extended density" format is the same as the Z-47's extended density format and is supported by both this system and Zenith's operating systems -- this is the recommended format for use with this system, even though Zenith does not support creating diskettes of this type on the Z-100.
* The optional high capacity 9*1024 format is not supported by Zenith, and may not be used as a boot device. Like the high capacity 5*1024 5" format, this format is sensitive to drives with a slightly fast rotational speed to a much greater degree than the other formats offered.

When the format is selected, you will be asked to insert the disk to be formatted, at which point the format operation will begin.

If an error is detected either during the write or during the subsequent verification of each track (which occurs automatically), you will receive an error message and the operation will be aborted. Note that error code 40h is a write-protected diskette.

When the operation is complete, you will have the option to format additional diskettes before returning to the operating system.

To FORMAT a hard drive partition, respond to the drive request with the drive to which the partition is to be ASSIGNED after it is formatted, followed by the partition and operating system name:

Which Drive Do You Wish To Use For
This Operation (A-P) ?
==>**F:=wordproc;CPM+**

This entry formats the partition whose name is wordproc, and whose operating system name is CPM+, and then assigns this partition to logical drive F:. The following restrictions apply:

* The partition being formatted must not be currently assigned to ANY logical drive (A: thru P:).
* The logical drive (F: in this example) must not currently have ANY device assigned to it.
* No more than 3 hard drive partitions may be assigned PRIOR to attempting to for-mat a hard drive partition.
* Partitions may be formatted only if their size is at least 41K and not more than 16,393K. See the WARNING below regarding large partitions.

WARNING: CP/M Plus, MP/M-II, MP/M-86, and Concurrent CP/M-86 all support partitions as large as 32 megabytes, while CP/M-80 (Zenith's CP/M-85) and CP/M-86 support partitions only up to 8 megabytes in size.

CP/M Plus will correctly format partitions larger than 8 megabytes. If you create such a partition, you are warned NOT to use it with CP/M-80, CP/M-85, or CP/M-86 under ANY circumstances. Loss of all data stored on the partition, and possibly even loss of data on the disk OUTSIDE of the partition is a distinct possibility under these condi-tions. A warning message to this effect is displayed on the console screen, if a partition larger than 8 megabytes is to be formatted.

User's Guide
ASSIGN for CP/M Plus

The ASSIGN Utility provides a means of:

* "mounting" hard drive partitions as logical drives (A: thru P:),
* "mapping" logical drives (A: thru P:) to physical diskette drives, and
* activating the BIOS' ability to access IBM-PC CP/M-86 diskettes as drive P:.

The ASSIGN Utility allows both command line specification of the assignment requested and prompt specification.

In command line specification, the assign-ment requested is placed immediately follow-ing the invocation of ASSIGN at the CCP prompt, for example:

A>**ASSIGN F:=UTILITIES;CPM**

In the prompt specification, ASSIGN is invoked without parameters:

A>**ASSIGN**

and then the program prints its own prompt, an asterisk, to which the user enters the desired assignment and a carriage return:

***F:=UTILITIES;CPM**

Either method may be used; in both methods, multiple assignments separated by commas are supported. For example:

A>**ASSIGN B:=,C:=F8;0,D:=WORDPROC;CPM+**

1. ASSIGNING HARD DRIVE PARTITIONS

To ASSIGN a hard drive partition, use the following command format:

ASSIGN X:=PARTNAME;OSNAME

Where X: is the logical drive to which the partition is to be assigned (A: thru P:), and PARTNAME and OSNAME are the partition and operating system names of the desired partition (as entered when the partition was created with the PART utility from the Z-100 Winchester Utilities Disk).

Note: The semicolon and OSNAME are optional, and, if omitted, the first partition with a partition name matching PARTNAME will be assigned to logical drive X:, regardless of its OSNAME.

Drive X: must not be assigned to a permanent device (hard drive or memory disk) when the command is issued. The partition being assigned must not currently be assigned to ANY logical device. If drive X: is assigned to a temporary device (e.g., floppy disk drive) when the command is issued, it must not be the current default device.

Note: Hard drive partition assignments are PERMANENT once made and CANNOT be altered without rebooting the system.

2. DE-ASSIGNING LOGICAL DRIVES

A logical device (A: thru P:) assigned to a temporary drive (floppy drive, but NOT a hard drive partition or memory disk) may be de-assigned as follows:

ASSIGN X:=

Then press RETURN or place a comma after the "=" sign. X: is the logical drive (A: thru P:) to be de-assigned. Once de-assigned, any attempts to access the logical drive further (until and unless it is re-assigned) will result in a select error.

Note: The current default drive cannot be de-assigned.

3. ASSIGNING FLOPPY DISK DRIVES

A floppy drive may be assigned to a logical drive via the following command:

ASSIGN X:=F5;Y For 5" floppy drives

or

ASSIGN X:=F8;Y For 8" floppy drives

Where X: is the logical drive (A: thru P:) to be assigned and Y is the physical unit (0 thru 3) of the type of floppy drive (5" or 8") specified. The physical drive unit specified will be de-assigned from the logical drive to which it is currently assigned and then reassigned to the requested logical drive. The current default logical device (A: thru P:) cannot be assigned to another physical unit. The physical device assigned to the current default device cannot be assigned to another logical device. Logical devices (A: thru P:) assigned to permanent physical devices (hard drive partitions and the memory disk) cannot be reassigned to another logical unit. Five-inch drives cannot be assigned to logical drive P: except as IBM compatible drives (discussed next).

4. ASSIGNING A DRIVE TO IBM-PC CP/M-86 FORMAT

Drive P: (only) can be assigned to read and write IBM-PC format CP/M-86 diskettes. This can be done as follows:

ASSIGN P:=IBM;X

Where X is the 5" unit number (0 thru 3) of the drive to be used as an IBM format drive. Note that if the current default drive is a 5" floppy drive, the unit assigned to it cannot be specified. The specified drive is de-assigned and then re-assigned as drive P: for use with the IBM-PC CP/M-86 format only. The drive can no longer be used for other formats unless it is first de-assigned from the PC IBM-PC format and then re-assigned.

User's Guide
WRITLDR for CP/M Plus

The WRITLDR Utility provides a means of writing system loaders onto the system tracks of floppy diskettes and hard drive partitions, thus making

them bootable. The bootstrap methodology used in this implementation assumes that the operating system to be loaded resides on a bootable disk as one or more files to be loaded by an operating system loader, OSLDR3.

On bootup, the MTR-100 ROM reads the first 512 bytes of track zero (or the first sector, if the device uses a 1K sector) into memory at 0:400 and transfers control to it at 0:400. This action must initiate a sequence of events which ultimately results in the successful loading and execution of the operating system.

OSLDR3.ABS is the loader which causes this sequence of events to occur; WRITLDR is the program which puts OSLDR3.ABS onto the system tracks when a disk is first made bootable.

This program is similar in external appearance and function to the SYSGEN program supplied with most 8-bit CP/M-80 systems, although its actual internal function and operation is quite different.

WRITLDR may be initiated with an optional filename specification:

A>**WRITLDR**

or

A>**WRITLDR A:OSLDR.ABS**

If an optional filename is specified without an explicit extension, the extension .ABS is assumed. The default drive is used if no drive is specified.

When WRITLDR is invoked with a filename specified, the file is read into the loader buffer within WRITLDR and the user is prompted for a destination drive name:

Destination Drive Name
(Or Return To reboot):

At this point, the user should enter the logical drive ID (A: thru P:) of the disk device on which he wishes to write the loader. When this is done, WRITLDR will request a confirmation:

Destination On X:, Then Type Return

Assuming that the user has entered the correct drive and wishes to continue, he should enter a carriage return (a **Ctrl-C** will abort the WRITLDR operation). WRITLDR will then read the label from track 0 sector 1 of the specified destination device, validate the label, copy it into the label area of the loader, and write the loader, which now has a label corresponding to the destination device (regardless of the source device), onto the destination device, thus making it bootable.

WRITLDR may also be invoked with no filename specified:

A>**WRITLDR**

In this case, the user will first be prompted for a source drive name and then be asked to confirm the source drive name in a manner similar to that in which the destination drive name was requested and confirmed:

Source Drive Name:
Source On X:, Then Type Return:

Assuming that the drive indicated in the second message is the drive from which the user wishes to read the loader, he should enter a carriage return, after which WRITLDR will read the loader from the specified drive and prompt the user for a destination drive:

Destination Drive (Or Return To Reboot):
Destination On X:, Then Type Return

Devices to be specified as sources and destinations may be either floppy disk drives or hard drive partitions. In either case, they must have been assigned and they must have a valid label in their first sector.

Assignment of floppy disk drives in a system is normally done automatically during bootup, thus no action is required to accomplish it, although a physical drive's assignment to a logical CP/M drive (A: thru P:) may be altered, or the drive may be de-assigned entirely, with the ASSIGN utility (see the ASSIGN utility User's Guide).

Assignment for hard drive partitions is automatic only if the boot device is a hard drive partition, and then only for the one partition which is the boot device; other hard drive partitions must be explicitly ASSIGNed with the ASSIGN command before WRITLDR can be used to put a loader on them.

Labels are written onto the first sector of disk devices by the FORMAT program. If you use CP/M-85 to format diskettes, keep in mind that CP/M-85's FORMAT program alone does not always write a full label compatible with CP/M Plus. However, if a disk formatted with CP/M-85 is then SYSGENed with CP/M-85 (using the CP/M-85 SYSGEN program), it will have a complete label.

There is no requirement whatsoever that the source and destination drives be of the same type. The system loader (OSLDR3.ABS) is identical for all 5" floppies, all 8" floppies, and all hard drive partitions, the only difference being in the label area (the 4th thru 29th bytes of Track 0 Sector 1).

Since the action of WRITLDR is to replace these bytes from the source device with the correct label from the destination device, it is entirely satisfactory for the source and destination device to differ.

As a final comment, when a file is specified, the file is written to the destination drive's system area from the beginning. That is, the first byte of the file becomes the first byte of the first sector of Track 0. There is no allowance, as there was in CP/M-80 SYSGEN, for the 800h bytes between 100h and 900h that were saved with the system image by a CP/M-80 SAVE command following a CP/M-80 MOVCPM command.

User's Guide
CONFIG for CP/M Plus

The CONFIG program is a replacement for Digital Research's DEVICE program, and its operation is sufficiently similar to the operation of DEVICE that the DEVICE documentation in the Digital Research manual may be used. However, CONFIG supports a number of additional options not present in DEVICE. For a full list of these, use the HELP option within CONFIG (type **CONFIG HELP** at the A> prompt, or type **HELP** once within CONFIG).

The following abbreviations are used to identify physical device options:

XON	XON-XOFF (Ctrl S - Ctrl-Q) Software handshaking. Rarely used.
ETX	ETX-ACK software handshaking. Used by most daisy-wheel printers.
RTSL	Req. to send (pin 4) ready when low hardware handshaking.
RTSH	Req. to send (pin 4) ready when high hardware handshaking.
DTRL	Data Terminal Ready (pin 20) when low hardware handshaking.
DTRH	Data Terminal Ready (pin 20) when high hardware handshaking.
NONE	No Handshaking.
1.0SB	Abbreviation for 1 stop bit.
1.5SB	Abbreviation for 1.5 stop bits.
2.0SB	Abbreviation for 2 stop bits.
5BW	Abbrev. for 5-bit word length.
6BW	Abbrev. for 6-bit word length.
7BW	Abbrev. for 7-bit word length.
8BW	Abbrev. for 8-bit word length.
OFF	Parity Off.
EVEN	Use Even Parity.
ODD	Use Odd Parity.

The following is an example of how to assign the CP/M list device (LST:) to both the parallel port (J3 on the back of the Z-100) and serial port A (J1 on the back of the Z-100) at the same time (e.g., printer output is to be printed on two printers at once), and configure the serial port for 1200 baud, 8-bit word length, even parity, 1 stop bit:

```
A>CONFIG LST:=PARPRT,SERAJ1[1200,ETX,
         1.0SB,EVEN,8BW]
```

This could also have been done by typing only the word CONFIG at the prompt and then entering the actual assignment (the rest of the line) in response to the prompt once in the CONFIG command.

For reference, the **STANDARD** configuration for several common printers are listed below (all normal devices will use an 8 bit word length, no parity and 1 stop bit at any baud rate above 300 baud.

A - Diablo 620
    300 Baud
    ETX-ACK Software handshaking
B - Diablo 630, 1600 Series (1610, 1620, 1640, 1650), Xerox 1700 series
    1200 Baud
    ETX-ACK Software handshaking
C - Epson MX-80 (via serial port)
    4800 Baud
    DTRH hardware handshaking
    (connect to serial port A/J1)
D - H/Z-25 or H/Z-125
    4800 Baud
    RTSH hardware handshaking
    (connect to serial port A/J1)
E - H-14 or WH-24 (Heath TI-810)
    4800 Baud
    RTSL hardware handshaking
    (connect to serial port A/J1)
F - 300 Baud Modem (also DEC LA-34, LA-36, many other 300 baud printers)
    300 Baud
    No Handshaking
G - 1200 Baud Modem
    1200 Baud
    No Handshaking

Connecting printers to serial port B (J2 on the Z-100) will normally require a special cable.

For ETX-ACK and XON-XOFF printers, it will normally be a so-called **"null modem"** cable that crosses pins (2,3) (4,5) and (6,20). These can be purchased already made up from most computer supply vendors.

For hardware handshaking printers, it may have to be a "true" custom cable that you will have to wire yourself.

Modems will normally connect directly to serial port B and would require a special cable for connection to port A, just the opposite of printers.

Note: A "standard" cable is defined as one which connects pins 1 thru 8 and 20 of each end to the same pin at the other end.

It was necessary to write CONFIG because while DEVICE was well-intentioned, it was far too simplistic to support a realistically acceptable range of configuration options (it did not, for example, allow for hardware handshaking at all, much less for selection of which line and polarity would be used; nor did it allow for more than one type of software handshaking).

Z-100 CP/M Plus Update 1.2

Changes From Previous Versions:

1. Interrupt Driven Peripheral Buffers

The Interrupt driven peripheral buffers are mostly transparent to the user's of the system. As supplied, they are small buffers intended to provide improved performance of the printers and to simplify implementation of communications software. The previous version of the Watzman implementation of CP/M Plus used a polled print buffer, and while it nearly always operated the printer(s) at full speed, it could slow down if the applications program went into a totally compute bound loop.

Now, however, full speed operation is assured, even if the applications program is compute bound. Further, communications programs are "buffered" by the interrupt driven FIFO buffers, which makes it much easier to implement high speed (1200 baud) communications software without danger of loss of characters.

There are three output buffers: one each for Serial Port A, Serial Port B, and the Parallel Printer Port.

There are two input buffers: one each for Serial Port A and Serial Port B.

The size of these buffers is controlled by equates near the front of the source listing: SAIBSIZ (Serial A Input Buffer SIZe), SAOBSIZ (Serial A Output Buffer SIZe), SBIBSIZ, SBOBSIZ, and PROBSIZ (PaRallel Output Buffer SIZe). The buffer size is altered simply by changing this equate and regenerating the BIOS:

```
A>ASM86 +BIOS88
A>GENCMD +BIOS88 8080 NHEAD
A>REN +BIOS88.SYS=+BIOS88.CMD
```

The buffer(s) may be made as large as desired providing that the total size of +BIOS88 does not exceed 64K.

2. 96 TPI Support

The support for 96 TPI drives is mostly transparent to the user. The BIOS supports a total of six 96 TPI formats (16*256, 8*512, and 5*1024 track, in single and double sided formats), while the FORMAT program supports formatting 4 of these, the 8*512 and 5*1024 track, single and double sided formats.

The boot loader fully supports 96 TPI also, but the cold start loader does not, due to space limitations (the entire cold start loader must fit in 512 bytes and must handle every supported single and double density, single and double sided 5" and 8" and hard drive partition, so it is already quite tight).

The implication of this is that all of the **SYSTEM TRACKS** must either be in cylinder zero or must not require double-stepping. In other words, you can't boot from a **single sided** 48 TPI floppy diskette inserted into a 96 TPI drive.

The operating system determines whether a drive is 48 TPI or 96 TPI by looking at a bit in the floppy disk controller's Aux. Status Port. This bit is controlled by a section of the DIP switch on the disk controller. This bit is looked at only once, during system initialization, but it is looked at separately for each drive in the system (in fact, it is looked at for all four possible 5" drives, even those for which the drive mask bit is not set, since it takes only a few milliseconds).

Note: Each drive is selected while the switch is being looked at to determine that drive's track density.

This means that modifications such as those published in *"BUSS"*, *"SEXTANT"*, and *"H-SCOOP"* which control this bit by tying it into the drive select circuitry will work, and it is possible to have mixed 48 TPI and 96 TPI drives operational simultaneously. There is no "manual override" of the automatic density selection (at least at this time).

The system supports reading of 48 TPI diskettes in 96 TPI drives, but such usage is strictly **read-only**; if a write were allowed, the diskette could never again be reliably read in a 48 TPI drive.

3. Screen Saver

The screen saver is a module in the BIOS which blanks the screen of the CRT display after a predetermined amount of time (the "inactivity threshold") has elapsed with no activity, either input (keyboard) or output (new characters displayed).

The purpose of the screen saver is to prevent burning a static image into the CRT phosphors during long periods of inactivity.

The screen saver is transparent to the user; the only options are to set the inactivity threshold or to turn the screen saver off.

The inactivity threshold is set with the CONFIG program. It is set in decimal minutes, from 1 to 255. If a value of zero is entered, the screen saver is turned off and the screen will never be blanked.

Once blanked, the screen restores automatically if there is either input to the keyboard or output on the screen.

4. Changes to CONFIG

The CONFIG program has changed quite a bit. An additional sub-menu now is present which is accessed by entering MISC in response to the CONFIG prompt or on the command line (e.g., A>**CONFIG MISC**).

Functions of the additional menu are mostly self-explanatory, except for the drive search mask. The drive search mask is a one byte bit-mask which determines which drives will be searched for during bootup. Since drive searching takes quite a bit of time (up to 15 seconds per drive in a worst case situation), and there are potentially eight drives to search for, a significant amount of time can be cut off of the boot time through the use of the drive mask.

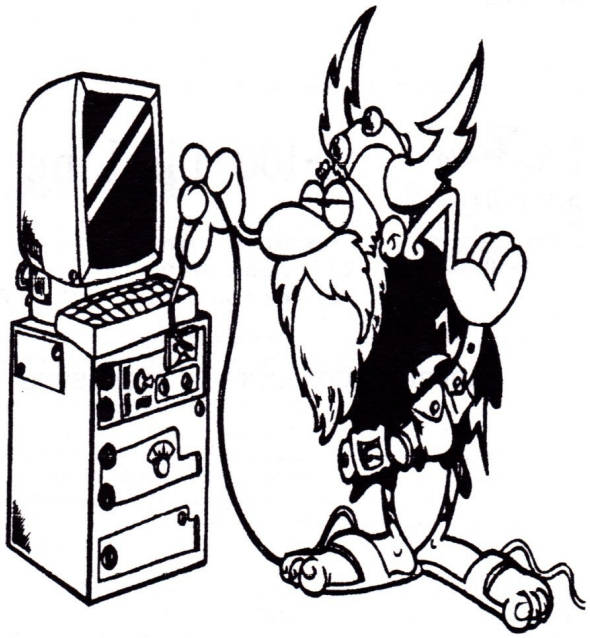
Each bit of the mask corresponds to a drive; the low nibble corresponds to the four possible 5" drives, while the high nibble corresponds to the four possible 8" drives. The least significant bit in each nibble is unit 0, while the most significant bit is unit 3. The initial value is 33h, which causes the system to look for two 5" and two 8" floppy drives.

Installation:

To install the new version of CP/M Plus, replace all of your existing files with the new files of the same name from the update diskette and follow the Installation Instructions and User Notes supplied with the original distribution.

Basically, you will have to rerun GENCPM, and use WRITLDR to put the new version of OSLDR3 onto the system tracks of all bootable disks or partitions.

NOTE: Revising the loader is NOT necessary unless you plan to use 96 tpi drives.



Check your Z-100 regularly!

I hope you enjoyed this article and find it helpful.

Cheers,

Steven W. Vagts
Editor and Publisher
"Z-100 LifeLine"
"Long live the H/Z-100"