ZCPR Version 1.0 Release Notice

Greetings, Everyone,

I am very pleased to announce (appropriately at Christmas time, so Merry Christmas) the formal release of ZCPR -- a Z80-based Command Processor Replacement. This program replaces the CP/M CCP and greatly extends its capabilities.

Some of you may have heard of CCPZ and MCPZ during various discussions. For those who have heard of these programs, ZCPR is the Baseline System to be released into the public domain. It was derived from CCPZ Version 4.0, and is an enhancement of this. It contains many internal differences which are not readily noticable to the CCPZ user, and some new key human engineering design changes over CCPZ Version 4.0 (specifically in the SAVE and REN commands). For those of you who have been considering design changes, including the unsanctioned Version 4.1 and numerous Version 4.2 implementations (these fell out of the control of CCP-GROUP), ZCPR Version 1.0 is the released baseline which CCP-GROUP has decided upon for public release. There are several internal "problems" with CCPZ Version 4.0 which have been resolved to the satisfaction of the recent working majority of CCP-GROUP. Hence, I (and I believe the majority of the above-named CCP-GROUP subset) strongly recommend backing up to ZCPR Version 1.0 and implementing your modifications from that.

Now, getting back to everyone in general, ZCPR is the result of a Group Effort by CCP-GROUP, a somewhat closely-nit group of individuals. For the individual credits, they are listed in the ASM and DOC files, and I sincerely apologize if I omitted anyone. If I did, the corrections will be issued later.

ZCPR is the result of several months of rather diligent and fascinating work. It completely replaces the Console Command Processor of CP/M and will only run on Z80-based CP/M microcomputer systems. CP/M is required in order to run it, since a "heart" of CP/M, the BDOS, is NOT provided with ZCPR. The MAC Assembler is required to assemble it. So far, based upon both user feedback and our own experiences, we feel that ZCPR is a significant contribution to the Public Domain, and everyone who has used it greatly prefers it over standard CP/M.

ZCPR is being released for Public Distribution through the SIG/M User's Group of the Amateur Computer Group of NJ. In the spirit of Public Domain software, ZCPR is by no means a panacea, but it IS a very nice stepping stone, and you are encouraged to feel free to modify it to please yourselves. Future releases of ZCPR are quite possible, but said releases from CCP-GROUP should not happen for some time. ZCPR has been extensively tested, and, although no software can be claimed to be perfect unless it is absolutely trivial, CCP-GROUP knows of no functional errors in ZCPR Version 1.0.

For those interested in pursuing acquisition of ZCPR further, I recommend reviewing the HELP File (ZCPR.HLP in the SIG/M Release). The opening Information Section gives a fair description of some, but not all, of the interesting features supported by ZCPR and NOT found in the CP/M CCP. It is roughly 16K long and should not take too long to print out.

The next page of this message presents several displays and some information on what files compose the ZCPR system.

CCP-GROUP hopes you enjoy using ZCPR.

Richard Conn

XDIR2 Listing of All Relevant Files

```
ZCPR FILES ... ET AL
XDIR II  Version 4.8, Vertical Listing by File Type
User Number:   0, File Attributes:  Non-System

Filename.Typ Size K   Filename.Typ Size K   Filename.Typ Size K
------------ ------   ------------ ------   ------------ ------
<Disk Name >      0   MAC      .COM     12   ZCPR     .MSG     6
BDOSLOC .ASM      2   SYSGEN   .COM      2   ZCPR     .WS     40
ZCPR    .ASM     52   ZCPR     .DOC     46   ZCPRMSG  .WS      6
CPM     .BIN     12   ZCPR     .HLP     16
B:     34 Entries &     14 Files -    172K Bytes Remaining
File Data:              11 Files -    194K Bytes Displayed
```

CRCK Values for Key Files

```
File:  ZCPR     .WS              CRC = A2 C2

File:  ZCPR     .DOC             CRC = 94 B7

File:  ZCPR     .HLP             CRC = 26 9D

File:  ZCPR     .ASM             CRC = 7A 46

File:  BDOSLOC  .ASM             CRC = EB D4
```

Explanation of Files

ZCPR.ASM -- Source to ZCPR; must be assembled by MAC and customized by user for his particular system

ZCPR.HLP -- HELP File for ZCPR; can be read by HELP Version 2.0 or simply TYPEd out; read this to see a fair summary of what ZCPR can do

ZCPR.DOC -- Full Documentation on ZCPR; this includes installation notes, customization notes, and detail which expands upon the ZCPR.HLP File

ZCPR.WS -- WordStar File from which ZCPR.DOC was created

ZCPR.MSG -- Introductory message on ZCPR

ZCPRMSG.WS -- WordStar File from which ZCPR.MSG was created

BDOSLOC.ASM -- BDOS and CCP Locator; this program is handy during initial installation of ZCPR and its use is documented in ZCPR.DOC

Table of Contents

ZCPR is a replacement for the CP/M Console Command Processor (CCP) which is designed to run as part of CP/M on Z80-based microcomputers. In most cases it is upward-compatible with the original CP/M Version 2.2 CCP.

ZCPR, however, provides many extensions to the CP/M CCP. Included in these extensions are the following features:

. The TYPE function can be made to page or not page its output at the user's discretion

. A LIST function is available which sends its output to the CP/M LST: Device and does NOT page

. The DIR command has been extended to allow the display of the system files or all files

. The ERA command now prints out the names of the files it is erasing

. The current user number may be included as part of the command prompt; if the user is under a number other than 0, the prompt is of the form 'du>' (like 'A2>' or 'B10>'), and, if the user is under 0, the prompt may be 'd>' or 'd0>' as per his choice

. The SUBMIT facility has been changed in two basic ways:

- the prompt changes to 'du$' or 'd$' when the SUBMIT command is printed
- the $$$.SUB is executed from drive A: (note that the original SUBMIT problem now exists, but the new SUB.COM facility corrects it); the CCP-GROUP definition of an Indirect Command File now applies, and this definition is that any sequence of commands which may be issued from the console is also a valid sequence of commands for execution from an Indirect Command File; hence, the sequence:

```
DIR
B:
DIR
A:
```

may be issued from either the console or an Indirect Command File, and the results of the execution of this sequence are the same. Basically, this says that Indirect Command Files are upward-compatible to the console input (but not necessarily that the contents of an Indirect Command File may be issued at the console without modification).

. A command-search hierarchy is now implemented which is executed roughly as follows:

- the user's command is checked against the CPR-resident commands and executed immediately if a match is found

- failing that, the current user number on the current disk is scanned for the COM file; the COM file is loaded and executed if found

- failing that, a default user number (initially 0 but can be reset with the DFU CPR-resident command) on the current disk is scanned for the COM file; the COM file is loaded and executed if found

- finally, failing that, the default user number on disk A: is scanned for the COM file; the COM file is loaded and executed if found or an error message (COMMAND?, when COMMAND was the user's command name) is printed

. The numeric argument for the SAVE command can be specified in hexadecimal so that the user may employ the values presented by tools such as DDT exactly as they are given

. A GET command which loads a file at a specified memory address and a JUMP command which "calls" the subroutine at a specified memory address have been added; a GO command which "calls" the subroutine at 100H (subset of the JUMP capability) has also been added

This document provides the user of ZCPR with the following information:

Part A: Installation Instructions
Part B: Usage Instructions and Explanation of Commands
Part C: ZCPR Command Levels and How to Use Them

Part A
Installation Instructions

In order to install ZCPR on a target microcomputer (must be currently running CP/M 2.2), the user must know two basic things:

1) Where his CCP is currently running in memory
2) Where his CCP is located in the SYSGEN image, or, for systems which don't support SYSGEN (such as P&T CP/M 2.2 for the TRS-80 Model II), where his CCP is located on disk and how to place the new ZCPR on top of it

The first question is answered relatively easily. A program, known as either BDOSLOC or BDLOC (for BDOS Locator), is provided with ZCPR. You should assemble this program for your particular computer (change the base ORG if you are running non-ORG-0 CP/M) and execute it. Upon execution, it will provide you with the base address of (1) the BDOS and (2) the CCP for your particular system. BDOSLOC has worked correctly for all systems tested so far, but there is always a chance that it may NOT work for some non-tested system. For the time being, assume that it works correctly and record the starting base page address of your CCP.

The second question is not answered nearly so easily. If you have the ability to SYSGEN your system, it is much easier (commonly) than if you do not. You must, after assembling the ZCPR properly, integrate it into the sysgen (or disk) image of CP/M. This can be done by obtaining a SYSGEN image of your system, scanning it via a debugger such as DDT to find the offset for the CCP, reading the new CPR in on top of the old one, and finally running SYSGEN again to place the resultant system on disk. If you DO NOT have SYSGEN capability, a Disk Utility program is required to locate the CCP on disk and then write the new ZCPR on top of the old one. The net result of this integration is the placement of the new ZCPR onto disk in the proper place so that it will be loaded with the rest of CP/M on cold boot and executed properly.

To find the original CCP, you typically have to locate it by its appearance. It is probably stored contiguously on disk, so, once it is found, a sequential overwrite is all that is required. Probability is extremely high that it is stored contiguously in the SYSGEN image. The CCP starts with two (2) and ONLY TWO jump instructions followed by a buffer area (possibly containing an initial command and/or the Digital Research copyright notice). The Digital Research manuals show the CCP to reside at address 980H in the SYSGEN image, but this may vary with system. To find this image, use DDT or some other such debugger, load the SYSGEN image you can get via SYSGEN, and examine memory starting at around 900H for the two (and ONLY two) jumps described above. If you find an area with more than two jumps (a group of them), you are probably looking at the BIOS and should go lower for the CCP. The CCP will probably start on an even page or half-page address (like 900H, 980H, 1100H, etc).

Now that the location of the CCP has been found, record this address for later. You are now ready for the integration of ZCPR into your system. To do this, perform the following steps using the information of the page address of the CCP (obtained from BDOSLOC and called CPRLOC within ZCPR) and the SYSGEN image address of the CCP (called IMAGE for reference in this document).

1. Edit ZCPR and set the CPRLOC equate to the value obtained from above. Also set any flags and values as you desire (see the section on ZCPR Customization below). When satisfied, end the edit session.

2. Assemble ZCPR with MAC (or equivalent). This assembler is required because of the MACROs used. Only the resultant HEX file is required.

3. Assuming that you can use SYSGEN, obtain a SYSGEN image of your current CP/M system and save it on disk.

4. Load the SYSGEN image into memory with DDT (or equivalent). Once loaded, verify that the original CCP is at the IMAGE address found above and compute the integration offset using the DDT H command:

H<IMAGE adr>,<CPRLOC adr>

The second number displayed gives you the OFFSET value required for step 5.

5. Integrate ZCPR into your SYSGEN image via DDT's I and ROFFSET commands. Use IZCPR.HEX (or the name of your version of ZCPR) to load the FCB and ROFFSET (where OFFSET was computed in step 4) to load the ZCPR.HEX file into memory at the proper location. Check to see that ZCPR is indeed properly loaded by examining the SYSGEN IMAGE area.

6. Place the new system on disk by running SYSGEN and NOT loading the system from disk (use the memory image).

For further clarification of the above process, the following is a sample terminal session which outlines the steps taken.

ZCPR Integration Example

```
B>; Sample terminal session for integrating ZCPR
B>sysgen
SYSGEN VER 2.2
SOURCE DRIVE NAME (OR RETURN TO SKIP)b
SOURCE ON B, THEN TYPE RETURN  <-- I hit the RETURN key here
FUNCTION COMPLETE                                  /
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)  <-- and here
B>save 44 cpm56.com       <-- We now have a SYSGEN image of CP/M
                                  to work with
```

```
B>xdir
XDIR   Version 2.6  User Number:   0,  Double Density
File Attributes:  Non-System

Filename.Typ Size K   Filename.Typ Size K   Filename.Typ Size K
------------ ------   ------------ ------   ------------ ------
!TEXTWRK.-12      0   CPR      .DOC      8   EE687    .TXT      4
CPR      .AQM    34   TFS      .HLP      6   EE687PRE.TXT       4
CPR      .ASM    50   CONTENTS.T01       6   SW1      .TXT     10
CPR      .BAK     4   CONTENTS.T02       4   SW2      .TXT      2
CPM56    .COM    12   CONTENTS.T03       4
B:  30 Entries &   22 Files --     338K Bytes Remaining
File Data:         14 Files --     154K Bytes Displayed
B>bdosloc       <-- Now to locate the CCP's address
The Base Page Address of this system's BDOS is C5
The Base Page Address of this system's CCP is  BD <-- This is it
B>ddt cpm56.com      <-- Now to find the CCP in the SYSGEN image
DDT VERS 2.0
NEXT  PC
2D00 0100
-d900,90f       <-- Start looking around here
0900 31 80 E7 3E 06 3C 3C FE 1B CA 00 C2 DA 11 E7 D6 1..>.<<.........
-da00,a0f
0A00 31 00 01 01 01 0C C5 CD 0F E4 21 00 BE 11 00 04 1.........!.....
-db00,b0f
0B00 31 00 01 01 01 11 C5 CD 0F E4 21 00 C0 11 00 02 1.........!.....
-db80,b8f
0B80 31 00 01 01 09 01 CD A8 00 21 00 D2 11 00 C2 0E 1........!......
        -- Detail Left Out --
-d1100          <-- I found it at 1100H; note the 2 JMP's
1100 C3 FF BD C3 FB BD 50 10 20 20 20 20 20 20 20 20 ......P.
1110 20 20 20 20 20 20 20 20 00 00 00 00 00 00 00 00          ........
1120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
        -- Detail Left Out --
-^C             <-- Return to CP/M; I know that CPRLOC will be
                    BD00H and the IMAGE offset is 1100H

B>ed cpr.asm {edit ZCPR here and place CPRLOC=BD00H}#
        -- Detail Left Out --

B>mac cpr $pz sz          <-- Now to assemble the CPR
CP/M MACRO ASSEM 2.0
C4F0                      <-- Note that CPR MUST end before BDOS
                              begins!

014H USE FACTOR
END OF ASSEMBLY

B>ddt cpm56.com           <-- Now to integrate!
DDT VERS 2.0
NEXT  PC
2D00 0100
-h1100,bd00               <-- Compute offset for new CPR
CE00 5400                 <-- Offset is 5400H
-icpr.hex                 <-- Init FCB
```

```
-r5400                  <-- Read in new CPR with offset
NEXT  PC
2D00 0000
-^C                     <-- Done!
B>sysgen                <-- Now to SYSGEN onto disk
SYSGEN VER 2.2
SOURCE DRIVE NAME (OR RETURN TO SKIP)    <-- Use memory image
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)b      <-- onto B:
DESTINATION ON B, THEN TYPE RETURN
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <-- Done for now

B>
```

Setting the ZCPR Inline Options

The following are the four basic options available to the user under ZCPR for customization of his package.

Option Name	Function
REL	Configures CPRLOC (CPRLOC equ 0) for integration via MOVCPM rather than the DDT/SYSGEN technique outlined above; set to TRUE for MOVCPM integration or FALSE for DDT/SYSGEN integration
BASE	Base address of your CP/M system; standard CP/M has a base of 0, but some CP/M systems (such as for the TRS-80 Model I and Heath/Zenith H89/Z89) start physical RAM memory at a higher address; equate BASE to the starting RAM memory address of your system
CPRLOC	This is the starting address of ZCPR; set the second CPRLOC equate to the address you obtain from BDOSLOC
RAS	This is an equate which masks out selected ZCPR command functions for security purposes on Remote Access Systems such as Bulletin Boards; the masked out functions currently include SAVE, ERA, REN, JUMP, GO, and GET; set RAS to TRUE to mask these out or FALSE to leave them in
SUBA	This is an equate which determines the drive onto which ZCPR will look for an executing Indirect Command File. If the basic philosophy of the Indirect Command File described above is to be maintained, this symbol should be set to TRUE (look on drive A: for the $$$.SUB file); if not, this symbol should be set to FALSE (look on the default drive from the $$$.SUB file). To review, the basic philosophy of the Indirect Command File is that any sequence of commands which may be issued from the console (within reason, which means NOT to erase a $$$.SUB file) may also be issued from within an Indirect Command File, and the resultant execution should be identical (same functions performed).
CLEVEL3	This equate enables or disables extended Command Level 3 Processing. If set to TRUE, extended Command Level 3 Processing is enabled and the user command line is automatically capitalized, the terminating zero is placed at the end of the buffer, and the internal CIBPTR is set correctly (see later for more information).

Customization Symbols

The following symbols are provided for further customization of ZCPR to a user's particular tastes and hardware facilities.

Option Name	Function
NLINES	Number of lines on the user's CRT for paging
WIDE	This equate is used to select a narrow or wide display under the DIR command; if WIDE is equated to TRUE, each file name is separated by two spaces, a FENCE, and two more spaces; if WIDE is equated to FALSE, each file name is separated by one space, a FENCE, and one more space
PGDFLT	This is the Paging Default flag for the TYPE command; if PGDFLT is set to TRUE, the TYPE command will page its output by default and the P option on the TYPE command (see below) will prohibit paging; if PGDFLT is set to FALSE, the TYPE command will NOT page its output by default and the P option will enable paging
PGDFLG	This sets the option character in the command line for the TYPE command (the 'P' mentioned above); if the user wishes to change this option character, he need only change this equate
MAXUSR	This is the largest user number recognized by the USER command; if the user wishes to protect the higher user areas, he may set this symbol to the highest area normally accessable; 15 is the largest permitted value for MAXUSR
SYSFLG	This is the option character for the DIR command line which is used to specify that DIR search All Files (both $SYS and $DIR) for its display; the distributed default for this is 'A'
SOFLG	This is the option character for the DIR command line which is used to specify that DIR search ONLY the $SYS files for its display; the distributed default for this is 'S'
SUPRES	Set SUPRES to TRUE to suppress printing the user number when the user is under User Number 0 or set SUPRES to FALSE to ALWAYS display the User Number with the CPR prompt; with SUPRES set to TRUE, a user on B: in user 0 sees 'B>' as the prompt, but with SUPRES set to FALSE, a user on B: in user 0 sees 'B0>' as the prompt

DEFUSR — This is the CPR-default user number which is searched in the command hierarchy for the COM files (distributed as 0); the DFU changes this temporarily until a Warm Boot or Cold Boot is done, at which time the search reverts to this value

SPRMPT — This is the CPR prompt character which indicates that a SUBMIT file is in execution; by default it is set to '$', so prompts like 'A$' appear during SUBMIT file execution

CPRMPT — This is the CPR prompt character which indicates that the CPR is awaiting a user console command; by default it is set to '>', so prompts like 'A>' appear during user input to the CPR

NUMBASE — This is the escape character used by those commands which require a DECIMAL number as an argument; placing this character after the number argument switches the base to HEXADECIMAL; for example, 'SAVE 15 MYFILE' can be expressed as 'SAVE FH MYFILE' if NUMBASE is set to 'H' (the default)

SECTFLG — This character constant is the suffix option for the SAVE command which specifies that sectors, as opposed to pages, are to be saved; the default value is 'S'

FENCE — This is the character printed to separate entries in a directory listing; it's default value is '|'

Patching SUBMIT.COM

SUBMIT.COM may be patched to run with ZCPR by the following procedure (this is recommended if the user does not have SUB.COM). This patch simply makes it always place the $$$.SUB file on Drive A:. Illustrative terminal session follows:

```
A>ddt b:submit.com
DDT VERS 2.0
NEXT  PC
0600 0100
-s5bb            <-- Patch is at 5BB Hex
05BB 00 1        <-- Change 0 (default drive) to 1 (drive A:)
05BC 24 .        <-- That's it!
-d5b0 5cf        <-- See change
05B0 00 00 00 00 00 00 30 30 31 20 24 01 24 24 24 20 ......001 $.$$$
05C0 20 20 20 20 53 55 42 00 00 00 1A 1A 1A 1A 1A 1A     SUB.........
-^C              <-- Done
A>save 5 newsubmt.com   <-- Save new SUBMIT.COM file
```

Part B
Usage Instructions and Explanation of Commands

The following instructions are written with the assumption that the reader is quite familiar with how to use CP/M 2.2 and its CCP. ZCPR is written as a logical extension of the CP/M 2.2 CCP philosophy and should be addressed as such.

The ZCPR Command Hierarchy Search

The first, and most basic thing, to learn about ZCPR is the order in which is searches for a COM file for execution or a file specified by the GET command. Under the CP/M 2.2 CCP, if the specified COM file command was not found on the current drive in the current user area, the CCP aborted with an error message. ZCPR, however, continues searching from this point a maximum of two more levels. This command hierarchy search was outlined above and is described here in further detail.

1. If the command is of the form 'COMMAND' and NOT 'd:COMMAND', the CPR-resident command list is searched for a match. If the match is found, the CPR-resident command is immediately processed. If the match is not found or the command is of the form 'd:COMMAND', the next step is taken. Note that the 'd:COMMAND' form is good for executing a command COM file which has the same name as a CPR-resident command (such as SAVE or DIR).

2. If the command is of the form 'd:COMMAND', disk drive 'd:' is temporarily logged in for the purpose of the command search. Otherwise, the currently logged-in drive is used.

3. Now the file named COMMAND.COM is searched for. If found, it is loaded into memory starting at 100H and executed. If not, proceed to step 4.

4. Now that the first search for COMMAND.COM has failed, the CPR checks to see if the user is under the current Default User Number. The Default User Number may be that set by the DEFUSR equate in the CPR or that set by the user via the DFU command. DEFUSR is in effect if DFU has not been issued since the last Warm or Cold Boot, and DFU is in effect if it was issued since the last Warm or Cold Boot. If the user is NOT under the current Default User Number, ZCPR temporarily logs him into it and searches the directory. If COMMAND.COM is found, it is loaded as described above and executed. If not, ZCPR proceeds to the next step.

5. The user is now in the Default User Number, and at this point, ZCPR checks to see if the user is on disk drive A:.

If not, it temporarily logs into A: and searches the default user number of A: for COMMAND.COM. If found, it is loaded as described above and executed. If not, ZCPR prints the command name as an error message and returns to command input mode, aborting the SUBMIT file if COMMAND came from it.

In all cases of the search above, if COMMAND.COM is found, after it is loaded into memory, ZCPR resets the user to his original disk drive and user number. Hence, the files referenced by the user by default are obtained from this environment.

To illustrate this command hierarchy search, consider the following examples:

Example 1: DEFUSR equ 0 {default user number is 0}

```
B10>            <-- User is on Drive B:, User Number 10
B10>ASM TEST.BBZ    <-- User wishes to assemble TEST.ASM in
                        Drive B:, User 10
     <-- At this point, ZCPR looks on B:/10 for ASM.COM, fails,
         looks on B:/0, fails, and finally looks on A:/0; it
         finds ASM.COM here and goes back to B:/10 for the file
```

Example 2: DEFUSR equ 0 and DFU issued

```
B10>            <-- User is on Drive B:, User Number 10
B10>DFU 5       <-- User Selects User 5 as default
B10>ASM TEST.BBZ    <-- As above
     <-- At this point, ZCPR looks on B:/10 for ASM.COM, fails,
         look on B:/5, fails, and finally looks on A:/5; it
         fails here also and prints ASM? as an error message
```

Example 3: DEFUSR equ 0

```
B>              <-- User is on Drive B:, User Number 0
B>ASM TEST.BBZ      <-- As above
     <-- At this point, ZCPR looks on B:/0 for ASM.COM, fails,
         looks on A:/0, fails, and prints error message
```

Example 4: DEFUSR equ 0

```
A10>            <-- User is on Drive A:, User Number 10
A10>ASM TEST.AAZ    <-- As above, but file on A:
     <-- At this point, ZCPR looks on A:/10 for ASM.COM, fails,
         looks on A:/0, fails, and prints error message
```

Another Example:

For example, if the user is logged into Drive B: in User Area 10, the Default User Number is 0, and the following COM files are present as indicated --

WM.COM on Drive A: in User 0

MBASIC.COM on Drive A: in User 0 and on
Drive B: in User 0
TEST.COM on Drive B: in User 10 and Drive B:
in User 0

then the following happens when the following commands are issued from the console (or Indirect Command File):

```
B10>WM TEST2.TXT
 \   \    \__ File to be edited
  \   \__ Invoke the WM.COM file (Word Master editor)
   \__ User is on Drive B: in User Area 10
```

Results:

ZCPR searches B: User 10, B: User 0, and A: User 0 for WM.COM; it finds WM.COM in A: User 0, loads it, logs the user back into B: User 10, and executes it.

```
B10>MBASIC
 \    \__ Invoke the MBASIC.COM file (MBASIC Interpreter)
  \__ User is on Drive B: in User Area 10
```

Results:

ZCPR searches B: User 10 and B: User 0 for MBASIC.COM; it finds MBASIC.COM in B: User 0, so it doesn't bother to look on A: User 0. MBASIC.COM is then loaded and executed as described in the previous example.

```
B10>TEST
  \   \__ Invoke the TEST.COM file (TEST program)
   \__ User is on Drive B: in User Area 10
```

Results:

ZCPR searches B: User 10 for TEST.COM; it finds TEST.COM in B: User 0, so it doesn't bother to look further (if it had, it would have found TEST.COM in B: User 0). TEST.COM is then loaded and executed as described above.

```
B10>TEST2
  \    \__ Invoke the TEST2.COM file (TEST2 program)
   \__ User is on Drive B: in User Area 10
```

Results:

ZCPR searches B: User 10, B: User 0, and A: User 0 for TEST2.COM; it doesn't find it, so it issues the error message 'TEST2?', which says it couldn't find TEST2.COM.

The following pages describe the ZCPR-Resident Commands. These are commands located within ZCPR itself which are executed from within ZCPR. The phrases <afn> and <ufn> refer to ambiguous file name and unambigous file name as per the CP/M convention.

```
Command: DIR
Function: To Display a listing of the names of the files on disk
Forms:
     DIR <afn>           <-- Displays $DIR files
     DIR <afn> S         <-- Displays $SYS files
     DIR <afn> A         <-- Displays both $DIR and $SYS files
Customization Variables:
     WIDE      SYSFLG    SOFLG     FENCE
Examples:
     DIR *.ASM           <-- All $DIR .ASM files
     DIR *.COM S         <-- All $SYS .COM files
     DIR *.COM A         <-- All .COM files
Notes:
```

If a file is scanned for and no such name exists on disk, the 'No Files' message will appear. However, if a file is scanned for and the name exists as a $SYS file and $DIR files are being scanned for, no file name is displayed but the 'No Files' message does NOT appear. For example, if TEST.COM is a $SYS file and 'DIR TEST.COM' is issued, no message appears. If 'DIR TEXT.COM' is issued and TEXT.COM does not exist on disk, the 'No Files' message is displayed.

```
Command: ERA
Function: To Erase the specified $R/W files from disk
Forms:
     ERA <afn>           <-- Erase both $DIR and $SYS files
Customization Variables:
     WIDE      FENCE
Examples:
     ERA *.ASM           <-- Erase all .ASM files
     ERA *.*             <-- Erase all files
Notes:
```

If a $R/O file is encountered, a BDOS error message will be displayed and the procedure is stopped. The user is unsure at this time as to which files have been erased and which have not and should check. Sorry for this problem! The ERASE command (to be given to SIG/M by RLC in the near future) is a solution to this problem.

```
Command: LIST
Function: To Print the specified file on the CP/M LST: device
Forms:
     LIST <ufn>          <-- Print the file (no paging)
Customization Variables:
     -None-
```

```
Examples:
     LIST TEST.TXT        <-- Print TEST.TXT on LST:
Notes:
     If the file has a $SYS attribute, it will be found as well
as those with $DIR attributes.


Command: TYPE
Function: To Print the specified file on the CP/M CON: device
Forms:
     TYPE <ufn>           <-- Print the file with the paging deflt
     TYPE <ufn> P         <-- Print the file with the paging deflt
                              negated
Customization Variables:
     NLINES    PGDFLT    PGDFLG
Examples:
     TYPE TEST.TXT
     TYPE TEST.TXT P
Notes:
     When the display pauses during paging, type any char to
continue or ^C to abort. ^S also works.


Command: SAVE
Function: To Copy the TPA starting at 100H to disk
Forms:
     SAVE <Number of Pages> <ufn>  <-- <Number of Pages> in DEC
     SAVE <Number of Pages>H <ufn> <-- <Number of Pages> in HEX
     SAVE <Number of Sectors> <ufn> S  <-- Number of sectors
     SAVE <Number of Sectors>H <ufn> S <-- Number of sectors
Customization Variables:
     NUMBASE   RAS
Examples:
     SAVE 15 MYFILE.TXT         <-- 15 pages saved
     SAVE FH MYFILE.TXT         <-- 15 pages saved
     SAVE 10H MYFILE.TXT S      <-- 16 sectors (8 pages) saved
Notes:
     If the file name to be saved already exists, then SAVE will
exit with the message 'Delete File?'; if the user REALLY wants to
save under this name, he may then type Y or y and the current
file will be deleted and then recreated containing the specified
part of the TPA.

Command: REN
Function: To Change the name of a disk file
Forms:
     REN <ufn new>=<ufn old>
Customization Variables:
     RAS
Examples:
     REN NEWFILE.TXT=OLDFILE.TXT
Notes:
     If <ufn new> already exists, the message 'Delete File?' will
be printed and the user may respond with Y or y to delete the
```

current <ufn new> and then rename <ufn old> to <ufn new>.

```
Command: USER
Function:  To Change the current user number
Forms:
     USER <User Number>        <-- <User Number> in DEC
     USER <User Number>H       <-- <User Number> in HEX
Customization Variables:
     -None-
Examples:
     USER 15         USER FH         USER 0
     USER       <-- Same as USER 0
Notes:
     -None-


Command: DFU
Function:  To Temporarily Change the default user number for the
           command hierarchy search
Forms:
     DFU <User Number>         <-- <User Number> in DEC
     DFU <User Number>H        <-- <User Number> in HEX
Customization Variables:
     -None-
Examples:
     DFU 15          DFU FH          DFU 0
     DFU        <-- Same as DFU 0
Notes:
     See above for explanation.


Command: JUMP
Function:  To "call" the subroutine at the specified page address
Forms:
     JUMP <Address> <Cmd Parms> <-- <Address> in HEX
Customization Variables:
     NUMBASE   RAS
Examples:
     JUMP E000 or JUMP E000H  <-- Jump to E000H
     JUMP                     <-- Jump to 000H
     JUMP 0                   <-- Jump to 000H
Notes:
     JUMP performs a subroutine "call", so the called routine may
return to the ZCPR by either a RET or a Warm Boot.


Command: GO
Function:  To "call" the subroutine starting at 100H
Forms:
     GO <Cmd Parms>            <-- Execute reentrant at 100H
Customization Variables:
     RAS
```

Examples:

```
     GO *.ASM                    <-- Assuming XDIR is loaded,
                                     gives directory of *.ASM
```

Notes:

This command is identical in function to JUMP 100H; JUMP, however, leaves the address as the first entry in CP/M BASE + 80H (the input line buffer), while GO has no such address.

Command: GET
Function: To load a file from disk into memory starting at the specified page
Forms:

```
     GET <Address> <ufn>         <-- <Address> in HEX
```

Customization Variables:

```
     NUMBASE   RAS
```

Examples:

```
     GET 8000 TEST.80            <-- Load TEST.80 starting at 8000H
     GET 100 TEST.80 or GET 100H TEST.80 <-- Load TEST.80
                                     starting at 100H
     GET 0 TEST.80               <-- Load TEST.80 starting at 000H
```

Notes:

GET searches for the specified file according to the same command hierarchy search employed by the ZCPR command scanner. Hence, if the user is on B:/10 and the file is on A:/0 with the current default user number at 0, GET will search from B:/10 to B:/0 to A:/0 in looking for the file.

ZCPR Error Messages

The following are the error messages issued by ZCPR and their meanings.

Message	Meaning
?	Printed after a command or an argument means that such was invalid
No File	From DIR, this means that DIR did not locate any files Also from ERA with the same meaning
All?	Issued in response ERA *.*, asks the user is he really wants to erase all the files. Unlike under the original CP/M 2.2 CCP, single character input is required (Y or y for yes and anything else for no) with NO <CR> to end the line
Full	From SAVE, means that there is not enough space on disk From GET or command load by CPR, means that there is not enough space in memory
Delete File?	From REN or SAVE, means that the file specified already exists on disk and the user may type Y or y to delete it and proceed with the REN or SAVE function

Part C
ZCPR Command Levels and How to Use Them

ZCPR Version 1.0 and beyond supports three distinct command levels in its implementation. Each level constitutes a different way to issue a command for ZCPR to process.

Command Levels 1 and 2 are common to all implementations of CP/M and CP/ZM from CP/M Version 1.4. Command Level 1 is that command level in which the command is issued by the user from his console terminal. The prompt 'd>' or 'du>' appears on the terminal, and the user is allowed to enter the command with editing from the terminal. Command Level 2 is that command level in which the command is entered from an executing $$$.SUB file.

In both cases, the command is stored in the internal ZCPR buffer called CIBUFF (Command Input BUFFer). Under both Command Levels 1 and 2, the command is placed into this buffer, the characters of the command line are capitalized, a character count which indicates the number of characters in the command line is stored in CBUFF (the byte before CIBUFF), an ending binary 0 is placed after the last character in the command line, and the internal pointer CIBPTR (Command Input Buffer PoinTeR) is set to point to CIBUFF (the first character of the command line).

Command Level 3 is an extended concept to Command Levels 1 and 2 which is specifically supported by ZCPR Version 1.0 and beyond. This command level allows a transient program to place a command line into CIBUFF and the character count into CBUFF and have this command line executed by ZCPR. Once control is transferred to ZCPR to execute the command line, the transient program which placed the command line loses control and the command is executed exactly as though it had been typed by the user at his console terminal.

In order for a transient program to utilize the Command Level 3 facility, this program MUST do the following:

1. Locate the ZCPR. Since the ZCPR is ALWAYS 2K bytes in size and located directly under the BDOS, the transient can locate the ZCPR by examining the BDOS entry page address at location 7 and subtracting 8 from this number (8 pages = 2K bytes). The resulting number is the base page address of ZCPR.

2. Store the command line in CIBUFF and the character count in CBUFF. Knowing the base page address of ZCPR, the following information is useful in doing this:

```
        ORG   CPRLOC    ;Base Address of ZCPR
        JMP   CPR       ;Enter ZCPR and Execute Default Cmd
        JMP   CPR1      ;Enter ZCPR and Don't Execute
MBUFF:  DB    BUFLEN    ;Size of CIBUFF in bytes
CBUFF:  DS    1         ;Number of Bytes in Command Line
```

```
CIBUFF:   DS   BUFLEN    ;Buffer for Command Line
          DS   1         ;Buffer for Ending 0 (set by ZCPR)
CIBPTR:   DS   2         ;Address of CIBUFF (set by ZCPR)
```

3. Obtain the User/Disk Flag. Location 4 contains this number, but the user may select a flag of his choice. This flag is one byte long, and the high-order nybble (4 bits) contains the user number and the low-order nybble contains the disk number to process the command from. The User/Disk Flag is to be passed to ZCPR in the C Register.

4. When ready, transfer control to ZCPR to process the command by JMPing to the base address of ZCPR. The first JMP in the JMP Table given above is at this address. At this time, ZCPR will log in the user and disk in the User/Disk Flag and process the Command Level 3 Command Line.

The following is a sample program which illustrates the steps outlined above:

```
;
;  Demonstration of Command Level 3 Facility by RLC
;
udflag  equ     4       ;Address of User/Disk Flag
bdos    equ     5       ;Address of BDOS Entry Point

        org     100h

        lxi     d,prmpt ;Print User Prompt
        mvi     c,9     ;PRINT function
        call    bdos

        lhld    bdos+1  ;Get address of BDOS
        mov     a,h     ;High-Order Address in A
        sui     8       ;A=High-Order Address of CPR
        mov     h,a     ;HL=Address of CPR
        mvi     l,0
        shld    cpr     ;Save address in buffer

        lxi     d,6     ;Point to command line buffer
        dad     d       ;HL points to command line buffer
        xchg            ;DE points to command line buffer
        mvi     c,10    ;READLN into this buffer
        call    bdos

        lhld    cpr     ;Get Address of CPR
        lda     udflag  ;Get User/Disk Flag
        mov     c,a     ; ... in C
        pchl            ;Run Command Line

cpr:    ds      2       ;CPR Address buffer
prmpt:  db      'User Command? $'
```

```
        title   'NZCPR v2.1 as of 12/28/82'
;
;  CP/M Z80 Command Processor Replacement (CCP) Version 2.1 in the NZCPR line.
;       To reconstruct the full file from NZCPR-17.DIF do the following:
            SSED2 ZCPR.ASM <NZCPR-21.DIF >NZCPR-21.ASM
                (where ZCPR.ASM is version 1.0)
;
;Note the name change: this was NZCPR-17, but Peter Pinchis came out with
;NZCPR-20 that was based on NZCPR-13 (...grrr....sigh)  <pst>
;
;*************************************************************************
;  ZCPR version 1.0 was created from CCPZ version 4.0 by RLC in a coordinated
; effort with the CCP-GROUP.  ZCPR was a group effort by CCP-GROUP, whose
; active membership involved in this project consisted of the following:
;    RLC - Richard Conn          KBP - Keith Peterson
;    RGF - Ron Fowler            FJW - Frank Wancho
; The following individuals also provided a contribution:
;    SBB - Steve Bogolub         PST - Paul Traina
;
;  Since RLC decided that ZCPR v1.0 was the last official version
; sanctioned by the CCPZ group, changes beyond that point are being
; called by consensus of a group of new changers "NZCPR Vx.x". The
; following individuals have put in their code or opinions:
;
; SBB - Steve Bogolub   The person that keeps ZCPR neat and not a chaotic mess
;                       (fixes my (PST's) bugs and writes great hacks too)
;                       (v1.1S, v1.5S (?), v1.6)
; PST - Paul Traina     The silly person behind SECURE mode. Numerous minor
;                       re-hacks to make life easier. (a bunch o'bugs too..)
;                       (v1.1, v1.3, v1.4, v1.5, v1.7, v2.1)
; HLB - Howard Booker   Cleaned up code, improved INPASS routines (v1.2)
;
; CAF - Chuch Forsberg  RAF - Bob Fischer       BB  - Ben Bronson
; PRG - Paul Grupp      PJH - Paul Homchick     HEW - Hal Walchli
; DR  - Dave Roznar     HK  - Harry Kaemmerer   SFK - Sigi Kluger
; PP  - Peter Pinchis
;
;   In an attempt to maintain a link to the past, changes between the
; current version of NZCPR are provided as both a difference file
; between NZCPR's (NZ16-17.DIF) and as a difference between the current
; version and the "official" ZCPR V1.0 (NZCPR-17.DIF).  These changes
; are made and supported by individuals in contact with each other through
; the Sysop CBBS (in the East) and OxGate-001 (in the West). Make comments or
; complaints there, to PST or SBB (or anyone else interested).
;
;   The most obvious differences between NZCPR and ZCPR are the security
; features, controlled by additional conditional assembly flags. Such
; features restrict access to ZCPR intrinsic commands, add additional
; levels of .COM file searching, and prevent access to higher drives
; or user levels, with either internal or external password control of
; these features. Less obvious differences involve code optimization to
; gain space, and some minor bug fixes in the TYPE command.
;
;******** Structure Notes ********

   NZCPR is divided into a number of major sections.  The following
; is an outline of these sections and the names of the major routines
; located therein.
;
; Section       Function/Routines
```

```
;  -------        ----------------
;
;    --           Opening Comments, Equates, and Macro Definitions
;
;     0           JMP Table into CPR
;
;     1           Buffers
;
;     2           CPR Starting Modules
;                         CPR1    CPR     RESTRT  RSTCPR  RCPRNL
;                         PRNNF   CMDTBL
;
;     3           Utilities
;                         CRLF    CONOUT  CONIN   LCOUT   LSTOUT
;                         READF   READ    BDOSB   PRINTC  PRINT
;                         GETDRV  DEFDMA  DMASET  RESET   BDOSJP
;                         LOGIN   OPENF   OPEN    GRBDOS  CLOSE
;                         SEARF   SEAR1   SEARN   SUBKIL  DELETE
;                         RESETUSR GETUSR SETUSR  PAGER   UCASE
;                         NOECHO
;
;      4          CPR Utilities
;                         SETUD   SETUOD  REDBUF  CNVBUF  CMDSER
;                         BREAK   USRNUM  ERROR   SDELM   ADVAN
;                         SBLANK  ADDAH   NUMBER  NUMERR  HEXNUM
;                         DIRPTR  SLOGIN  DLOGIN  COMLOG  SCANER
;
;      5          CPR-Resident Commands and Functions
;      5A                 DIR     DIRPR   FILLQ
;      5B                 ERA
;      5C                 LIST
;      5D                 TYPE
;      5E                 SAVE
;      5F                 REN
;      5G                 USER
;      5H                 DFU
;      5I                 JUMP
;      5J                 GO
;      5K                 COM     CALLPROG        ERRLOG  ERRJMP
;      5L                 GET     MEMLOAD PRNLE
;      5M                 PASS    NORM
;
;
FALSE   EQU     0
TRUE    EQU     NOT FALSE
;
;  CUSTOMIZATION EQUATES
;
;  The following equates may be used to customize this CPR for the user's
;    system and integration technique.  The following constants are provided:
;
;    REL - TRUE if integration is to be done via MOVCPM
;        - FALSE if integration is to be done via DDT and SYSGEN
;
;    SECURE -  TRUE to conditionally disable potentially-harmful
;              commands (GO, ERA, SAVE, REN, DFU, GET, JUMP). Under
;              SECURE, if WHEEL contains RESTRCT, do not accept those
;              commands, and search for COM files under current user
;              then user "DEFUSR" only. If WHEEL does not contain
;              RESTRCT (presumably from passworded change), allow
```

```
;               all commands, and search current user, then last user
;               set by DFU (originally "RESUSR"), then user "DEFUSR"
;               for COM files, giving access with password to an
;               additional level of COM files.
;
;               (Note: WHEEL must point to a safe place in memory that
;                won't be overlayed)
;
;       If you have chosen a SECURE system,  all resident commands may be
; activated by entering:  PASS <password> <cr>  Where <password> is a sequence
; of characters placed at PASSID (if INPASS is true, otherwise, see
; documentation in PST's PASS.ASM).  If the password is incorrect. the system
; will come back with PASS? as if it was looking for a COM file.
;       NORM is the reverse of PASS, it will disable the WHEEL mode.
;
;    INPASS -  If in the SECURE mode, you wish to use a program similar
;              to PST's PASS.ASM, set this false, otherwise, ZCPR will
;              handle the PASSword coding with a built in command.
;
;    DRUSER -  Set this EQU false if you wish to disable RAF's neat hack
;              that allows you the type B: 7 to move to drive B: user area
;              seven.  This also removes the USER command.  Basically, set
;              this equate false if you want to use USERPW or some other pgm.
;
;    RAS    -  Remote-Access System; setting this equate to TRUE disables
;              certain CPR commands that are considered harmful in a Remote-
;              Access environment; use under Remote-Access Systems (RBBS) for
;              security purposes.  Note: SECURE is the direct enemy of RAS,
;              DON'T define both equates or you will be VERY sorry.
;              The advantage SECURE has over RAS is that by saying a magic
;              word, all of the normal commands pop into existance.
;
;    MAXDRIV - Maximum legal drive number stored in this location.
;              (0 means only A:, etc.)  0000H disables this feature.
;              The value MAXDR is stuffed into MAXDRIV at cold boot,
;              and presumably will be changed later by a passworded
;              program if desired.
;
;              (This code is in addition to BIOS checks. It's needed here
;              because X: can hang if X: is off line in some BIOS
;              implementations. Personally, I think CAF and others should fix
;              their BIOS instead. Mine works right...SBB).
;
;    USRMAX -  Maximum legal user # + 1 stored in this location. 0000H
;              disables this feature, and uses the value of MAXUSR+1 instead.
;
;    BASE   -  Base Address of user's CP/M system (normally 0 for DR version)
;              This equate allows easy modification by non-standard CP/M (eg,H89)
;
;    CPRLOC -  Base Page Address of CPR; this value can be obtained by running
;              the BDOSLOC program on your system, or by setting the
;              MSIZE and BIOSEX equates to the system memory size in
;              K-bytes and the "extra" memory required by your BIOS
;              in K-bytes. BIOSEX is zero if your BIOS is normal size,
;              and can be negative if your BIOS is in PROM or in
;              non-contiguous memory.
;
;    EPRMPT -  Set TRUE to be prompted "Ok?" after seeing what files will
;              be erased. No, this is NOT for individual file prompting,
;              it is just to confirm deletion of all selected files at once.
```

```
;
;  Various individuals keep trying to yank out the TYPE, LIST, and DIR
; commands, either to use the space for other options or just because
; they prefer replacement COM files. To these individuals, I (SBB) say
; keep your paws off these commands. For compatibility with the stock
; CCP, intrinsic DIR and TYPE commands are required. And many users in
; MY neighborhood find it more convenient to use the intrinsic LIST
; command than to have a LIST/PRINT program on every disk. If you want
; to call a transient program by an intrinsic, then CHANGE THE INTRINSIC
; NAME IN THE TABLE. Even setting the name to blanks is fine to get
; rid of it. The point is, don't remove features others may want, just
; because you disagree, then throw it back in our laps. For those who
; simply MUST be rid of these commands, the following symbols control
; generation of the code in a CLEAN ACCEPTABLE fashion that allows
; others to have these features:
;
;    CPRTYP -   Set to TRUE to generate code for intrinsic TYPE command.
;
;    WSTYPE -   Set to TRUE to generate an extra three lines of code
;               to correctly interpret the WordStar (tm) internal
;               end of line hyphen for display, which is the ASCII
;               NEWLINE code (1FH) and normally non-printing or
;               troublemaking -- thanks to PJH for this one. CPRTYP
;               must be TRUE, or this symbol will be ignored.
;
;    CPRLST -   Set to TRUE to generate code for intrinsic LIST command.
;               Since almost all of the LIST code is common to the
;               TYPE code, CPRTYP must be set TRUE as well, or this
;               symbol will be ignored.
;
;    CPRDIR -   Set to TRUE to generate code for intrinsic DIR command.
;               Note that unlike the various directory programs, a
;               restricted DIR command here allows displaying the names
;               of SYS file ONLY, so many RCPM operators WANT this code.
;
;  Remember, you only get a total of 2048 (0800H) bytes of space for
; ALL of the generated code, or many other areas of your system
; generation will be affected. For example, to be fully SECURE, you
; would set SECURE to TRUE, and define MAXDRIV and USRMAX, and maybe
; use the internal password by setting INPASS to TRUE (external is
; MUCH recommended for easier modification). Those options absolutely
; generate too much code unless either CPRTYP or CPRDIR or both are
; set FALSE. A system with SECURE set to FALSE is right on the edge,
; and requires a give and take on options to fit, i.e. you can have
; MAXDRIV and USRMAX with DIR and TYPE if you leave out LIST and
; querying on ERASE, and so on.
;
;*****************************************************************
;* Be careful when playing with different combinations of these equates. *
;* You might not have enough memory to some combinations.  Check this    *
;* if you have problems, if they still persist, gripe to me (PST).       *
;*****************************************************************
;
REL     EQU     FALSE   ;SET TO TRUE FOR MOVCPM INTEGRATION
;
BASE    EQU     0       ;BASE OF CP/M SYSTEM (SET FOR STANDARD CP/M)
;
        IF      REL
CPRLOC  EQU     0       ;MOVCPM IMAGE
        ELSE
```

```
;
; If REL is FALSE, the value of CPRLOC may be set in one
; of two ways.  The first way is to set MSIZE and BIOSEX
; as described above using the following three lines:

;MSIZE  EQU     56      ;SIZE OF MEM IN K-BYTES
;BIOSEX EQU     2       ;EXTRA # K-BYTES IN BIOS
;CPRLOC EQU     3400H+(MSIZE-20-BIOSEX)*1024    ;CPR ORIGIN
;
; The second way is to obtain the origin of your current
; CPR using BDSLOC or its equivalent, then merely set CPRLOC
; to that value as in the following line:
;
CPRLOC  EQU     0CB00H  ;FILL IN WITH BDOSLOC SUPPLIED VALUE
                        ;This is for the Osborne I
;
;
; Note that you should only use one method or the other.
; Do NOT define CPRLOC twice!
;
; The following gives the required offset to load the CPR into the
; CP/M SYSGEN Image through DDT (the Roffset command); Note that this
; value conforms with the standard value presented in the CP/M reference
; manuals, but it may not necessarily conform with the location of the
; CCP in YOUR CP/M system; several systems (Morrow Designs, P&T, Heath
; Org-0 to name a few) have the CCP located at a non-standard address in
; the SYSGEN Image
;
;CPRR   EQU     0E00H-CPRLOC    ;DDT LOAD OFFSET FOR APPLE SOFTCARD 56K
CPRR    EQU     0980H-CPRLOC    ;DDT LOAD OFFSET FOR D.R. STANDARD SYSGEN
;CPRR   EQU     1600H-CPRLOC    ;DDT LOAD OFFSET FOR COMPUPRO DISK-1
;CPRR   EQU     1100H-CPRLOC    ;DDT LOAD OFFSET FOR MORROW DESIGNS
        ENDIF
;
RAS     EQU     FALSE   ;SET TO TRUE IF CPR IS FOR A REMOTE-ACCESS SYSTEM
                        ;AND YOU DON'T WANT TO RUN SECURE (FOO...)
;
USRMAX  EQU     0000H           ;LOCATION OF BYTE IN MEMORY CONTAINING
                                ; NUMBER OF HIGHEST ALLOWABLE USER CODE + 1
                                ; THIS VALUE IS SET BY CPR ON COLD BOOT,
                                ; AND PRESUMABLY CONTROLLED AFTER THAT
                                ; BY A PASSWORD PROGRAM. IF USRMAX=0, THEN
                                ; MAXUSR BELOW IS USED FOR CHECKING ONLY.
                                ; 03FH IS RECOMMENDED IF USED  ***
MAXUSR  EQU     15              ;MAX ALLOWED USER NUMBER, THIS + 1 IS STUFFED
                                ; INTO USRMAX ON COLD BOOT, OR USED DIRECTLY
                                ; IF USRMAX=0
;
MAXDRIV EQU     0000H   ;LOCATION THAT HAS MAX LEGAL DRIVE #
                        ;SET IT TO ZERO TO DISABLE THIS CROCK
                        ;03DH IS RECOMMENDED IF USED ***
MAXDR   EQU     1       ;MAX DRIVE # TO SET INTO MAXDRIV ON COLD BOOT
;
SECURE  EQU     FALSE   ;SET TRUE FOR SECURE ENVIRONMENT...

EFUSR   EQU     0       ;DEFAULT USER FOR UNRESTRICTED COM FILES
;
        IF      SECURE
WHEEL   EQU     3EH     ;SET TO "RESTRCT" FOR LIMITED ACCESS
RESTRCT EQU     0       ;WHEN (WHEEL)==RESTRCT, LIMIT COMMANDS
```

```
RESUSR  EQU     15      ;CHECK HERE FOR RESTRICTED ACCESS COM FILES (LIKE PIP)
                        ; UNTIL CHANGED BY DFU OR WARM BOOT
        ENDIF   ;SECURE
;
PASS    EQU     FALSE   ;SET TRUE IF RUNNING SECURE AND NOT PASS.COM
;
DRUSER  EQU     TRUE    ;TRUE TO ALLOW USER COMMAND AND DRIVE/USER HACK
;
EPRMPT  EQU     TRUE    ;TRUE TO PROMPT BEFORE ERASING ALL FILES
;
CPRTYP  EQU     TRUE    ;TRUE TO GENERATE TYPE CODE
WSTYPE  EQU     TRUE    ;TRUE TO GENERATE WORDSTAR HYPHEN CHECK (CPRTYP
                        ; MUST BE TRUE TOO)
CPRLST  EQU     TRUE    ;TRUE TO GENERATE LIST CODE (CPRTYP MUST BE TRUE TOO)
CPRDIR  EQU     TRUE    ;TRUE TO GENERATE DIR CODE
;
;  ***  Note to Apple Softcard Users  ***
;
;  In their infinite (?) wisdom (???), Microsoft decided that the way to
; get a two-column directory display instead of four-column (narrow 40-col
; screen, remember) was to have their BIOS poke CCP every time it was
; loaded, if there was no terminal interface card in I/O slot 3.
; Naturally, that will turn into a random poke on any non-standard
; CCP, like this one.  The best way to get this CPR up on the Apple is to
; load it into CPM56.COM, at location 0E00H in the image.  The BIOS code
; that pokes the CPR can also be modified at that time.  The poke is done
; by "STA 0C8B2H", found at 24FEH in the CPM56 image.  To keep this
; feature, change the 0C8B2H address in that instruction by hand to
; the value generated for the symbol TWOPOK in the DIR routine.  If
; you have assembled out the DIR code by setting CPRDIR to FALSE, then
; disable this feature by changing the "STA" to "LDA", i.e. set the
; contents of location 24FEH from 32H to 3AH. If you wish to force
; a two-column display in all cases, set the TWOCOL switch below to a
; value of TRUE, and disable the poke.
;
TWOCOL  EQU     false           ;TRUE IF TWO COL DIR INSTEAD OF FOUR
;
; The following is presented as an option, but is not generally user-customiz-
; able.  A basic design choice had to be made in the design of ZCPR concerning
; the execution of SUBMIT files.  The original CCP had a problem in this sense
; in that it ALWAYS looked for the SUBMIT file from drive A: and the SUBMIT
; program itself (SUBMIT.COM) would place the $$$.SUB file on the currently
; logged-in drive, so when the user was logged into B: and he issued a SUBMIT
; command, the $$$.SUB was placed on B: and did not execute because the CCP
; looked for it on A: and never found it.
;
;       After much debate it was decided to have ZCPR perform the same type of
; function as CCP (look for the $$$.SUB file on A:), but the problem with
; SUBMIT.COM still exists.  Hence, RGF designed SuperSUB and RLC took his
; SuperSUB and designed SUB from it; both programs are set up to allow the
; selection at assembly time of creating the $$$.SUB on the logged-in drive
; or on drive A:.
;
;       A final definition of the Indirect Command File ($$$.SUB or SUBMIT
; File) is presented as follows:
;
;               "An Indirect Command File is one which contains
;                a series of commands exactly as they would be
;                entered from a CP/M Console.  The SUBMIT Command
;                (or SUB Command) reads this files and transforms
```

```
;                it for processing by the ZCPR (the $$$.SUB File).
;                ZCPR will then execute the commands indicated
;                EXACTLY as if they were typed at the Console."
;
;       Hence, to permit this to happen, the $$$.SUB file must always
; be present on a specific drive, and A: is the choice for said drive.
; With this facility engaged as such, Indirect Command Files like:
;
;                DIR
;                A:
;                DIR
;
; can be executed, even though the currently logged-in drive is changed
; during execution.  If the $$$.SUB file was present on the currently
; logged-in drive, the above series of commands would not work since the
; ZCPR would be looking for $$$.SUB on the logged-in drive, and switching
; logged-in drives without moving the $$$.SUB file as well would cause
; processing to abort.
;
SUBA    EQU     TRUE    ; Set to TRUE to have $$$.SUB always on A:
                        ; Set to FALSE to have $$$.SUB on the logged-in drive
;
;   The following flag enables extended processing for user-program supplied
; command lines.  This is for Command Level 3 of ZCPR.  Under the current
; ZCPR philosophy, three command levels exist:
;
;       (1) that command issued by the user from his console at the '>' prompt
;       (2) that command issued by a $$$.SUB file at the '$' prompt
;       (3) that command issued by a user program by placing the command into
;           CIBUFF and setting the character count in CBUFF
;
;   Setting CLEVEL3 to TRUE enables extended processing of the third level of
; ZCPR command.  All the user program need do is to store the command line and
; set the character count; ZCPR will initialize the pointers properly, store
; the ending zero properly, and capitalize the command line for processing.
; Once the command line is properly stored, the user executes the command line
; by reentering the ZCPR through CPRLOC [NOTE:  The C register MUST contain
; a valid User/Disk Flag (see location 4) at this time.]
;
CLEVEL3 EQU     TRUE            ;ENABLE COMMAND LEVEL 3 PROCESSING
;
;
;*** TERMINAL AND 'TYPE' CUSTOMIZATION EQUATES
;
NLINES  EQU     24              ;NUMBER OF LINES ON CRT SCREEN
WIDE    EQU     true            ;TRUE IF WIDE DIR DISPLAY
FENCE   EQU     '|'             ;SEP CHAR BETWEEN DIR FILES
;
PGDFLT  EQU     TRUE            ;SET TO FALSE TO DISABLE PAGING BY DEFAULT
PGDFLG  EQU     'P'             ;FOR TYPE COMMAND: PAGE OR NOT (DEP ON PGDFLT)
                                ;   THIS FLAG REVERSES THE DEFAULT EFFECT
;
SYSFLG  EQU     'A'             ;FOR DIR COMMAND: LIST $SYS AND $DIR
;
SOFLG   EQU     'S'             ;FOR DIR COMMAND: LIST $SYS FILES ONLY
;
SUPRES  EQU     false           ;SUPRESSES USER # REPORT FOR USER 0
;
SPRMPT  EQU     '$'             ;CPR PROMPT INDICATING SUBMIT COMMAND
CPRMPT  EQU     '>'             ;CPR PROMPT INDICATING USER COMMAND
```

```
;
NUMBASE EQU     'H'             ;CHARACTER USED TO SWITCH FROM DEFAULT
                                ; NUMBER BASE
;
STFLG EQU       'S'             ;OPTION CHAR FOR SAVE COMMAND TO SAVE SECTORS
;
; END OF CUSTOMIZATION SECTION
;
CR      EQU     0DH
LF      EQU     0AH
TAB     EQU     09H
FFEED   EQU     0CH
BEL     EQU     07H
;
WBOOT   EQU     BASE+0000H              ;CP/M WARM BOOT ADDRESS
UDFLAG  EQU     BASE+0004H              ;USER NUM IN HIGH NYBBLE, DISK IN LOW
BDOS    EQU     BASE+0005H              ;BDOS FUNCTION CALL ENTRY PT
TFCB    EQU     BASE+005CH              ;DEFAULT FCB BUFFER
TBUFF   EQU     BASE+0080H              ;DEFAULT DISK I/O BUFFER
TPA     EQU     BASE+0100H              ;BASE OF TPA
;
;
; MACROS TO PROVIDE Z80 EXTENSIONS
;   MACROS INCLUDE:
;
$-MACRO                 ;FIRST TURN OFF THE EXPANSIONS
;
;       JR      - JUMP RELATIVE
;       JRC     - JUMP RELATIVE IF CARRY
;       JRNC    - JUMP RELATIVE IF NO CARRY
;       JRZ     - JUMP RELATIVE IF ZERO
;       JRNZ    - JUMP RELATIVE IF NO ZERO
;       DJNZ    - DECREMENT B AND JUMP RELATIVE IF NO ZERO
;       LDIR    - MOV @HL TO @DE FOR COUNT IN BC
;       LXXD    - LOAD DOUBLE REG DIRECT
;       SXXD    - STORE DOUBLE REG DIRECT
;
;
;
;       @GENDD MACRO USED FOR CHECKING AND GENERATING
;       8-BIT JUMP RELATIVE DISPLACEMENTS
;
@GENDD  MACRO   ?DD     ;;USED FOR CHECKING RANGE OF 8-BIT DISPLACEMENTS
        IF (?DD GT 7FH) AND (?DD LT 0FF80H)
        DB      100H    ;Displacement Range Error on Jump Relative
        ELSE
        DB      ?DD
        ENDIF
        ENDM
;
;
; Z80 MACRO EXTENSIONS
;
JR      MACRO   ?N      ;;JUMP RELATIVE
        DB      18H
        @GENDD  ?N-$-1
        ENDM
;
JRC     MACRO   ?N      ;;JUMP RELATIVE ON CARRY
        DB      38H
```

```
        @GENDD  ?N-$-1
        ENDM
;
JRNC    MACRO   ?N      ;;JUMP RELATIVE ON NO CARRY
        DB      30H
        @GENDD  ?N-$-1
        ENDM
;
JRZ     MACRO   ?N      ;;JUMP RELATIVE ON ZERO
        DB      28H
        @GENDD  ?N-$-1
        ENDM
;
JRNZ    MACRO   ?N      ;;JUMP RELATIVE ON NO ZERO
        DB      20H
        @GENDD  ?N-$-1
        ENDM
;
DJNZ    MACRO   ?N      ;;DECREMENT B AND JUMP RELATIVE ON NO ZERO
        DB      10H
        @GENDD  ?N-$-1
        ENDM
;
LDIR    MACRO           ;;LDIR
        DB      0EDH,0B0H
        ENDM
;
LDED    MACRO   ?N      ;;LOAD DE DIRECT
        DB      0EDH,05BH
        DW      ?N
        ENDM
;
LBCD    MACRO   ?N      ;;LOAD BC DIRECT
        DB      0EDH,4BH
        DW      ?N
        ENDM
;
SDED    MACRO   ?N      ;;STORE DE DIRECT
        DB      0EDH,53H
        DW      ?N
        ENDM
;
SBCD    MACRO   ?N      ;;STORE BC DIRECT
        DB      0EDH,43H
        DW      ?N
        ENDM
;
; END OF Z80 MACRO EXTENSIONS
;
;
;**** Section 0 ****
;
        ORG     CPRLOC
;
;  ENTRY POINTS INTO ZCPR
;
;     If the ZCPR is entered at location CPRLOC (at the JMP to CPR), then
; the default command in CIBUFF will be processed.  If the ZCPR is entered
; at location CPRLOC+3 (at the JMP to CPR1), then the default command in
; CIBUFF will NOT be processed.
```

```
;
;     NOTE:  Entry into ZCPR in this way is permitted under this version,
; but in order for this to work, CIBUFF and CBUFF MUST be initialized properly
; AND the C register MUST contain a valid User/Disk Flag (see Location 4: the
; most significant nybble contains the User Number and the least significant
; nybble contains the Disk Number).
;
;     Some user programs (such as SYNONYM3) attempt to use the default
; command facility.  Under the original CCP, it was necessary to initialize
; the pointer after the reserved space for the command buffer to point to
; the first byte of the command buffer.  Under current versions, this is
; no longer the case.  The CIBPTR (Command Input Buffer PoinTeR) is located
; to be compatible with such programs (provided they determine the buffer
; length from the byte at MBUFF [CPRLOC + 6]), but under ZCPR this is
; no longer necessary, since this buffer pointer is automatically
; initialized in all cases.
;
ENTRY:
        JMP     FSTJMP  ; To properly set the Osborne CCP, the first
;                       ; two bytes must be C3, 5C (see ONZCPR21.DOC)
        JMP     CPR     ; Process potential default command, and set
;                       ; USRMAX to MAXUSR default
;       JMP     CPR1    ; Do NOT process potential default command
;
;**** Section 1 ****
; BUFFERS ET AL
;
; INPUT COMMAND LINE AND DEFAULT COMMAND
;
;   The command line to be executed is stored here.  This command line
; is generated in one of three ways:
;
;       (1) by the user entering it through the BDOS READLN function at
;           the du> prompt [user input from keyboard]
;       (2) by the SUBMIT File Facility placing it there from a $$$.SUB
;           file
;       (3) by an external program or user placing the required command
;           into this buffer
;
;   In all cases, the command line is placed into the buffer starting at
; CIBUFF.  This command line is terminated by the last character (NOT Carriage
; Return), and a character count of all characters in the command line
; up to and including the last character is placed into location CBUFF
; (immediately before the command line at CIBUFF).  The placed command line
; is then parsed, interpreted, and the indicated command is executed.
; If CLEVEL3 is permitted, a terminating zero is placed after the command
; (otherwise the user program has to place this zero) and the CIBPTR is
; properly initialized (otherwise the user program has to init this ptr).
; If the command is placed by a user program, entering at CPRLOC is enough
; to have the command processed.  Again, under the current ZCPR, it is not
; necessary to store the pointer to CIBUFF in CIBPTR; ZCPR will do this for
; the calling program if CLEVEL3 is made TRUE.
;
;   WARNING:  The command line must NOT exceed BUFLEN characters in length.
; For user programs which load this command, the value of BUFLEN can be
; obtained by examining the byte at MBUFF (CPRLOC + 6).
;
BUFLEN  EQU     80                      ;MAXIMUM BUFFER LENGTH
MBUFF:  DB      BUFLEN                  ;MAXIMUM BUFFER LENGTH
CBUFF:  DB      0                       ;NUMBER OF VALID CHARS IN COMMAND LINE
```

```
CIBUFF: DB      '               '       ;DEFAULT (COLD BOOT) COMMAND
;
;  The copyright notice from Digital Research is genned into the
; stock CCP at this location. It should be maintained in ZCPR,
; since Digital Research grants permission for ZCPR to exist.
;
        DB      '  Copyright (C) 1979, Digital Research  '
CIBUF:
        DB      0                       ;COMMAND STRING TERMINATOR
        DS      BUFLEN-($-CIBUFF)+1     ;TOTAL IS 'BUFLEN' BYTES
;
CIBPTR:
        DW      CIBUFF          ;POINTER TO COMMAND INPUT BUFFER
CIPTR:
        DW      CIBUF           ;POINTER TO CURR COMMAND FOR
                                ; ERROR REPORTING
;
        DS      26              ;STACK AREA
STACK   EQU     $               ;TOP OF STACK
;
; FILE TYPE FOR COMMAND
;
COMMSG:
        DB      'COM'
;
; SUBMIT FILE CONTROL BLOCK
;
SUBFCB:
        IF      SUBA            ;IF $$$.SUB ON A:
        DB      1               ;DISK NAME SET TO DEFAULT TO DRIVE A:
        ENDIF
;
        IF      NOT SUBA        ;IF $$$.SUB ON CURRENT DRIVE
        DB      0               ;DISK NAME SET TO DEFAULT TO CURRENT DRIVE
        ENDIF
;
        DB      '$$$'           ;FILE NAME
        DB      '   '
        DB      'SUB'           ;FILE TYPE
        DB      0               ;EXTENT NUMBER
        DB      0               ;S1
SUBFS2:
        DS      1               ;S2
SUBFRC:
        DS      1               ;RECORD COUNT
        DS      16              ;DISK GROUP MAP
SUBFCR:
        DS      1               ;CURRENT RECORD NUMBER
  COMMAND FILE CONTROL BLOCK

CBDN:
        DS      1               ;DISK NAME
CBFN:
        DS      8               ;FILE NAME
CBFT:
        DS      3               ;FILE TYPE
        DS      1               ;EXTENT NUMBER
        DS      2               ;S1 AND S2
        DS      1               ;RECORD COUNT
CBDM:
```

```
        DS      16              ;DISK GROUP MAP
FCBCR:
        DS      1               ;CURRENT RECORD NUMBER
;
;OTHER BUFFERS
;
PAGCNT:
        DB      NLINES-2        ;LINES LEFT ON PAGE
CHRCNT:
        DB      0               ;CHAR COUNT FOR TYPE
QMCNT:
        DB      0               ;QUESTION MARK COUNT FOR FCB TOKEN SCANNER
;
;
;**** Section 2 ****
; CPR starting points.  Note that some CP/M implementations
; require the cold start address to be in the starting page
; of the CCP, for dynamic CCP loading.  CMDTBL was moved for
; this reason.
;
; Set USRMAX and/or MAXDRIV to default values on cold boot
; if required. Note that some BIOS implementations will end
; up here instead of at the warm boot, defeating passwording
; of these options. I reccomend that such a BIOS be fixed.
;
        IF      USRMAX OR MAXDRIV
CPR:
        IF      USRMAX
        MVI     A,MAXUSR+1      ;SET USRMAX ON COLD BOOT
        STA     USRMAX
        ENDIF                   ;USRMAX
;
        IF      MAXDRIV
        MVI     A,MAXDR         ;SET MAXDRIV ON COLD BOOT
        STA     MAXDRIV
        ENDIF                   ;MAXDRIV
;
        JR      CPR2            ; THEN PROCEED
        ENDIF                   ;USRMAX OR MAXDRIV
;
; Start CPR and don't process default command stored
;
CPR1:   XRA     A               ;SET NO DEFAULT COMMAND
        STA     CBUFF
;
; Start CPR and possibly process default command
;
; Note on modification by RGF: BDOS returns OFFh in
; accumulator whenever it logs in a directory, if any
; file name contains a '$' in it.  This is now used as
; a clue to determine whether or not to do a search
; for submit file, in order to eliminate wasteful searches.
;
        IF      USRMAX OR MAXDRIV
CPR2:
        ELSE
CPR:
        ENDIF                   ;USRMAX OR MAXDRIV
;
        LXI     SP,STACK        ;RESET STACK
```

```
        PUSH    B
        MOV     A,C             ;C=USER/DISK NUMBER (SEE LOC 4)
        RAR                     ;EXTRACT USER NUMBER
        RAR
        RAR
        RAR
        ANI     0FH
        MOV     E,A             ;SET USER NUMBER
        CALL    SETUSR
        CALL    RESET           ;RESET DISK SYSTEM
        STA     RNGSUB          ;SAVE SUBMIT CLUE FROM DRIVE A:
        POP     B
        MOV     A,C             ;C=USER/DISK NUMBER (SEE LOC 4)
        ANI     0FH             ;EXTRACT DEFAULT DISK DRIVE
        STA     TDRIVE          ;SET IT
        JRZ     NOLOG           ;SKIP IF 0...ALREADY LOGGED
        CALL    LOGIN           ;LOG IN DEFAULT DISK
;
        IF      NOT SUBA        ;IF $$$.SUB IS ON CURRENT DRIVE
        STA     RNGSUB          ;BDOS '$' CLUE
        ENDIF
;
NOLOG:  LXI     D,SUBFCB        ;CHECK FOR $$$.SUB ON CURRENT DISK
RNGSUB  EQU     $+1             ;POINTER FOR IN-THE-CODE MODIFICATION
        MVI     A,0             ;2ND BYTE (IMMEDIATE ARG) IS THE RNGSUB FLAG
        ORA     A               ;SET FLAGS ON CLUE
        CMA                     ;PREPARE FOR COMING 'CMA'
        CNZ     SEAR1
        CMA                     ;OFFH IS RETURNED IF NO $$$.SUB, SO COMPLEMENT
        STA     RNGSUB          ;SET FLAG (0=NO $$$.SUB)
        LDA     CBUFF           ;EXECUTE DEFAULT COMMAND?
        ORA     A               ;0=NO
        JRNZ    RS1
;
; PROMPT USER AND INPUT COMMAND LINE FROM HIM
;
RESTRT: LXI     SP,STACK        ;RESET STACK
;
; PRINT PROMPT (DU>)
;
        CALL    CRLF            ;PRINT PROMPT
        CALL    GETDRV          ;CURRENT DRIVE IS PART OF PROMPT
        ADI     'a'             ;CONVERT TO ASCII A-P
        CALL    CONOUT
        CALL    GETUSR          ;GET USER NUMBER
;
        IF      SUPRES          ;IF SUPPRESSING USR # REPORT FOR USR 0
        ORA     A
        JRZ     RS000
        ENDIF
;
        CPI     10              ;USER < 10?
        JRC     RS00
        SUI     10              ;SUBTRACT 10 FROM IT
        PUSH    PSW             ;SAVE IT
        MVI     A,'1'           ;OUTPUT 10'S DIGIT
        CALL    CONOUT
        POP     PSW
RS00:   ADI     '0'             ;OUTPUT 1'S DIGIT (CONVERT TO ASCII)
        CALL    CONOUT
```

```
;
; READ INPUT LINE FROM USER OR $$$.SUB
;
RSOOO:  CALL    REDBUF          ;INPUT COMMAND LINE FROM USER (OR $$$.SUB)
;
; PROCESS INPUT LINE
;
RS1:
;
        IF      CLEVEL3         ;IF THIRD COMMAND LEVEL IS PERMITTED
        CALL    CNVBUF          ;CAPITALIZE COMMAND LINE, PLACE ENDING 0,
                                ; AND SET CIBPTR VALUE
        ENDIF
;
        CALL    DEFDMA          ;SET TBUFF TO DMA ADDRESS
        CALL    GETDRV          ;GET DEFAULT DRIVE NUMBER
        STA     TDRIVE          ;SET IT
        CALL    SCANER          ;PARSE COMMAND NAME FROM COMMAND LINE
        CNZ     ERROR           ;ERROR IF COMMAND NAME CONTAINS A '?'
        LXI     D,RSTCPR        ;PUT RETURN ADDRESS OF COMMAND
        PUSH    D               ;ON THE STACK
        LDA     TEMPDR          ;IS COMMAND OF FORM 'D:COMMAND'?
        ORA     A               ;NZ=YES
        JNZ     COM             ; IMMEDIATELY
        CALL    CMDSER          ;SCAN FOR CPR-RESIDENT COMMAND
        JNZ     COM             ;NOT CPR-RESIDENT
        MOV     A,M             ;FOUND IT:  GET LOW-ORDER PART
        INX     H               ;GET HIGH-ORDER PART
        MOV     H,M             ;STORE HIGH
        MOV     L,A             ;STORE LOW
        PCHL                    ;EXECUTE CPR ROUTINE
;
; ENTRY POINT FOR RESTARTING CPR AND LOGGING IN DEFAULT DRIVE
;
RSTCPR: CALL    DLOGIN          ;LOG IN DEFAULT DRIVE
;
; ENTRY POINT FOR RESTARTING CPR WITHOUT LOGGING IN DEFAULT DRIVE
;
RCPRNL: CALL    SCANER          ;EXTRACT NEXT TOKEN FROM COMMAND LINE
        LDA     FCBFN           ;GET FIRST CHAR OF TOKEN
        SUI     ' '             ;ANY CHAR?
        LXI     H,TEMPDR
        ORA     M
        JNZ     ERROR
        JR      RESTRT
;
; No File Error Message
;
PRNNF:  CALL    PRINTC          ;NO FILE MESSAGE
        DB      'No Fil','e'+80H
        RET
;
; CPR BUILT-IN COMMAND TABLE
;
NCHARS  EQU     4               ;NUMBER OF CHARS/COMMAND
;
; CPR COMMAND NAME TABLE
;   EACH TABLE ENTRY IS COMPOSED OF THE 4-BYTE COMMAND AND 2-BYTE ADDRESS
;
CMDTBL:
```

```
;
        IF      INPASS AND SECURE
        DB      'PASS'          ;ENABLE WHEEL (SYSOP) MODE
        DW      PASS
        ENDIF                   ;INPASS AND SECURE
;
        IF      DRUSER
        DB      'USER'          ;CHANGE USER AREAS
        DW      USER
        ENDIF                   ;DRUSER
;
        IF      CPRTYP
        DB      'TYPE'          ;TYPE A FILE TO CON:
        DW      TYPE
        ENDIF                   ;CPRTYP
;
        IF      CPRDIR
        DB      'DIR '          ;PULL A DIRECTORY OF DISK FILES
        DW      DIR
        ENDIF                   ;CPRDIR

NRCMDS  EQU     ($-CMDTBL)/(NCHARS+2)   ;PUT ANY COMMANDS THAT ARE OK TO
                                        ;RUN WHEN NOT UNDER WHEEL MODE
                                        ;IN FRONT OF THIS LABEL
        IF      CPRLST AND CPRTYP
        DB      'LIST'          ;LIST FILE TO PRINTER
        DW      LIST
        ENDIF                   ;CPRLST AND CPRTYP
;
        IF      INPASS AND SECURE
        DB      'NORM'          ;DISABLE WHEEL MODE
        DW      NORM
        ENDIF                   ;INPASS AND SECURE
;
        IF      NOT RAS         ;FOR NON-RAS
        DB      'GO  '          ;JUMP TO 100H
        DW      GO
        DB      'ERA '          ;ERASE FILE
        DW      ERA
        DB      'SAVE'          ;SAVE MEMORY IMAGE TO DISK
        DW      SAVE
        DB      'REN '          ;RENAME FILE
        DW      REN
        DB      'DFU '          ;SET DEFAULT USER
        DW      DFU
        DB      'GET '          ;LOAD FILE INTO MEMORY
        DW      GET
        DB      'JUMP'          ;JUMP TO LOCATION IN MEMORY
        DW      JUMP
        ENDIF                   ;RAS
;
NCMNDS  EQU     ($-CMDTBL)/(NCHARS+2)
;
;**** Section 3 ****
; I/O UTILITIES

; OUTPUT CHAR IN REG A TO CONSOLE AND DON'T CHANGE BC
;
;
; OUTPUT <CRLF>
```

```
;
CRLF:   MVI     A,CR
        CALL    CONOUT
        MVI     A,LF            ;FALL THRU TO CONOUT

CONOUT: PUSH    B
        MVI     C,02H
OUTPUT: ANI     7FH             ;PREVENT INADVERTANT GRAPHIC OUTPUT
                                ;TO EPSON-TYPE PRINTERS
        MOV     E,A
        PUSH    H
        CALL    BDOS
        POP     H
        POP     B
        RET
;
CONIN:  MVI     C,01H           ;GET CHAR FROM CON: WITH ECHO
        CALL    BDOSB
;
; CONVERT CHAR IN A TO UPPER CASE
;
UCASE:  CPI     61H             ;LOWER-CASE A
        RC
        CPI     7BH             ;GREATER THAN LOWER-CASE Z?
        RNC
        ANI     5FH             ;CAPITALIZE
        RET
;
        IF      CPRTYP
LCOUT:
        ENDIF                   ;CPRTYP
;
        IF      CPRTYP AND CPRLST
        PUSH    PSW             ;OUTPUT CHAR TO CON: OR LST: DEP ON PRFLG
PRFLG   EQU     $+1             ;POINTER FOR IN-THE-CODE MODIFICATION
        MVI     A,0             ;2ND BYTE (IMMEDIATE ARG) IS THE PRINT FLAG
        ORA     A               ;0=TYPE
        JRZ     LC1
        POP     PSW             ;GET CHAR
;
; OUTPUT CHAR IN REG A TO LIST DEVICE
;
LSTOUT: PUSH    B
        MVI     C,05H
        JR      OUTPUT
LC1:    POP     PSW             ;GET CHAR
        ENDIF                   ;CPRTYP AND CPRLST
;
        IF      CPRTYP
        PUSH    PSW
        CALL    CONOUT          ;OUTPUT TO CON:
        POP     PSW
        CPI     LF              ;CHECK FOR PAGING
        RNZ                     ;DONE IF NOT EOL YET
;
; COUNT DOWN LINES AND PAUSE FOR INPUT (DIRECT) IF COUNT EXPIRES
;
        PUSH    H
        LXI     H,PAGCNT        ;COUNT DOWN
        DCR     M
```

```
        JRNZ    PGBAK           ;JUMP IF NOT END OF PAGE
        MVI     M,NLINES-2      ;REFILL COUNTER
;
PGFLG   EQU     $+1             ;POINTER TO IN-THE-CODE BUFFER PGFLG
        MVI     A,0             ;0 MAY BE CHANGED BY PGFLG EQUATE
        CPI     PGDFLG          ;PAGE DEFAULT OVERRIDE OPTION WANTED?
;
        IF      PGDFLT          ;IF PAGING IS DEFAULT
        JRZ     PGBAK           ;PGDFLG MEANS NO PAGING, PLEASE
        ELSE                    ;IF PAGING NOT DEFAULT
        JRNZ    PGBAK           ;  PGDFLG MEANS PLEASE PAGINATE
        ENDIF
;
WTLOOP: CALL    BREAK           ;GET CHAR BUT DON'T ECHO TO SCREEN
        JRZ     WTLOOP          ;NOTHING THERE YET.... SO LOOP
        CPI     'C'-'@'         ;^C
        JZ      RSTCPR          ;RESTART CPR

PGBAK:  POP     H               ;RESTORE HL
        RET
        ENDIF                   ;CPRTYP
;
READF:  LXI     D,FCBDN         ;FALL THRU TO READ
READ:   MVI     C,14H           ;FALL THRU TO BDOSB
;
; CALL BDOS AND SAVE BC
;
BDOSB:  PUSH    B
        CALL    BDOS
        POP     B
        ORA     A
        RET
;
; Print string ending with zero byte or character high bit set
; pointed to by ret address,  start with <CR><LF>
;
PRINTC: PUSH    PSW             ;SAVE FLAGS
        CALL    CRLF            ;NEW LINE
        POP     PSW
;
PRINT:  XTHL                    ;GET PTR TO STRING
        PUSH    PSW             ;SAVE FLAGS
        CALL    PRIN1           ;PRINT STRING
        POP     PSW             ;GET FLAGS
        XTHL                    ;RESTORE HL AND RET ADR
        RET
;
; Print string ending with zero byte or character high bit set
; pointed to by HL
;
PRIN1:  MOV     A,M             ;GET NEXT BYTE
        CALL    CONOUT          ;PRINT CHAR
        MOV     A,M             ;GET NEXT BYTE AGAIN FOR TEST
        INX     H               ;PT TO NEXT BYTE
        ORA     A               ;SET FLAGS
        RZ                      ;DONE IF ZERO
        RM                      ;DONE IF MSB SET
        JR      PRIN1
;
; BDOS FUNCTION ROUTINES
```

```
;
;
; RETURN NUMBER OF CURRENT DISK IN A
;
GTDRV:  MVI     C,19H
        JR      BDOSJP
;
; SET 80H AS DMA ADDRESS
;
DEFDMA: LXI     D,TBUFF         ;80H=TBUFF
DMASET: MVI     C,1AH
        JR      BDOSJP
;
RESET:  MVI     C,0DH
BDOSJP: JMP     BDOS
;
LOGIN:  MOV     E,A             ;MOVE DESIRED # TO BDOS REG
;
        IF      MAXDRIV
        LDA     MAXDRIV         ;CHECK FOR LEGAL DRIVE #
        CMP     E
        JC      ERROR           ;DON'T DO IT IF TOO HIGH
        ENDIF                   ;MAXDRIV
;
        MVI     C,0EH
        JR      BDOSJP          ;SAVE SOME CODE SPACE
;
OPENF:  XRA     A
        STA     FCBCR
        LXI     D,FCBDN         ;FALL THRU TO OPEN
;
OPEN:   MVI     C,0FH           ;FALL THRU TO GRBDOS
;
GRBDOS: CALL    BDOS
        INR     A               ;SET ZERO FLAG FOR ERROR RETURN
        RET
;
CLOSE:  MVI     C,10H
        JR      GRBDOS
;
SEARF:  LXI     D,FCBDN         ;SPECIFY FCB
SEAR1:  MVI     C,11H
        JR      GRBDOS
;
SEARN:  MVI     C,12H
        JR      GRBDOS
;
; CHECK FOR SUBMIT FILE IN EXECUTION AND ABORT IT IF SO
;
SUBKIL: LXI     H,RNGSUB        ;CHECK FOR SUBMIT FILE IN EXECUTION
        MOV     A,M
        ORA     A               ;0=NO
        RZ
        MVI     M,0             ;ABORT SUBMIT FILE
        LXI     D,SUBFCB        ;DELETE $$$.SUB

DELETE: MVI     C,13H
        JR      BDOSJP          ;SAVE MORE SPACE
;
; RESET USER NUMBER IF CHANGED
```

```
;
RESETUSR:
TMPUSR  EQU     $+1             ;POINTER FOR IN-THE-CODE MODIFICATION
        MVI     A,0             ;2ND BYTE (IMMEDIATE ARG) IS TMPUSR
        MOV     E,A             ;PLACE IN E
        JR      SETUSR          ;THEN GO SET USER

GETUSR: MVI     E,0FFH          ;GET CURRENT USER NUMBER
SETUSR: MVI     C,20H           ;SET USER NUMBER TO VALUE IN E (GET IF E=FFH)
        JR      BDOSJP          ;MORE SPACE SAVING
;
; END OF BDOS FUNCTIONS
;
;
;**** Section 4 ****
; CPR utilities
;
; Set user/disk flag to current user and default disk
;
SETUD:  CALL    GETUSR          ;GET NUMBER OF CURRENT USER
        ADD     A               ;PLACE IT IN HIGH NYBBLE
        ADD     A
        ADD     A
        ADD     A
        LXI     H,TDRIVE        ;MASK IN DEFAULT DRIVE NUMBER (LOW NYBBLE)
        ORA     M               ;MASK IN
        STA     UDFLAG          ;SET USER/DISK NUMBER
        RET
;
; Set user/disk flag to user 0 and default disk
;
SETUOD:
TDRIVE  EQU     $+1             ;POINTER FOR IN-THE-CODE MODIFICATION
        MVI     A,0             ;2ND BYTE (IMMEDIATE ARG) IS TDRIVE
        STA     UDFLAG          ;SET USER/DISK NUMBER
        RET
;
; Input next command to CPR
;       This routine determines if a SUBMIT file is being processed
; and extracts the command line from it if so or from the user's console
;
REDBUF: LDA     RNGSUB          ;SUBMIT FILE CURRENTLY IN EXECUTION?
        ORA     A               ;0=NO
        JRZ     RB1             ;GET LINE FROM CONSOLE IF NOT
        LXI     D,SUBFCB        ;OPEN $$$.SUB
        PUSH    D               ;SAVE DE
        CALL    OPEN
        POP     D               ;RESTORE DE
        JRZ     RB1             ;ERASE $$$.SUB IF END OF FILE AND GET CMND
        LDA     SUBFRC          ;GET VALUE OF LAST RECORD IN FILE
        DCR     A               ;PT TO NEXT TO LAST RECORD
        STA     SUBFCR          ;SAVE NEW VALUE OF LAST RECORD IN $$$.SUB
        CALL    READ            ;DE=SUBFCB
        JRNZ    RB1             ;ABORT $$$.SUB IF ERROR IN READING LAST REC
        LXI     D,CBUFF         ;COPY LAST RECORD (NEXT SUBMIT CMND) TO CBUFF
        LXI     H,TBUFF         ;  FROM TBUFF
        LXI     B,BUFLEN        ;NUMBER OF BYTES
        LDIR
        LXI     H,SUBFS2        ;PT TO S2 OF $$$.SUB FCB
        MVI     M,0             ;SET S2 TO ZERO
```

```
        INX     H               ;PT TO RECORD COUNT
        DCR     M               ;DECREMENT RECORD COUNT OF $$$.SUB
        LXI     D,SUBFCB        ;CLOSE $$$.SUB
        CALL    CLOSE
        JRZ     RB1             ;ABORT $$$.SUB IF ERROR
        MVI     A,SPRMPT        ;PRINT SUBMIT PROMPT
        CALL    CONOUT
        LXI     H,CIBUFF        ;PRINT COMMAND LINE FROM $$$.SUB
        CALL    PRIN1
        CALL    BREAK           ;CHECK FOR ABORT (ANY CHAR)
;
        IF      CLEVEL3         ;IF THIRD COMMAND LEVEL IS PERMITTED
        RZ                      ;IF <NULL> (NO ABORT), RETURN TO CALLER AND RUN
        ENDIF
;
        IF      NOT CLEVEL3     ;IF THIRD COMMAND LEVEL IS NOT PERMITTED
        JRZ     CNVBUF          ;IF <NULL> (NO ABORT), CAPITALIZE COMMAND
        ENDIF
;
        CALL    SUBKIL          ;KILL $$$.SUB IF ABORT
        JMP     RESTRT          ;RESTART CPR
;
; INPUT COMMAND LINE FROM USER CONSOLE
;
RB1:    CALL    SUBKIL          ;ERASE $$$.SUB IF PRESENT
        CALL    SETUD           ;SET USER AND DISK
        MVI     A,CPRMPT        ;PRINT PROMPT
        CALL    CONOUT
        MVI     C,0AH           ;READ COMMAND LINE FROM USER
        LXI     D,MBUFF
        CALL    BDOS
;
        IF      CLEVEL3         ;IF THIRD COMMAND LEVEL IS PERMITTED
        JMP     SETUOD          ;SET CURRENT DISK NUMBER IN LOWER PARAMS
        ENDIF
;
        IF      NOT CLEVEL3     ;IF THIRD COMMAND LEVEL IS NOT PERMITTED
        CALL    SETUOD          ;SET CURRENT DISK NUMBER IF LOWER PARAMS
                                ; AND FALL THRU TO CNVBUF
        ENDIF
;
; CAPITALIZE STRING (ENDING IN 0) IN CBUFF AND SET PTR FOR PARSING
;
CNVBUF: LXI     H,CBUFF         ;PT TO USER'S COMMAND
        MOV     B,M             ;CHAR COUNT IN B
        INR     B               ;ADD 1 IN CASE OF ZERO
CB1:    INX     H               ;PT TO 1ST VALID CHAR
        MOV     A,M             ;CAPITALIZE COMMAND CHAR
        CALL    UCASE
        MOV     M,A
        DJNZ    CB1             ;CONTINUE TO END OF COMMAND LINE
CB2:    MVI     M,0             ;STORE ENDING <NULL>
        LXI     H,CIBUFF        ;SET COMMAND LINE PTR TO 1ST CHAR
        SHLD    CIBPTR
        RET

; CHECK FOR ANY CHAR FROM USER CONSOLE;RET W/ZERO SET IF NONE
;
BREAK:  PUSH    D               ;SAVE DE
        MVI     C,6             ;DIRECT CONSOLE I/O
```

```
        MVI     E,0FFh          ;INPUT MODE
        CALL    BDOSB           ;GET CHARACTER (IF ANY)
        POP     D               ;RESTORE DE
        JNZ     UCASE           ;WE HAVE SOMETHING, CASEIFY AND RE
        RET
;
; GET THE REQUESTED USER NUMBER FROM THE COMMAND LINE AND VALIDATE
;
USRNUM: CALL    NUMBER
;
        IF      USRMAX
        LXI     H,USRMAX        ;PT TO MAXUSR + 1
        CMP     M               ;NEW VALUE ALLOWED?
        ELSE
        CPI     MAXUSR+1        ;NEW VALUE ALLOWED?
        ENDIF                   ;USRMAX
;
        RC                      ;RETURN TO CALLER IF SO,
                                ; ELSE FLAG AS ERROR
;
; INVALID COMMAND -- PRINT IT
;
ERROR:  CALL    CRLF            ;NEW LINE
        LHLD    CIPTR           ;PT TO BEGINNING OF COMMAND LINE
ERR2:   MOV     A,M             ;GET CHAR
        CPI     ' '+1           ;SIMPLE '?' IF <SP> OR LESS
        JRC     ERR1
        PUSH    H               ;SAVE PTR TO ERROR COMMAND CHAR
        CALL    CONOUT          ;PRINT COMMAND CHAR
        POP     H               ;GET PTR
        INX     H               ;PT TO NEXT
        JR      ERR2            ;CONTINUE
ERR1:   CALL    PRINT           ;PRINT '?'
        DB      '?'+80H
        CALL    SUBKIL          ;TERMINATE ACTIVE $$$.SUB IF ANY
        JMP     RESTRT          ;RESTART CPR
;
; CHECK TO SEE IF DE PTS TO DELIMITER; IF SO, RET W/ZERO FLAG SET
;
SDELM:  LDAX    D
        ORA     A               ;0=DELIMITER
        RZ
        CPI     ' '             ;ERROR IF < <SP>
        JRC     ERROR
        RZ                      ;<SP>=DELIMITER
        CPI     '='             ;'='=DELIMITER
        RZ
        CPI     5FH             ;UNDERSCORE=DELIMITER
        RZ
        CPI     '.'             ;'.'=DELIMITER
        RZ
        CPI     ':'             ;':'=DELIMITER
        RZ
        CPI     ';'             ;';'=DELIMITER
        RZ
        CPI     '<'             ;'<'=DELIMITER
        RZ
        CPI     '>'             ;'>'=DELIMITER
        RET
;
```

```
; ADVANCE INPUT PTR TO FIRST NON-BLANK AND FALL THROUGH TO SBLANK
;
ADVAN:  LDED    CIBPTR
;
;  SKIP STRING PTED TO BY DE (STRING ENDS IN 0) UNTIL END OF STRING
;    OR NON-BLANK ENCOUNTERED (BEGINNING OF TOKEN)
;
SBLANK: LDAX    D
        ORA     A
        RZ
        CPI     ' '
        RNZ
        INX     D
        JR      SBLANK
;
; ADD A TO HL (HL=HL+A)
;
ADDAH:  ADD     L
        MOV     L,A
        RNC
        INR     H
        RET
;
;
        JMP     NADA            ;NULL COMMAND TO MAINTAIN PROGRAM FLOW
FSTJMP: ORG     0CE5CH
        JMP     CPR
NADA:   NOP                     ;AROUND "FSTJMP"
;
; EXTRACT DECIMAL NUMBER FROM COMMAND LINE
;   RETURN WITH VALUE IN REG A  ;ALL REGISTERS MAY BE AFFECTED
;
NUMBER: CALL    SCANER          ;PARSE NUMBER AND PLACE IN FCBFN
        LXI     H,FCBFN+10      ;PT TO END OF TOKEN FOR CONVERSION
        MVI     B,11            ;11 CHARS MAX
;
; CHECK FOR SUFFIX FOR HEXADECIMAL NUMBER
;
NUMS:   MOV     A,M             ;GET CHARS FROM END, SEARCHING FOR SUFFIX
        DCX     H               ;BACK UP
        CPI     ' '             ;SPACE?
        JRNZ    NUMS1           ;CHECK FOR SUFFIX
        DJNZ    NUMS            ;COUNT DOWN
        JR      NUM0            ;BY DEFAULT, PROCESS
NUMS1:  CPI     NUMBASE         ;CHECK AGAINST BASE SWITCH FLAG
        JRZ     HNUM0
;
; PROCESS DECIMAL NUMBER
;
NUM0:   LXI     H,FCBFN         ;PT TO BEGINNING OF TOKEN
        LXI     B,1100H         ;C=ACCUMULATED VALUE, B=CHAR COUNT
                                ; (C=0, B=11)
NUM1:   MOV     A,M             ;GET CHAR
        CPI     ' '             ;DONE IF <SP>
        JRZ     NUM2
        INX     H               ;PT TO NEXT CHAR
        SUI     '0'             ;CONVERT TO BINARY (ASCII 0-9 TO BINARY)
        CPI     10              ;ERROR IF >= 10
        JRNC    NUMERR
        MOV     D,A             ;DIGIT IN D
```

```
        MOV     A,C             ;NEW VALUE = OLD VALUE * 10
        RLC
        RLC
        RLC
        ADD     C               ;CHECK FOR RANGE ERROR
        JRC     NUMERR
        ADD     C               ;CHECK FOR RANGE ERROR
        JRC     NUMERR
        ADD     D               ;NEW VALUE = OLD VALUE * 10 + DIGIT
        JRC     NUMERR          ;CHECK FOR RANGE ERROR
        MOV     C,A             ;SET NEW VALUE
        DJNZ    NUM1            ;COUNT DOWN
;
; RETURN FROM NUMBER
;
NUM2:   MOV     A,C             ;GET ACCUMULATED VALUE
        RET
;
; NUMBER ERROR ROUTINE FOR SPACE CONSERVATION
;
NUMERR: JMP     ERROR           ;USE ERROR ROUTINE - THIS IS RELATIVE PT
;
; EXTRACT HEXADECIMAL NUMBER FROM COMMAND LINE
;   RETURN WITH VALUE IN REG A; ALL REGISTERS MAY BE AFFECTED
;
HEXNUM: CALL    SCANER          ;PARSE NUMBER AND PLACE IN FCBFN
HNUM0:  LXI     H,FCBFN         ;PT TO TOKEN FOR CONVERSION
        LXI     D,0             ;DE=ACCUMULATED VALUE
        MVI     B,11            ;B=CHAR COUNT
HNUM1:  MOV     A,M             ;GET CHAR
        CPI     ' '             ;DONE?
        JRZ     HNUM3           ;RETURN IF SO
        CPI     NUMBASE         ;DONE IF NUMBASE SUFFIX
        JRZ     HNUM3
        SUI     '0'             ;CONVERT TO BINARY
        JRC     NUMERR          ;RETURN AND DONE IF ERROR
        CPI     10              ;0-9?
        JRC     HNUM2
        SUI     7               ;A-F?
        CPI     10H             ;ERROR?
        JRNC    NUMERR
HNUM2:  INX     H               ;PT TO NEXT CHAR
        MOV     C,A             ;DIGIT IN C
        MOV     A,D             ;GET ACCUMULATED VALUE
        RLC                     ;EXCHANGE NYBBLES
        RLC
        RLC
        RLC
        ANI     0F0H            ;MASK OUT LOW NYBBLE
        MOV     D,A
        MOV     A,E             ;SWITCH LOW-ORDER NYBBLES
        RLC
        RLC
        RLC
        RLC
        MOV     E,A             ;HIGH NYBBLE OF E=NEW HIGH OF E,
                                ;  LOW NYBBLE OF E=NEW LOW OF D
        ANI     0FH             ;GET NEW LOW OF D
        ORA     D               ;MASK IN HIGH OF D
        MOV     D,A             ;NEW HIGH BYTE IN D
```

```
        MOV     A,E
        ANI     0F0H            ;MASK OUT LOW OF E
        ORA     C               ;MASK IN NEW LOW
        MOV     E,A             ;NEW LOW BYTE IN E
        DJNZ    HNUM1           ;COUNT DOWN
;
; RETURN FROM HEXNUM
;
HNUM3:  XCHG                    ;RETURNED VALUE IN HL
        MOV     A,L             ;LOW-ORDER BYTE IN A
        RET
;
; PT TO DIRECTORY ENTRY IN TBUFF WHOSE OFFSET IS SPECIFIED BY A AND C
;
DIRPTR: LXI     H,TBUFF         ;PT TO TEMP BUFFER
        ADD     C               ;PT TO 1ST BYTE OF DIR ENTRY
        CALL    ADDAH           ;PT TO DESIRED BYTE IN DIR ENTRY
        MOV     A,M             ;GET DESIRED BYTE
        RET
;
; CHECK FOR SPECIFIED DRIVE AND LOG IT IN IF NOT DEFAULT
;
SLOGIN: XRA     A               ;SET FCBDN FOR DEFAULT DRIVE
        STA     FCBDN
        CALL    COMLOG          ;CHECK DRIVE
        RZ
        JR      DLOG5           ;DO LOGIN OTHERWISE
;
; CHECK FOR SPECIFIED DRIVE AND LOG IN DEFAULT DRIVE IF SPECIFIED<>DEFAULT
;
DLOGIN:
        CALL    COMLOG          ;CHECK DRIVE
        RZ                      ;ABORT IF SAME
        LDA     TDRIVE          ;LOG IN DEFAULT DRIVE
;
DLOG5:  JMP     LOGIN
;
; ROUTINE COMMON TO BOTH LOGIN ROUTINES; ON EXIT, Z SET MEANS ABORT
;
COMLOG:
TEMPDR  EQU     $+1             ;POINTER FOR IN-THE-CODE MODIFICATION
        MVI     A,0             ;2ND BYTE (IMMEDIATE ARG) IS TEMPDR
        ORA     A               ;0=NO
        RZ
        DCR     A               ;COMPARE IT AGAINST DEFAULT
        LXI     H,TDRIVE
        CMP     M
        RET                     ;ABORT IF SAME
;
; EXTRACT TOKEN FROM COMMAND LINE AND PLACE IT INTO FCBDN;
;   FORMAT FCBDN FCB IF TOKEN RESEMBLES FILE NAME AND TYPE (FILENAME.TYP);
;   ON INPUT, CIBPTR PTS TO CHAR AT WHICH TO START SCAN;
;   ON OUTPUT, CIBPTR PTS TO CHAR AT WHICH TO CONTINUE AND ZERO FLAG IS RESET
;     IF '?' IS IN TOKEN
;
; ENTRY POINTS:
;       SCANER - LOAD TOKEN INTO FIRST FCB
;       SCANX - LOAD TOKEN INTO FCB PTED TO BY HL
;
SCANER: LXI     H,FCBDN         ;POINT TO FCBDN
```

```
SCANX:  XRA     A               ;SET TEMPORARY DRIVE NUMBER TO DEFAULT
        STA     TEMPDR
        CALL    ADVAN           ;SKIP TO NON-BLANK OR END OF LINE
        SDED    CIPTR           ;SET PTR TO NON-BLANK OR END OF LINE
        LDAX    D               ;END OF LINE?
        ORA     A               ;0=YES
        JRZ     SCAN2
        SBI     'A'-1           ;CONVERT POSSIBLE DRIVE SPEC TO NUMBER
        MOV     B,A             ;STORE NUMBER (A:=0, B:=1, ETC) IN B
        INX     D               ;PT TO NEXT CHAR
        LDAX    D               ;SEE IF IT IS A COLON (:)
        CPI     ':'
        JRZ     SCAN3           ;YES, WE HAVE A DRIVE SPEC
        DCX     D               ;NO, BACK UP PTR TO FIRST NON-BLANK CHAR
SCAN2:  LDA     TDRIVE          ;SET 1ST BYTE OF FCBDN AS DEFAULT DRIVE
        MOV     M,A
        JR      SCAN4
SCAN3:  MOV     A,B             ;WE HAVE A DRIVE SPEC
        STA     TEMPDR          ;SET TEMPORARY DRIVE
        MOV     M,B             ;SET 1ST BYTE OF FCBDN AS SPECIFIED DRIVE
        INX     D               ;PT TO BYTE AFTER ':'
;
; EXTRACT FILENAME FROM POSSIBLE FILENAME.TYP
;
SCAN4:  XRA     A               ;A=0
        STA     QMCNT           ;INIT COUNT OF NUMBER OF QUESTION MARKS IN FCB
        MVI     B,8             ;MAX OF 8 CHARS IN FILE NAME
        CALL    SCANF           ;FILL FCB FILE NAME
;
; EXTRACT FILE TYPE FROM POSSIBLE FILENAME.TYP
;
        MVI     B,3             ;PREPARE TO EXTRACT TYPE
        CPI     '.'             ;IF (DE) DELIMITER IS A '.', WE HAVE A TYPE
        JRNZ    SCAN15          ;FILL FILE TYPE BYTES WITH <SP>
        INX     D               ;PT TO CHAR IN COMMAND LINE AFTER '.'
        CALL    SCANF           ;FILL FCB FILE TYPE
        JR      SCAN16          ;SKIP TO NEXT PROCESSING
SCAN15: CALL    SCANF4          ;SPACE FILL
;
; FILL IN EX, S1, S2, AND RC WITH ZEROES
;
SCAN16: MVI     B,4             ;4 BYTES
SCAN17: INX     H               ;PT TO NEXT BYTE IN FCBDN
        MVI     M,0
        DJNZ    SCAN17
;
; SCAN COMPLETE -- DE PTS TO DELIMITER BYTE AFTER TOKEN
;
        SDED    CIBPTR
;
; SET ZERO FLAG TO INDICATE PRESENCE OF '?' IN FILENAME.TYP
;
        LDA     QMCNT           ;GET NUMBER OF QUESTION MARKS
        ORA     A               ;SET ZERO FLAG TO INDICATE ANY '?'
        RET

;  SCANF -- SCAN TOKEN PTED TO BY DE FOR A MAX OF B BYTES; PLACE IT INTO
;     FILE NAME FIELD PTED TO BY HL; EXPAND AND INTERPRET WILD CARDS OF
;     '*' AND '?'; ON EXIT, DE PTS TO TERMINATING DELIMITER
;
```

```
SCANF:  CALL    SDELM           ;DONE IF DELIMITER ENCOUNTERED - <SP> FILL
        JRZ     SCANF4
        INX     H               ;PT TO NEXT BYTE IN FCBDN
        CPI     '*'             ;IS (DE) A WILD CARD?
        JRNZ    SCANF1          ;CONTINUE IF NOT
        MVI     M,'?'           ;PLACE '?' IN FCBDN AND DON'T ADVANCE DE IF SO
        CALL    SCQ             ;SCANNER COUNT QUESTION MARKS
        JR      SCANF2
SCANF1: MOV     M,A             ;STORE FILENAME CHAR IN FCBDN
        INX     D               ;PT TO NEXT CHAR IN COMMAND LINE
        CPI     '?'             ;CHECK FOR QUESTION MARK (WILD)
        CZ      SCQ             ;SCANNER COUNT QUESTION MARKS
SCANF2: DJNZ    SCANF           ;DECREMENT CHAR COUNT UNTIL 8 ELAPSED
SCANF3: CALL    SDELM           ;8 CHARS OR MORE - SKIP UNTIL DELIMITER
        RZ                      ;ZERO FLAG SET IF DELIMITER FOUND
        INX     D               ;PT TO NEXT CHAR IN COMMAND LINE
        JR      SCANF3
;
;  FILL MEMORY POINTED TO BY HL WITH SPACES FOR B BYTES
;
SCANF4: INX     H               ;PT TO NEXT BYTE IN FCBDN
        MVI     M,' '           ;FILL FILENAME PART WITH <SP>
        DJNZ    SCANF4
        RET
;
;  INCREMENT QUESTION MARK COUNT FOR SCANNER
;    THIS ROUTINE INCREMENTS THE COUNT OF THE NUMBER OF QUESTION MARKS IN
;    THE CURRENT FCB ENTRY
;
SCQ:    LDA     QMCNT           ;GET COUNT
        INR     A               ;INCREMENT
        STA     QMCNT           ;PUT COUNT
        RET
;
; CMDTBL (COMMAND TABLE) SCANNER
;   ON RETURN, HL PTS TO ADDRESS OF COMMAND IF CPR-RESIDENT
;   ON RETURN, ZERO FLAG SET MEANS CPR-RESIDENT COMMAND
;
CMDSER: LXI     H,CMDTBL        ;PT TO COMMAND TABLE
;
        IF      SECURE
        MVI     C,NRCMDS
        LDA     WHEEL           ;SEE IF NON-RESTRCTED
        CPI     RESTRCT
        JRZ     CMS1            ;PASS IF RESTRCTED
        ENDIF   ;SECURE
;
        MVI     C,NCMNDS        ;SET COMMAND COUNTER
CMS1:   LXI     D,FCBFN         ;PT TO STORED COMMAND NAME
        MVI     B,NCHARS        ;NUMBER OF CHARS/COMMAND (8 MAX)
CMS2:   LDAX    D               ;COMPARE AGAINST TABLE ENTRY
        CMP     M
        JRNZ    CMS3            ;NO MATCH
        INX     D               ;PT TO NEXT CHAR
        INX     H
        DJNZ    CMS2            ;COUNT DOWN
        LDAX    D               ;NEXT CHAR IN INPUT COMMAND MUST BE <SP>
        CPI     ' '
        JRNZ    CMS4
        RET                     ;COMMAND IS CPR-RESIDENT (ZERO FLAG SET)
```

```
CMS3:   INX     H               ;SKIP TO NEXT COMMAND TABLE ENTRY
        DJNZ    CMS3
CMS4:   INX     H               ;SKIP ADDRESS
        INX     H
        DCR     C               ;DECREMENT TABLE ENTRY NUMBER
        JRNZ    CMS1
        INR     C               ;CLEAR ZERO FLAG
        RET                     ;COMMAND IS DISK-RESIDENT (ZERO FLAG CLEAR)
;
;**** Section 5 ****
; CPR-Resident Commands
;
;
;Section 5A
;Command: DIR
;Function:  To display a directory of the files on disk
;Forms:
;       DIR <afn>       Displays the DIR files
;       DIR <afn> S     Displays the SYS files
;       DIR <afn> A     Display both DIR and SYS files
;
        IF      CPRDIR
;
DIR:    MVI     A,80H           ;SET SYSTEM BIT EXAMINATION
        PUSH    PSW
        CALL    SCANER          ;EXTRACT POSSIBLE D:FILENAME.TYP TOKEN
        CALL    SLOGIN          ;LOG IN DRIVE IF NECESSARY
        LXI     H,FCBFN         ;MAKE FCB WILD (ALL '?') IF NO FILENAME.TYP
        MOV     A,M             ;GET FIRST CHAR OF FILENAME.TYP
        CPI     ' '             ;IF <SP>, ALL WILD
        CZ      FILLQ
        CALL    ADVAN           ;LOOK AT NEXT INPUT CHAR
        MVI     B,0             ;SYS TOKEN DEFAULT
        JRZ     DIR2            ;JUMP; THERE ISN'T ONE
        CPI     SYSFLG          ;SYSTEM FLAG SPECIFIER?
        JRZ     GOTSYS          ;GOT SYSTEM SPECIFIER
        CPI     SOFLG           ;SYS ONLY?
        JRNZ    DIR2
        MVI     B,80H           ;FLAG SYS ONLY
GOTSYS: INX     D
        SDED    CIBPTR
        CPI     SOFLG           ;SYS ONLY SPEC?
        JRZ     DIR2            ;THEN LEAVE BIT SPEC UNCHAGNED
        POP     PSW             ;GET FLAG
        XRA     A               ;SET NO SYSTEM BIT EXAMINATION
        PUSH    PSW
DIR2:   POP     PSW             ;GET FLAG

DIR2A:                          ;DROP INTO DIRPR TO PRINT DIRECTORY
                                ; THEN RESTART CPR
        ENDIF   ;CPRDIR
;
; DIRECTORY PRINT ROUTINE; ON ENTRY, MSB OF A IS 1 (80H) IF SYSTEM FILES
; EXCLUDED. THIS ROUTINE IS ALSO USED BY ERA.
;
DIRPR:  MOV     D,A             ;STORE SYSTEM FLAG IN D
        MVI     E,0             ;SET COLUMN COUNTER TO ZERO
        PUSH    D               ;SAVE COLUMN COUNTER (E) AND SYSTEM FLAG (D)
        MOV     A,B             ;SYS ONLY SPECIFIER
        STA     SYSTST
```

```
        CALL    SEARF           ;SEARCH FOR SPECIFIED FILE (FIRST OCCURRANCE)
        CZ      PRNNF           ;PRINT NO FILE MSG;REG A NOT CHANGED
;
; ENTRY SELECTION LOOP; ON ENTRY, A=OFFSET FROM SEARF OR SEARN
;
DIR3:   JRZ     DIR11           ;DONE IF ZERO FLAG SET
        DCR     A               ;ADJUST TO RETURNED VALUE
        RRC                     ;CONVERT NUMBER TO OFFSET INTO TBUFF
        RRC
        RRC
        ANI     60H
        MOV     C,A             ;OFFSET INTO TBUFF IN C (C=OFFSET TO ENTRY)
        MVI     A,10            ;ADD 10 TO PT TO SYSTEM FILE ATTRIBUTE BIT
        CALL    DIRPTR
        POP     D               ;GET SYSTEM BIT MASK FROM D
        PUSH    D
        ANA     D               ;MASK FOR SYSTEM BIT
SYSTST  EQU     $+1             ;POINTER TO IN-THE-CODE BUFFER SYSTST
        CPI     0
        JRNZ    DIR10
        POP     D               ;GET ENTRY COUNT (=<CR> COUNTER)
        MOV     A,E             ;ADD 1 TO IT
        INR     E
        PUSH    D               ;SAVE IT
;
        IF      TWOCOL
        ANI     01H             ;OUTPUT <CRLF> IF 2 ENTRIES PRINTED IN LINE
        ENDIF   ;TWOCOL
;
        IF      NOT TWOCOL
TWOPOK  EQU     $+1             ;FOR APPLE PATCHING
        ANI     03H             ;OUTPUT <CRLF> IF 4 ENTRIES PRINTED IN LINE
        ENDIF   ;NOT TWOCOL
;
        PUSH    PSW
        JRNZ    DIR4
        CALL    CRLF            ;NEW LINE
        JR      DIR5
DIR4:   CALL    PRINT
;
        IF      WIDE
        DB      '  '            ;2 SPACES
        DB      FENCE           ;THEN FENCE CHAR
        DB      ' ',' '+80H     ;THEN 2 MORE SPACES
        ENDIF
;
        IF      NOT WIDE
        DB      ' '             ;SPACE
        DB      FENCE           ;THEN FENCE CHAR
        DB      ' '+80H         ;THEN SPACE
        ENDIF
;
DIR5:   MVI     B,01H           ;PT TO 1ST BYTE OF FILE NAME
DIR6:   MOV     A,B             ;A=OFFSET
        CALL    DIRPTR          ;HL NOW PTS TO 1ST BYTE OF FILE NAME
        ANI     7FH             ;MASK OUT MSB
        CPI     ' '             ;NO FILE NAME?
        JRNZ    DIR8            ;PRINT FILE NAME IF PRESENT
        POP     PSW
        PUSH    PSW
```

```
        CPI     03H
        JRNZ    DIR7
        MVI     A,09H           ;PT TO 1ST BYTE OF FILE TYPE
        CALL    DIRPTR          ;HL NOW PTS TO 1ST BYTE OF FILE TYPE
        ANI     7FH             ;MASK OUT MSB
        CPI     ' '             ;NO FILE TYPE?
        JRZ     DIR9            ;CONTINUE IF SO
DIR7:   MVI     A,' '           ;OUTPUT <SP>
DIR8:   CALL    CONOUT          ;PRINT CHAR
        INR     B               ;INCR CHAR COUNT
        MOV     A,B
        CPI     12              ;END OF FILENAME.TYP?
        JRNC    DIR9            ;CONTINUE IF SO
        CPI     09H             ;END IF FILENAME ONLY?
        JRNZ    DIR6            ;PRINT TYP IF SO
        MVI     A,'.'           ;PRINT DOT BETWEEN FILE NAME AND TYPE
        CALL    CONOUT
        JR      DIR6
DIR9:   POP     PSW
DIR10:  CALL    BREAK           ;CHECK FOR ABORT
        JRNZ    DIR11
        CALL    SEARN           ;SEARCH FOR NEXT FILE
        JR      DIR3            ;CONTINUE
DIR11:  POP     D               ;RESTORE STACK
        RET
;
; FILL FCB @HL WITH '?'
;
FILLQ:  MVI     B,11            ;NUMBER OF CHARS IN FN & FT
FQLP:   MVI     M,'?'           ;STORE '?'
        INX     H
        DJNZ    FQLP
        RET
;
;Section 5B
;Command: ERA
;Function:  Erase files
;Forms:
;       ERA <afn>       Erase Specified files and print their names
;
        IF      NOT RAS         ;NOT FOR REMOTE-ACCESS SYSTEM
;
ERA:    CALL    SCANER          ;PARSE FILE SPECIFICATION
        IF      NOT EPRMPT      ;<<< IF EPRMPT TRUE, "All?" PROMPT SKIPPED
        CPI     11              ;ALL WILD (ALL FILES = 11 '?')?
        JRNZ    ERA1            ;IF NOT, THEN DO ERASES
        CALL    PRINTC
        DB      'All','?'+80H
        CALL    CONIN           ;GET REPLY
        CPI     'Y'             ;YES?
ERARJ:  JNZ     RESTRT          ;RESTART CPR IF NOT
        CALL    CRLF            ;NEW LINE
ERA1:   ENDIF                   ;<<<ADDED
        CALL    SLOGIN          ;LOG IN SELECTED DISK IF ANY
        XRA     A               ;PRINT ALL FILES (EXAMINE SYSTEM BIT)
        MOV     B,A             ;NO SYS-ONLY OPT TO DIRPR
        CALL    DIRPR           ;PRINT DIRECTORY OF ERASED FILES
;
        IF      EPRMPT
;
```

```
;  QUERY USER AFTER FILES ARE SEEN, AND GIVE ONE LAST CHANCE TO BACK OUT
;
        MOV     A,E             ;HOW MANY FILES DISPLAYED?
        ORA     A
        JZ      ERRLOG          ;<<<REPLACES LINE BELOW TO FIX BUG
                                ;(SEE "NZCPRERA.FIX")
;       JZ      RESTRT          ;IF NONE, DON'T ASK OR DELETE
        CALL    PRINTC          ;PROMPT
        DB      'Ok','?'+80H
        CALL    CONIN           ;GET REPLY FOLDED
        CPI     'Y'             ;YES?
        JNZ     ERRLOG          ;<<<REPLACES LINE BELOW TO FIX BUG
;       JRNZ    ERARJ           ;GET OUT IF NOT
        ENDIF   ;EPRMPT
;
        LXI     D,FCBDN         ;DELETE FILE(S) SPECIFIED
        JMP     DELETE          ;RESTART CPR AFTER DELETE
;
        ENDIF                   ;RAS
;
;Section 5C
;Command: LIST
;Function:  Print out specified file on the LST: Device
;Forms:
;       LIST <ufn>      Print file (NO Paging)
;
        IF      CPRLST
LIST:   MVI     A,0FFH          ;TURN ON PRINTER FLAG
        JR      TYPE0
        ENDIF                   ;CPRLST
;
;Section 5D
;Command: TYPE
;Function:  Print out specified file on the CON: Device
;Forms:
;       TYPE <ufn>      Print file
;       TYPE <ufn> P    Print file with paging flag
;
        IF      CPRTYP
TYPE:
        ENDIF                   ;CPRTYP
;
        IF      CPRTYP AND CPRLST
        XRA     A               ;TURN OFF PRINTER FLAG
;
; ENTRY POINT FOR CPR LIST FUNCTION (LIST)
;
TYPE0:  STA     PRFLG           ;SET FLAG
        ENDIF   ;CPRTYP AND CPRLST
;
        IF      CPRTYP
        CALL    SCANER          ;EXTRACT FILENAME.TYP TOKEN
        JNZ     ERROR           ;ERROR IF ANY QUESTION MARKS
        CALL    ADVAN           ;GET PGDFLG IF IT'S THERE
        STA     PGFLG           ;SAVE IT AS A FLAG
        JRZ     NOSLAS          ;JUMP IF INPUT ENDED
        INX     D               ;PUT NEW BUF POINTER
        XCHG
        SHLD    CIBPTR
```

```
NOBOP:  CALL    LCOUT           ;PRINT IT
;
; CONTINUE PROCESSING
;
;
TYPE2L: CALL    BREAK           ;CHECK FOR ABORT
        JRZ     TYPE1           ;CONTINUE IF NO CHAR
        CPI     'C'-'@'         ;^C?
        RZ                      ;RESTART IF SO
        JR      TYPE1

TYPE3:  DCR     A               ;NO ERROR?
        RZ                      ;RESTART CPR

TYPE4:  JMP     ERRLOG
        ENDIF   ;CPRTYP
;
;Section 5E
;Command: SAVE
;Function:  To save the contents of the TPA onto disk as a file
;Forms:
;       SAVE <Number of Pages> <ufn>
;                               Save specified number of pages (start at 100H)
;                               from TPA into specified file; <Number of
;                               Pages> is in DEC
;       SAVE <Number of Sectors> <ufn> S
;                               Like SAVE above, but numeric argument specifies
;                               number of sectors rather than pages
;
        IF      NOT RAS         ;NOT FOR REMOTE-ACCESS SYSTEM
;
SAVE:   CALL    NUMBER          ;EXTRACT NUMBER FROM COMMAND LINE
        MOV     L,A             ;HL=PAGE COUNT
        MVI     H,0
        PUSH    H               ;SAVE PAGE COUNT
        CALL    EXTEST          ;TEST FOR EXISTENCE OF FILE AND ABORT IF SO
        MVI     C,16H           ;BDOS MAKE FILE
        CALL    GRBDOS
        POP     H               ;GET PAGE COUNT
        JRZ     SAVE3           ;ERROR?
        XRA     A               ;SET RECORD COUNT FIELD OF NEW FILE'S FCB
        STA     FCBCR
        CALL    ADVAN           ;LOOK FOR 'S' FOR SECTOR OPTION
        INX     D               ;PT TO AFTER 'S' TOKEN
        CPI     SECTFLG
        JRZ     SAVE0
        DCX     D               ;NO 'S' TOKEN, SO BACK UP
        DAD     H               ;DOUBLE IT FOR HL=SECTOR (128 BYTES) COUNT
SAVE0:  SDED    CIBPTR          ;SET PTR TO BAD TOKEN OR AFTER GOOD TOKEN
        LXI     D,TPA           ;PT TO START OF SAVE AREA (TPA)
SAVE1:  MOV     A,H             ;DONE WITH SAVE?
        ORA     L               ;HL=0 IF SO
        JRZ     SAVE2
        DCX     H               ;COUNT DOWN ON SECTORS
        PUSH    H               ;SAVE PTR TO BLOCK TO SAVE
        LXI     H,128           ;128 BYTES PER SECTOR
        DAD     D               ;PT TO NEXT SECTOR
        PUSH    H               ;SAVE ON STACK
        CALL    DMASET          ;SET DMA ADDRESS FOR WRITE (ADDRESS IN DE)
        LXI     D,FCBDN         ;WRITE SECTOR
```

```
        MVI     C,15H           ;BDOS WRITE SECTOR
        CALL    BDOSB           ;SAVE BC
        POP     D               ;GET PTR TO NEXT SECTOR IN DE
        POP     H               ;GET SECTOR COUNT
        JRZ     SAVE1           ;CONTINUE IF NO WRITE ERROR
        JR      PRNLE           ;GO PRINT ERROR AND RESET DMA
SAVE2:  LXI     D,FCBDN         ;CLOSE SAVED FILE
        CALL    CLOSE
        INR     A               ;ERROR?
        JRNZ    SAVE3           ;PASS IF OK
;
        ENDIF   ;NOT RAS

;
;  PRNLE IS ALSO USED BY MEMLOAD FOR TPA FULL ERROR
;
PRNLE:  CALL    PRINTC          ;DISK OR MEM FULL
        DB      'Ful','l'+80H
;
SAVE3:  JMP     DEFDMA          ;SET DMA TO 0080 AND RESTART CPR
                                ; OR RETURN TO MLERR
;
        IF      NOT RAS
;
; Test File in FCB for existence, ask user to delete if so, and abort if he
;  choses not to
;
EXTEST: CALL    SCANER          ;EXTRACT FILE NAME
        JNZ     ERROR           ;'?' IS NOT PERMITTED
        CALL    SLOGIN          ;LOG IN SELECTED DISK
        CALL    SEARF           ;LOOK FOR SPECIFIED FILE
        LXI     D,FCBDN         ;PT TO FILE FCB
        RZ                      ;OK IF NOT FOUND
        PUSH    D               ;SAVE PTR TO FCB
        CALL    PRINTC
        DB      'Erase','?'+80H
        CALL    CONIN           ;GET RESPONSE
        POP     D               ;GET PTR TO FCB
        CPI     'Y'             ;KEY ON YES
        JNZ     RSTCPR          ;RESTART IF NO, SP RESET EVENTUALLY
        PUSH    D               ;SAVE PTR TO FCB
        CALL    DELETE          ;DELETE FILE
        POP     D               ;GET PTR TO FCB
        RET
;
        ENDIF                   ;RAS
;
;Section 5F
;Command: REN
;Function:  To change the name of an existing file
;Forms:
;       REN <New ufn>=<Old ufn> Perform function
;
        IF      NOT RAS         ;NOT FOR REMOTE-ACCESS SYSTEM
;
R       CALL    EXTEST          ;TEST FOR FILE EXISTENCE AND RETURN
                                ; IF FILE DOESN'T EXIST; ABORT IF IT DOES
        LDA     TEMPDR          ;SAVE CURRENT DEFAULT DISK
        PUSH    PSW             ;SAVE ON STACK
RENO:   LXI     H,FCBDN         ;SAVE NEW FILE NAME
```

```
        LXI     D,FCBDM
        LXI     B,16            ;16 BYTES
        LDIR
        CALL    ADVAN           ;ADVANCE CIBPTR
        CPI     '='             ;'=' OK
        JRNZ    REN4
REN1:   XCHG                    ;PT TO CHAR AFTER '=' IN HL
        INX     H
        SHLD    CIBPTR          ;SAVE PTR TO OLD FILE NAME
        CALL    SCANER          ;EXTRACT FILENAME.TYP TOKEN
        JRNZ    REN4            ;ERROR IF ANY '?'
        POP     PSW             ;GET OLD DEFAULT DRIVE
        MOV     B,A             ;SAVE IT
        LXI     H,TEMPDR        ;COMPARE IT AGAINST CURRENT DEFAULT DRIVE
        MOV     A,M             ;MATCH?
        ORA     A
        JRZ     REN2
        CMP     B               ;CHECK FOR DRIVE ERROR
        MOV     M,B
        JRNZ    REN4
REN2:   MOV     M,B
        XRA     A
        STA     FCBDN           ;SET DEFAULT DRIVE
        LXI     D,FCBDN         ;RENAME FILE
        MVI     C,17H           ;BDOS RENAME FCT
        CALL    GRBDOS
        RNZ
REN3:   CALL    PRNNF           ;PRINT NO FILE MSG
REN4:   JMP     ERRLOG
;
        ENDIF                   ;RAS
;
;Section 5G
;Command: USER
;Function:  Change current USER number
;Forms:
;       USER <unum>     Select specified user number;<unum> is in DEC
;
        IF      DRUSER          ;IF DRIVE/USER CODE OK...
USER:   CALL    USRNUM          ;EXTRACT USER NUMBER FROM COMMAND LINE
        MOV     E,A             ;PLACE USER NUMBER IN E
SUSER:  CALL    SETUSR          ;SET SPECIFIED USER
        ENDIF   ;DRUSER

RSTJMP: JMP     RCPRNL          ;RESTART CPR
;
;Section 5H
;Command: DFU
;Function:  Set the Default User Number for the command/file scanner
;           (MEMLOAD)
;          Note: When under SECURE mode, this will select the second
;                user area to check for programs (normally user 15).
;
;Forms:
;       DFU <unum>      Select Default User Number;<unum> is in DEC
;
        IF      NOT RAS         ;NOT FOR REMOTE-ACCESS SYSTEM
DFU:    CALL    USRNUM          ;GET USER NUMBER
        STA     DFUSR           ;PUT IT AWAY
        JR      RSTJMP          ;RESTART CPR (NO DEFAULT LOGIN)
```

```
        ENDIF   ;NOT RAS
;
;Section 5I
;Command: JUMP
;Function:  To Call the program (subroutine) at the specified address
;           without loading from disk
;Forms:
;       JUMP <adr>              Call at <adr>;<adr> is in HEX
;
        IF      NOT RAS         ;NOT FOR REMOTE-ACCESS SYSTEM
;
JUMP:   CALL    HEXNUM          ;GET LOAD ADDRESS IN HL
        JR      CALLPROG        ;PERFORM CALL
;
        ENDIF                   ;RAS
;
;Section 5J
;Command: GO
;Function:  To Call the program in the TPA without loading
;           from disk. Same as JUMP 100H, but much
;           more convenient, especially when used with
;           parameters for programs like STAT. Also can be
;           allowed on remote-access systems with no problems.
;
;Form:
;       GO <parameters like for COMMAND>
;
        IF      NOT RAS         ;ONLY IF RAS
;
GO:     LXI     H,TPA           ;Always to TPA
        JR      CALLPROG        ;Perform call
;
        ENDIF                   ;END OF GO FOR RAS
;
;Section 5K
;Command: COM file processing
;Function:  To load the specified COM file from disk and execute it
;Forms:
;       <command>
;
COM:    LDA     FCBFN           ;ANY COMMAND?
        CPI     ' '             ;' ' MEANS COMMAND WAS 'D:' TO SWITCH
        JRNZ    COM1            ;NOT <SP>, SO MUST BE TRANSIENT OR ERROR
        LDA     TEMPDR          ;LOOK FOR DRIVE SPEC
        ORA     A               ;IF ZERO, JUST BLANK
        JZ      RCPRNL
        DCR     A               ;ADJUST FOR LOG IN
        STA     TDRIVE          ;SET DEFAULT DRIVE
        CALL    LOGIN           ;LOG IN DRIVE
        CALL    SETUD           ;SET DRIVE WITH CURRENT USER AREA
;
        IF      DRUSER          ;DRIVE/USER HACKERY OK?
        CALL    USRNUM          ;GET USER #, IF ANY
        MOV     E,A             ;GET IT READY FOR BDOS
        LDA     FCBFN           ;SEE IF # SPECIFIED
        CPI     ' '
        JRNZ    SUSER           ;SELECT IF WANTED
        ENDIF   ;DRUSER
;
        JMP     RCPRNL          ;RESTART CPR
```

```
COM1:   LDA     FCBFT           ;FILE TYPE MUST BE BLANK
        CPI     ' '
        JNZ     ERROR
        LXI     H,COMMSG        ;PLACE DEFAULT FILE TYPE (COM) INTO FCB
        LXI     D,FCBFT         ;COPY INTO FILE TYPE
        LXI     B,3             ;3 BYTES
        LDIR
        LXI     H,TPA           ;SET EXECUTION/LOAD ADDRESS
        PUSH    H               ;SAVE FOR EXECUTION
        CALL    MEMLOAD         ;LOAD MEMORY WITH FILE SPECIFIED IN CMD LINE
                                ; (NO RETURN IF ERROR OR TOO BIG)
        POP     H               ;GET EXECUTION ADDRESS
;
; CALLPROG IS THE ENTRY POINT FOR THE EXECUTION OF THE LOADED
;   PROGRAM. ON ENTRY TO THIS ROUTINE, HL MUST CONTAIN THE EXECUTION
;   ADDRESS OF THE PROGRAM (SUBROUTINE) TO EXECUTE
;
CALLPROG:
        SHLD    EXECADR         ;PERFORM IN-LINE CODE MODIFICATION
        CALL    DLOGIN          ;LOG IN DEFAULT DRIVE
        CALL    SCANER          ;SEARCH COMMAND LINE FOR NEXT TOKEN
        LXI     H,TEMPDR        ;SAVE PTR TO DRIVE SPEC
        PUSH    H
        MOV     A,M             ;SET DRIVE SPEC
        STA     FCBDN
        LXI     H,FCBDN+10H     ;PT TO 2ND FILE NAME
        CALL    SCANX           ;SCAN FOR IT AND LOAD IT INTO FCBDN+16
        POP     H               ;SET UP DRIVE SPECS
        MOV     A,M
        STA     FCBDM
        XRA     A
        STA     FCBCR
        LXI     D,TFCB          ;COPY TO DEFAULT FCB
        LXI     H,FCBDN         ;FROM FCBDN
        LXI     B,33            ;SET UP DEFAULT FCB
        LDIR
        LXI     H,CIBUFF-1
COM4:   INX     H
        MOV     A,M             ;SKIP TO END OF 2ND FILE NAME
        ORA     A               ;END OF LINE?
        JRZ     COM5
        CPI     ' '             ;END OF TOKEN?
        JRNZ    COM4
;
; LOAD COMMAND LINE INTO TBUFF
;
COM5:   MVI     B,-1            ;SET CHAR COUNT
        LXI     D,TBUFF         ;PT TO CHAR POS
        DCX     H
COM6:   INR     B               ;INCR CHAR COUNT
        INX     H               ;PT TO NEXT
        INX     D
        MOV     A,M             ;COPY COMMAND LINE TO TBUFF
        STAX    D
        ORA     A               ;DONE IF ZERO
        JRNZ    COM6
;
; RUN LOADED TRANSIENT PROGRAM
;
```

```
COM7:   MOV     A,B             ;SAVE CHAR COUNT
        STA     TBUFF
        CALL    CRLF            ;NEW LINE
        CALL    DEFDMA          ;SET DMA TO 0080
        CALL    SETUD           ;SET USER/DISK
;
; EXECUTION (CALL) OF PROGRAM (SUBROUTINE) OCCURS HERE
;
EXECADR EQU     $+1             ;CHANGE ADDRESS FOR IN-LINE CODE MODIFICATION
        CALL    TPA             ;CALL TRANSIENT
        CALL    DEFDMA          ;SET DMA TO 0080, IN CASE
                                ;PROG CHANGED IT ON US
        CALL    SETUOD          ;SET USER 0/DISK
        CALL    LOGIN           ;LOGIN DISK
        JMP     RESTRT          ;RESTART CPR
;
;Section 5L
;Command: GET
;Function:  To load the specified file from disk to the specified address
;Forms:
;       GET <adr> <ufn> Load the specified file at the specified page;
;                       <adr> is in HEX
;
        IF      NOT RAS         ;NOT FOR REMOTE-ACCESS SYSTEM
;
GET:    CALL    HEXNUM          ;GET LOAD ADDRESS IN HL
        PUSH    H               ;SAVE ADDRESS
        CALL    SCANER          ;GET FILE NAME
        POP     H               ;RESTORE ADDRESS
        JNZ     ERROR           ;MUST BE UNAMBIGUOUS
;
; FALL THRU TO MEMLOAD
;
        ENDIF                   ;RAS
;
; Load memory with the file whose name is specified in the command line
; on input, HL contains starting address to load.
;
; Exit bact to caller if no error. If the COM file is too big, or another
; error, then exit directly to MLERR.
;
MEMLOAD:
        SHLD    LOADADR         ;SET LOAD ADDRESS
        CALL    GETUSR          ;GET CURRENT USER NUMBER
        STA     TMPUSR          ;SAVE IT FOR LATER
        STA     TSELUSR         ;TEMP USER TO SELECT
;
;   MLA is a reentry point for a non-standard CP/M Modification
; This is the return point for when the .COM (or GET) file is not found the
; first time, Drive A: is selected for a second attempt
;
MLA:    CALL    SLOGIN          ;LOG IN SPECIFIED DRIVE IF ANY
        CALL    OPENF           ;OPEN COMMAND.COM FILE
        JRNZ    MLA1            ;FILE FOUND - LOAD IT
:
        IF      SECURE
;
; If secure is enabled, search the current drive, current user, then
; if in wheel mode, search under last user set by DFU (set to "resusr"
; on warm boot) on current drive. If not found, or not in wheel mode,
```

```
; then search on current drive under user area "defusr." If file still
; hasn't been found, then do the same thing again except on drive A:
;
DFLAG   EQU     $+1             ;MARK IN-THE-CODE VARIABLE
        MVI     A,0             ;HAVE WE CHECKED THIS DRIVE ALREADY?
        ORA     A
        JRNZ    MLAO            ;PASS IF SO TO GO TO DRIVE A:
        LDA     WHEEL           ;RESTRICTED PROGS ALLOWED?
        CPI     RESTRCT
        JRZ     MLAOO           ;PASS IF NOT
        PUSH    B               ;PUSH BC
        LDA     DFUSR           ;LOAD DEFAULT USER
        MOV     B,A             ;PUT IT IN B
        LDA     TSELUSR         ;CHECK CURR USER
DFUSR   EQU     $+1             ;DEFAULT USER LOCATION
        CPI     RESUSR          ;RESTRICTED USER?
        MOV     A,B             ;ASSUME NOT
        POP     B               ;RESTORE BC
        JRNZ    SETTSE          ;GO TRY IF NOT
MLAOO:                          ;SS IF NOT
TSELUSR EQU     $+1             ;MARK IN-THE-CODE VARIABLE
        MVI     A,0             ;GET CURR USER
        SUI     DEFUSR          ;IS IT UNRESTRICTED COM AREA?
        JRZ     MLAO            ;NO MORE CHOICES IF SO
        STA     DFLAG           ;MAKE DFLAG NON-ZERO IF NOT
        MVI     A,DEFUSR        ; AND TRY UNRESTRICTED COM AREA
SETTSE:
        ENDIF   ;SECURE
;
        IF      NOT SECURE
DFUSR   EQU     $+1             ;MARK IN-THE-CODE VARIABLE
        MVI     A,DEFUSR        ;GET DEFAULT USER
TSELUSR EQU     $+1             ;MARK IN-THE-CODE VARIABLE
        CPI     DEFUSR          ;CHECK FOR THE USER AREA..
        JRZ     MLAO            ;..EQUAL DEFAULT, AND JUMP IF SO
        ENDIF   ;NOT SECURE
;
        STA     TSELUSR         ;PUT DOWN NEW ONE
        MOV     E,A
        CALL    SETUSR          ;GO SET NEW USER NUMBER
        JR      MLA             ;AND TRY AGAIN
;
; Error routine to select drive A: if default was originally selected
;
MLAO:
        LXI     H,TEMPDR        ;GET DRIVE FROM CURRENT COMMAND
        XRA     A               ;A=0
;
        IF      SECURE
        STA     DFLAG           ;ALLOW A: SEARCH
        ENDIF   ;SECURE
;
        ORA     M
        JNZ     MLERR           ;ERROR IF ALREADY DISK A:
        MVI     M,1             ;SELECT DRIVE A:

        IF      NOT SECURE
        JR      MLA
        ENDIF   ;NOT SECURE
;
```

```
        IF      SECURE
        LDA     TMPUSR          ;GO TO 'CURRENT' USER CODE
        JR      SETTSE
        ENDIF   ;SECURE

; FILE FOUND -- PROCEED WITH LOAD
;
MLA1:
LOADADR EQU     $+1
        LXI     H,TPA
ML2:    MVI     A,ENTRY/256-1   ;GET HIGH-ORDER ADR OF JUST BELOW CPR
        CMP     H               ;ARE WE GOING TO OVERWRITE THE CPR?
        JRC     ML4             ;ERROR IF SO
        PUSH    H               ;SAVE ADDRESS OF NEXT SECTOR
        XCHG                    ;... IN DE
        CALL    DMASET          ;SET DMA ADDRESS FOR LOAD
        LXI     D,FCBDN         ;READ NEXT SECTOR
        CALL    READ
        POP     H               ;GET ADDRESS OF NEXT SECTOR
        JRNZ    ML3             ;READ ERROR OR EOF?
        LXI     D,128           ;MOVE 128 BYTES PER SECTOR
        DAD     D               ;PT TO NEXT SECTOR IN HL
        JR      ML2
;
ML3:    DCR     A               ;LOAD COMPLETE
        JZ      RESETUSR        ;IF ZERO, OK, GO RESET CORRECT USER #
                                ; ON WAY OUT, ELSE FALL THRU TO PRNLE
;
;  TPA FULL
;
ML4:    CALL    PRNLE           ;PRINT MSG AND RESET DEF DMA
;
; TRANSIENT LOAD ERROR
;
MLERR:
                                ;NOTE THAT THERE IS AN EXTRA RETURN ADDRESS ON
                                ;THE STACK.  IT WILL BE TOSSED WHEN ERROR
                                ;EXITS TO RESTRT, WHICH RELOADS SP.
        CALL    RESETUSR        ;RESET CURRENT USER NUMBER
                                ; RESET MUST BE DONE BEFORE LOGIN
ERRLOG: CALL    DLOGIN          ;LOG IN DEFAULT DISK
        JMP     ERROR           ;FLAG ERROR
;
;
;Section: 5M
;PASS:  Enable wheel mode.
;NORM:  Disable wheel mode.
;
;  Type PASS <password> <cr> to CP/M prompt to enter wheel mode.
; This code can be replaced with PST's PASS.ASM which gives many
; nice little options like no keyboard echo, etc.
;
        IF      INPASS          ;WE WANT TO USE THIS CODE, NOT PASS.COM
PASS:   LXI     H,PASSWD        ;SET UP POINTERS
        LXI     D,CIBUFF+NCHARS+1
        MVI     B,PASEND-PASSWD ;B= LENGTH
CKPASS: LDAX    D               ;TRIAL PW TO A
        CMP     M               ;CHECK FOR MATCH
        JNZ     COM             ;NOPE.. LOOK FOR PASS.COM
        INX     H               ;INCREMENT COUNTER
```

```
        INX     D
        DJNZ    CKPASS          ;CONTINUE IF MORE
        MVI     A,NOT RESTRCT   ;WHEEL = NOT RESTRCT

OUT:    STA     WHEEL
        JMP     RESTRT
;
NORM:   MVI     A,RESTRCT
        JR      PWOUT
;
PASSWD: DB      'YOURPW'        ;YOUR PASSWORD
PASEND: EQU     $               ;END OF PASSWORD
;
        ENDIF                   ;INPASS
;
        END
```

ONZCPR21.MSG
REVISED 04/03/83

Some Advantages of the ZCPR (Z-80 based Console Processor Replacement) over the standard CCP (Console Command Processor):

--> You can TYPE a file without having to use the ^S. You can select the Paging option (usually assembled to default on this option) so that text is scrolled a page at a time and waits for a <CR> to display the next page.

--> If you log onto Drive B: you can call a .COM file on either Drive without specifying a Drive.

--> There are other features you might want to use, and can tailor the Replacement CCP to your own needs (using the .ASM file and a MACRO-ASSEMBLER).

****
* *
* HOW TO SET UP NZCPR-21.HEX ON THE DD (CBIOS 1.4) OSBORNE I *
* *
****

1) On a single or double density disk (formatted and Sysgened for the 1.4 BIOS) put the following files:

DDT.COM, MOVCPM.COM, SYSGEN.COM Plus one of the .HEX versions of NZCPR-21 especially assembled for the Osborne I ((ONZCPR21.HEX, Available on many RCPMs, is one.... I have assembled NZCPR-21.HEX for my own uses (mainly so I KNOW EXACTLY what options have been selected) and intend to submit it -- along with documentation-- to the FOG Library))

2) Insert this disk in drive A and boot up on it

3) A>MOVCPM 59 *<CR> (Enter this line, "<CR>" is the "RETURN" key)

CONSTRUCTING 59K CP/M vers. 2.2 (This message appears)

READY FOR "SYSGEN" OR "SAVE 39 CP/M 59.COM" (This message appears)

4) A>SAVE 39 CPM59.COM<CR> (Enter this line)

5) A>DDT CPM59.COM<CR> (Enter this line)

DDT VERS 2.2 (This message appears)
NEXT PC
2800 0100

6) -IONZCPR21.HEX<CR> (Enter this line if using ONZCPR21.HEX; otherwise, use

I plus the name of the .HEX
file (use no spaces))

7) -R3E80<CR> (Enter this line) The standard CCP has now been overlayed by the ZCPR

NEXT PC (This message appears)
2800 0000

8) At this point, you may want to modify CPM59.COM so that you can boot up on any .COM file you wish (or none -- if that's what you want). You can save the space the AUTOST.COM takes, and the extra time it uses up. Here are the locations you may want to modify (Use the "S"command of DDT):

201C --> This location contains a number which designates an autobooting option:

0 = No autoboot (The computer will boot up, but won't load a .COM file. The screen clears, the cursor homes and

Osborne Computer System
59K CP/M vers. 2.2
CBIOS 1.4

appears.

1 = Autoboot on cold start

2 = Autoboot on warm start

3 = Autoboot on both

201D--> Length (plus 1) of the name of the file to be Autobooted (length of file name only -- the file type, .COM, is assumed)

201E--> Name of the file to be Autostarted begins here. (again, the file type is assumed) The name takes up a maximum of 8 memory locations... 201D indicates the actual number used... It is not necessary to fill the unused locations with spaces (20 HEX).

NOTE: If you want to save programmed keys, you might want to save them (use SETUP.COM) on a blank diskette before you move the ZCPR to your CPM disk, etc... UNLESS you know how to modify the system tracks directly (see Doug Hurst's article in the March FOGHORN in order to learn how), you may have to MANUALLY reprogram the keys, because SETUP.COM also moves the Autoboot area...

TER you have made all desired modifications:

9) ^C (Enter "CTRL C")

10) A>SYSGEN<CR> (Enter this line)

11) Next you are asked for the source drive. Hit "<CR>", which takes the system from memory.

12) Next you are asked for destination. Hit "A<CR>"

13) Hit another "<CR>" to leave SYSGEN. If you have used ONZCPR21.HEX, your cursor will look like this: --> A0> (my version of NZCPR-21 is set not to display the user area, if user 0)

14) Transfer the replacement CCP to your other disks using the disk in Drive A: and SYSGEN.COM.

J.E. Crowell
San Jose, CA