QP/M 2.7

Installation Guide

and Supplements

August, 1985

Created by
MICROCode Consulting
www.microcodeconsulting.com

CP/M 2.2 is a registered trademark of Digital Research, Incorporated. Z-80 is a registered trademark of Zilog, Incorporated. NSC-800 is a trademark of National Semiconductor Corporation. WordStar is a trademark of MicroPro International. Macro-80, Link-80, and Lib-80 are registered trademarks of MicroSoft Corporation.

DISCLAIMER

MICROCode Consulting makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantibility or fitness for any particular purpose. Further, MICROCode Consulting reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of MICROCode Consulting to notify any person of such revision or changes.

TABLE OF CONTENTS

A. Introduction

QP/M 2.7 is a superset of CP/M 2.2. It is intended to upgrade any CP/M system which uses a Z80 or NSC-800 microprocessor, to a faster, more powerful operating system. Original CP/M packages included the CCP (Command Control Processor) and BDOS (Basic Disk Operating System). The CP/M 2.2 CCP has been totally replaced by an upgraded version called QCP which includes a most important TIME command; CP/M 2.2 BDOS has been replaced by QP/M 2.7 QDOS. Two new unique features appear in QDOS: Drive and User Search whenever a file is opened, and automatic time/date stamping of files. Your original CP/M 2.2 BIOS (Basic Input/Output System) can be used as is; however, in order to use the time/date capability, a new vector (called TIMDAT) must be patched somewhere into your system to return the current date and time. (This vector does not have to be in BIOS; see the Supplemental Section.)

If you have purchased one of the QBIOS packages with QP/M, you only need to refer to section D if you wish to change the default settings of QP/M. The remaining sections may be ignored.

B. Restrictions

QP/M is fully compatible with the CP/M 2.2 operating system. However, there are some restrictions imposed in order to run QP/M:

1. You should have a minimum of 32k of memory in your system. This restriction is imposed by the QINSTALL program as two complete system images plus loaders must exist together in memory during installation. If you have a smaller system and are the adventurous type, you can try to install QP/M; depending on your SYSGEN program, it may or may not work.

2. You must either be currently running CP/M 2.2 (so you can use your existing BIOS) or be able to program in assembly language to write your own BIOS. MICROCode Consulting has pre-written BIOS packages for a number of popular systems, but we do not have a BIOS for all Z-80 systems. (If you do not have a BIOS, it usually means that you also lack an INIT and a SYSGEN program which are even more hardware-specific than a BIOS.) Contact MICROCode Consulting to determine if your system is directly supported with QBIOS and QBOOT modules or to recieve technical help should you choose to write your own.

3. Unless your machine is specifically supported, it will help to have some knowledge in programming in order to establish the QP/M-to-real-time-clock connection (as detailed later). The new TDCNFG utility will be of assistance. If you are still unable to make the connection, the time/date stamping feature of QP/M will be disabled. If you purchased QP/M solely for this purpose, please contact MICROCode Consulting for assistance, or for return shipping instructions to obtain a full refund.

C. Before Installation

Your BIOS configuration determines how the default drive/user is set within QCP and if the QCP can be preset to execute a command/program on cold boot. If your BIOS supports the execution of a program on "cold" boot (called auto-boot), then it is configured correctly. If not, it is recommended that auto-boot capability be added, although it is not a requirement.

Upon "cold" entry into QCP (at location QCP + 0000H), QDOS is set to an initial default drive/user (specified during installation via QINSTALL), and any autoboot command/program will be executed. Should you decide to change the default drive and/or user, QCP modifies the memory locations associated with these values. Upon "warm" entry into QCP (at location QCP + 0003H), QDOS is reset to the most recent values of default drive/user specified in the QCP. (The auto-boot command/program will NOT be executed).

In order for your BIOS to support auto-boot capability, the BIOS must use the "cold" entry into QCP during cold boot and use the "warm" entry thereafter.

If your BIOS does not support auto-boot, you must disable this feature of QP/M; otherwise, upon every cold AND warm boot, QDOS will be set back to the initial default drive/user as specified during QINSTALL. Even worse, if you have specified an initial command/program for QCP, *every* warm boot will repeat the command/program.

A more important feature to implement is the QP/M-to-real-time-clock interface. In order to have FULL time/date capability with QP/M, your system must support a real-time clock in either hardware or software. To interface with QP/M, a jump vector called TIMDAT must be set up (anywhere in memory) to return the address of the "clock string" in register pair HL. The "clock string" is defined as a 6-byte string as follows:

```
HL   +0  +1    +2   +3   +4  +5
     day month year hour min sec
```

These values must be in binary with the time in 24-hour format. For example, November 17, 1985 at 6:12 p.m. would be expressed as:

```
     HL   +0 +1 +2 +3 +4 +5
(decimal) 17 11 85 18 12 00
  (hex)   11 0B 55 12 0C 00
```

Let's assume that the system has a software clock located at 0F504H and there are 7 bytes free at 0FF79H. The code would be as follows:

```
FF79:  TIMDAT:      JMP FF7C
FF7C:               LXI H,F504
FF7F:               RET
```

Note the jump vector (at FF79H), which is *always* the first instruction of a TIMDAT routine.

Since MICROCode Consulting realizes that not every system supports a real-time clock, or, even more important, that a required clock module has not been loaded from disk yet, QDOS examines the first byte of the TIMDAT routine. If it is *not* a JMP (0C3H) instruction, the clock is considered missing.

If you are currently running a software-updated clock on your system, your QP/M TIMDAT routine should perform any conversion, if necessary, into binary format and place the values in the order listed above.

Due to the variety of systems available, MICROCode Consulting provides a time/date configuration utility TDCNFG.COM. This program will assist in setting up the proper TIMDAT interface routine for most systems, and will facilitate installation of QP/M. It is recommended that TDCNFG.COM be executed before QINSTALL.COM is executed since TDCNFG.COM displays the location of the QP/M-to-real-time-clock interface vector which is required by QINSTALL.COM. ALL clock hardware that MICROCode Consulting has tested is included in TDCNFG.COM; please contact us if your clock is missing.

Refer to the Supplements for further information.

D. Using QINSTALL

Before running QINSTALL, you need to determine the user-definable values. This is most easily done by previewing the QINSTALL program options as they appear during actual operation. Take your time and understand what each value means. That way, the first execution of QINSTALL will take only a minute or two, and you will be running under QP/M in no time.

Note that you should have your SYSGEN program (could be another name) on a disk that is in your system. Since QINSTALL could not possibly write to the system tracks of every CP/M machine, it uses your original SYSGEN program to create the new system. Of course, once you have created QP/M on one of your disks, SYSGEN can be used, as always, to transfer the system to other disks. QINSTALL should only be run for first-time installation or when you wish to change the user-definable values.

When you are ready, take a few minutes to review the QINSTALL options shown below. Each option is listed, along with the default values. The blank column to the right can be used to fill in your own values.

```
          ***   QCP System Installation settings   ***

                                                  Default New Value

<0>  Number of lines on screen ......................... 24    _____

<1>  Show user number when user is zero ................ YES   _____

<2>  QCP commands enabled .............................. ALL   _____

<3>  Pause when screen fills during TYPE ............... YES   _____

<4>  Pass control characters to console/printer ......... NO    _____

<5>  Printer top-of-form character (decimal) ............ 12    _____

<6>  Standard QP/M prompt .............................. >     _____

<7>  Maximum user number allowed by QCP ................ 15    _____

<8>  Show SYStem files in DIR listing .................. NO    _____

<9>  Separator character between files listed with DIR .. :     _____

<A>  Default drive setting after cold boot .............. A     _____

<B>  Default user setting after cold boot ............... 0     _____

<C>  Word (16-bits) to hold default drive and user at ... 0008H _____

<D>  Location to hold user number when greater than 15 .. n/a   _____

<E>  Initial program/command to execute:       <none>   _______________
```

What the options mean:

<0> Enter the vertical height of your screen in lines. This value is only used to determine pause position during a TYPE command.

<1> QCP displays the user number after the drive letter when the current user area is not zero. Optionally, QCP will always show the user number when set to YES.

<2> If you are running your system as a Remote System, setting this value to NONE will disable all transient QCP commands. You can also remove selective commands by selecting PARTIAL which will bring up another menu described later.

<3> When TYPEing a file under QCP, it will normally wait after the page is full. Pressing any key will resume printing of the next page, and so on. Setting this value to FALSE will result in no pauses unless the 'P' option of TYPE is used.

<4> Whenever an unknown command is entered (including any command with non-printable control character present), the QCP echoes the command followed by a question mark. However, if any control character is present, the QCP only prints the command up to that character. Allowing all characters to pass through on command echo (setting the option to YES) may let you initialize the printer or clear the screen by entering the control character string. However, beware of console or printer side-effects that other control characters may have if entered.

<5> Most printers use 12 decimal (0DH, 'FF' or FormFeed in ASCII) as the top-of-form character. This value is sent to the printer whenever the TOF transient command is executed.

<6> Prompt during normal (user) console entry of QCP (A>, B>).

<7> Maximum user number allowed with the USER command. Can be changed to protect higher user areas during normal operation. The usual maximum is 15. However, the maximum can be as high as 31 with the extended user capability within QCP. An additional memory location is required when the user number is greater than 15 as specified by option <C>.

<8> Normally, the transient DIR command does not display files tagged as SYStem. If this value is set to YES, DIR will also display files which have a SYStem tag.

<9> Separator character used between files displayed on the screen when using the DIR or ERA command.

<A> Default drive set upon "cold" entry into QCP. Can be any drive A through O or disabled. This value is used by QDOS when searching for files. (A default drive of P is not allowed.)

<B> Default user set upon "cold" entry into QCP. Can be any user between 0 and 14 or disabled. This value is used by QDOS when searching for files. (A default user area of 15 or higher is not allowed.)

<C> Address of 2-byte (word) value which holds the default drive and user. This is used by the QCP during warm boot to restore default drive and user to the value last specified by the transient DFU and DFD commands. This value will normally be left as is; however, if this location conflicts with other parts of your system, you have the option of locating the default values elsewhere in memory.

<D> Location to hold user number when option <7> is greater than 15. Since 31 user areas cannot be specified by the usual drive/user flag used in QCP, an additional location is required. The value defaults to the byte following the default drive/user locations of <B>. However, you may choose another value.

<E> Use this option to specify a command that is executed on each "cold" entry into QCP. If your BIOS does NOT have auto-boot capability, do NOT enter any command as it will be re-executed on EVERY warm boot. Maximum string length is 40.

When option <2> is set to Partial, a menu of all the QCP internal commands appears. This menu will *not* appear if either the ALL or NONE option for QCP commands is selected. Consult QP/M Features and Facilities (Users Guide) for a more complete description of each command.

```
                QCP commands that are enabled:

<0>  DFD Set default drive ....................... YES   _____

<1>  DFLT Show default drive and user ............. YES   _____

<2>  DFU Set default user ........................ YES   _____

<3>  DIR Show directory .......................... YES   _____

<4>  ERA Erase file .............................. YES   _____

<5>  GET Get file into memory .................... YES   _____

<6>  GO Execute (re-execute) program in TPA ...... YES   _____

<7>  JUMP Execute routine at the specified address . YES   _____

<8>  LIST List file to printer ................... YES   _____

<9>  REN Rename file ............................. YES   _____

<A>  SAVE Save TPA to disk ....................... YES   _____

<B>  TIME Show current date/time ................. YES   _____

<C>  TOF Send Top-of-form character to printer ..... YES   _____

<D>  TYPE Display file ........................... YES   _____

<E>  USER Set user area .......................... YES   _____
```

The other area of QP/M that has user-definable values is the QDOS. The QDOS "image" inside QINSTALL is shipped with values that will allow QP/M to run on an un-modified CP/M system. Each option is listed below along with the default values. The blank column to the right is an aid for filling in your own values.

```
     ***   QDOS System Installation settings   ***

                                                  Default New Value

<0>  Save character obtained during console scan .. YES       ______

<1>  BIOS supports BDOS error code table ......... NO        ______

<2>  Address of time/date jump vector ............. DISABLED  ______

<3>  Drive/user search feature .................... ENABLED   ______

<4>  Automatic disk re-log ........................ ENABLED   ______
```

What the options mean:

<0> When characters are being sent TO the console, a check is made for a <CTRL+S> FROM the console. A <CTRL+S> puts QDOS into a loop until another character is pressed; both <CTRL+S> and the additional character are always discarded. However, if a key other than a <CTRL+S> is pressed, the choice you make will determine what happens:

YES: This will save the character until the next console input request. This setting is recommended for interrupt-driven console input to allow type-ahead. However, since QDOS can only hold one character at a time, typing a non-<CTRL+S> character will disable <CTRL+S> checking until that character is read via a console input call.

NO: This will discard the character. This setting is recommended for a non-interrupt-driven console input. The advantage here is that typing any character other than a <CTRL+S> will not disable the <CTRL+S> check which can occur with YES.

<1> During disk operations (read/write), BIOS returns an error code in the A-register. A value of 00H means success, any other value means an error occurred.

If your BIOS does not support the disk error protocol, then all read/write errors give the message "QDOS error on d: Bad Sector", since QDOS only checks for a zero/non-zero value. If your BIOS supports the disk error protocol, then QDOS uses the value in the A-register to give a more specific explanation:

```
       register A = 0: No error
                    1: Bad Sector - permanent error found
                    2: Select Error - drive not ready
                    3: R/O - drive is read-only (write protected)
```

If your BIOS supports the BDOS disk error protocol, then answer YES. If it does not or you are not sure, then answer NO.

<2> This option is used to enter the location of the real-time clock-to-QP/M interface vector (TIMDAT). Owners of QBIOS will find the TIMDAT address listed within Section A of the QP/M Supplements. If you do not have QBIOS, you will have to enter your own value or a value provided by TDCNFG.COM. Note that a value of zero is designated as disabling the QP/M-to-real-time-clock interface.

<3> Normally, the drive/user search feature of QP/M is enabled. However, it is possible to unconditionally disable drive/user search (effectively disabling the DFU and DFD as well as QDOS functions 41 and 42).

<4> This option will toggle the automatic disk re-log feature of QP/M. In some applications, it may be desirable to force the user to press <CTRL+C> and get a message if the disk has been changed. Normally, forcing the user to perform disk reset is redundant.

Once all of the user-definable values have been set, QINSTALL will generate a system size the same as that you are currently running. When you see the message

```
Press <C> to continue or <A> to abort
```

QINSTALL is ready to proceed with the actual system installation. If special installation is required due to QINSTALL not finding the system image, or if you wish to specify a different size QP/M system than the current one, you would type the special installation option <CTRL+F> at this point. Otherwise, press <C> and QINSTALL will run your SYSGEN program.

When you enter the SYSGEN program the first time, you should instruct SYSGEN to READ the system image from a bootable disk. Once you have finished reading the system image, DO NOT continue and WRITE the system image as QP/M has not yet been installed. Instead, do a normal EXIT from the SYSGEN program and QINSTALL will resume control. After exiting SYSGEN, a message like

```
System image found at xxxxH
```

should appear. If it does not, a message will be displayed informing you the system image was not found. If this happens, it is either because the system image on disk is different than the one currently running, the image is not located on a page (256-byte) boundary, or the system tracks of the disk you just read do not contain a system. In any case, you should run QINSTALL again using another disk which contains a system. If QINSTALL fails to find the image again, experienced users should refer to sections E and F for further help. Otherwise, contact MICROCode Consulting for technical help.

Once the previous system image has been found, QINSTALL overlays the QP/M portions into the system image. The message

```
Press <C> to continue or <A> to abort
```

will appear. You must now instruct your SYSGEN program to WRITE the system image (DO NOT READ again). Now exit the SYSGEN program, and QP/M will be installed on your disk. You can now use your SYSGEN program normally to move QP/M from this disk to any other disks just as you would CP/M.

NOTE: Once you first instruct SYSGEN to READ the system tracks, QINSTALL will overlay QP/M into your system image. If you READ the system tracks again (when QINSTALL returns you to SYSGEN the second time), you will destroy the QP/M image and you will have to start the QINSTALL process over again. Most SYSGEN programs give a message like

```
Enter source drive (RETURN to skip)
```

You should press <RETURN> since the image is already in memory, then WRITE the image onto the destination drive.

E. Specifying the System Image Address

Normally, QINSTALL searches for the system image automatically, to accommodate the wide variety of SYSGEN programs which have images at unknown addresses. If for some reason QINSTALL failed to find the system image, it is possible to force QINSTALL to put the QP/M image at any arbitrary address. However, you MUST be sure of the system image address or QP/M will not be installed properly. Execute QINSTALL up to the message:

```
Press <C> to continue  -OR-  <A> to abort --
```

Pressing **<CTRL+F>** will put QINSTALL into special installation mode; you will be prompted for the address.

F. Creating an Arbitrary System Size

This section of QINSTALL should only be selected if you wish to install a system size which is NOT the same as your current system size. Note that your BIOS and BOOT routines must also be changed as QINSTALL will not modify these.

If you are not running CP/M 2.2 and wish to change the system size, you must first modify the BIOS and BOOT of the system image, then write it to disk with SYSGEN. (The system tracks are not usable yet.) In this case, it is likely you will have to specify the system image location (via **<CTRL+F>**) as well as the new system size.

To specify an arbitrary system size, you must complete the normal QINSTALL process up to the message:

```
Press <C> to continue  -OR-  <A> to abort --
```

Pressing **<CTRL+U>** will put QINSTALL into the user-definable system size mode. QP/M can be configured in 1/4k increments by first specifying the integer size, then the 1/4k size. Using this option does not affect the system image search, which is automatic. After generating the new system size, you will be returned to the normal QINSTALL procedure.

If the installation is unsuccessful, you may first have to specify the system image address, as explained later.

G. QINSTALL Summary

BEFORE running QINSTALL, verify the following:

1. Minimum 32k memory size

2. A jump vector TIMDAT has been set up in memory to return the address of the "clock string" in register pair HL (optional)

3. QCP and QDOS installation options have been determined in advance

4. SYSGEN or similar program is on one of the available drives

5. Size of the system located on the system tracks of your disk (which SYSGEN will read), and the operating system currently in memory are the same

H. Specific BIOS Utilities

Included here are specific BIOS utilities for the Xerox 820-I, Xerox 820-II and BigBoard-I and –II computers.

H-1. QBIOS1/SETX1

QBIOS1 is a special BIOS for the Xerox 820-I operating under QP/M. The program package consists of QBIOS1, QBOOT1, and SETX1.

QBOOT1 loads the BIOS and configures a parallel printer, if one is attached. The COMM port is also initialized to 1200 baud, 8-bit, 2 stop bits, no parity. Assembly conditionals in the QBOOT1 source file must be set for type of disk system (5.25" or 8"), step rate, and type of printer (serial or parallel) being attached. After you have assembled and linked to create QBOOT1.COM, the file can be overlayed into the system image at 0900H.

QBIOS1 is the interface between QP/M and the Xerox hardware. It has been designed specifically for the standard Xerox monitor for single density disk systems. It is configured such that the clock MUST be initialized BEFORE the user is allowed to enter QP/M. Essentially, QBIOS1 executes SETX1 so that a date and time may be entered. The auto-boot capability of QP/M is still available to the user.

QBIOS1 can be configured for disk system (5.25" or 8", single or double side), type of printer (serial through the PRINTER port or parallel through the internal parallel port), and whether to check that the clock has a valid time before allowing the user to enter QP/M. QBIOS1.COM is overlayed into the system image at 01F80H.

SETX1 is a special program which loads a clock into high memory (at FD00H in the monitor) and accepts the current date and time from the user. Once set, the time can only be reset by performing another cold boot. The user may optionally display the time in the upper right of the screen.

During execution of QINSTALL, you will need to know the location of the clock routine. SETX1 installs the time/date jump vector at 0FD00H. Enter this value for option <2> in the QDOS Installation section.

If a hardware clock is installed in your system, you will probably want to execute another program to load the clock; this can be done by either changing the name of your clock program to SETX1 or changing the name of the program to be executed in QBIOS1. At present, MICROCode Consulting supports a limited number of hardware clock loaders. Please contact MICROCode for specific clock initialization modules available.

A sample session showing installation of QBOOT1.COM and QBIOS1.COM is included on the next page.

```
A>SYSGEN1


SYSGEN v1.0  (c) 1985 MICROCode Consulting

Xerox 820-I with 8" drive(s)


Enter source drive <A-D> or <RETURN> if system image in memory: A
Read system from A.  Press <RETURN> to continue --
Function complete.


Enter destination drive <A-D> or <CTRL+C> to exit: ^C

Exiting to system.

A>SAVE 40 QPM.SYS
A>DDT QPM.SYS
DDT VERS 2.2
NEXT PC
2900 0100
-IQBIOS1.COM
-R1E80
NEXT PC
2900 0100
-IQBOOT1.COM
-R800
NEXT PC
2900 0100
-I              <--- 'I' MUST be followed by a 'space' or 'blank'
-G100


SYSGEN v1.0  (c) 1985 MICROCode Consulting

Xerox 820-I with 8" drive(s)


Enter source drive <A-D> or <RETURN> if system image in memory:


Enter destination drive <A-D> or <CTRL+C> to exit: B
Write system to B.  Press <RETURN> to continue --
Function complete.


Enter destination drive <A-D> or <CTRL+C> to exit: ^C

Exiting to system.
```

H-2. QBIOSB1/SETB1

QBIOSB1 is a special BIOS for the BigBoard-I operating under QP/M. The program package consists of QBIOSB1, QBOOTB1, and SETB1.

QBOOTB1 loads the BIOS and configures a parallel printer, if one is attached. Port A is also initialized to 1200 baud, 8-bit, 2 stop bits, no parity. Assembly conditionals in the QBOOT1 source file must be set for type of disk system (5.25" or 8"), step rate, and type of printer (serial or parallel) being attached. After you have assembled and linked to create QBOOTB1.COM, the file can be overlayed into the system image at 0900H.

QBIOSB1 is the interface between QP/M and the BigBoard hardware. It has been designed specifically for the standard BigBoard monitor for single density disk systems. It is configured such that the clock MUST be initialized BEFORE the user is allowed to enter QP/M. Essentially, QBIOSB1 executes SETB1 so that a date and time may be entered. The auto-boot capability of QP/M is still available to the user.

QBIOSB1 can be configured for any standard disk system having assembler equates for the type of disk system (5.25" or 8", single or double side), type of printer (serial through the port B or parallel through the internal parallel port), and whether to check that the clock has a valid time before allowing the user to enter QP/M. QBIOSB1.COM is overlayed into the system image at 1F80H.

SETB1 is a special program which loads a clock into high memory (at FD00H in the monitor) and accepts the current date and time from the user. Once set, the time can only be reset by performing another cold boot. The user may optionally display the time in the upper right of the screen.

During execution of QINSTALL, you will need to know the location of the clock routine. SETB1 installs the time/date jump vector at 0FD00H, enter this value for option <2> in the QDOS Installation section.

If a hardware clock is installed in your system, you will probably want to execute another program to load the clock; this can be done by either changing the name of your clock program to SETB1 or changing the name of the program to be executed in QBIOSB1. At present, MICROCode Consulting supports a limited number of hardware clock loaders. Please contact MICROCode for specific clock initialization modules available.

Installing QBOOTB1/QBIOSB1

```
A>SYSGENB


SYSGEN v1.0  (c) 1985 MICROCode Consulting

BigBoard-I with 8" drive(s)


Enter source drive <A-D> or <RETURN> if system image in memory: A
Read system from A.  Press <RETURN> to continue --
Function complete.


Enter destination drive <A-D> or <CTRL+C> to exit: ^C

Exiting to system.

A>SAVE 40 QPM.SYS
A>DDT QPM.SYS
DDT VERS 2.2
NEXT PC
2900 0100
-IQBIOSB1.COM
-R1E80
NEXT PC
2900 0100
-IQBOOTB1.COM
-R800
NEXT PC
2900 0100
-I              <--- 'I' MUST be followed by a 'space' or 'blank'
-G100


SYSGEN v1.0  (c) 1985 MICROCode Consulting

BigBoard-I with 8" drive(s)


Enter source drive <A-D> or <RETURN> if system image in memory:


Enter destination drive <A-D> or <CTRL+C> to exit: B
Write system to B.  Press <RETURN> to continue --
Function complete.


Enter destination drive <A-D> or <CTRL+C> to exit: ^C

Exiting to system.
```

H-3. QBIOS2/SETX2

QBIOS2 is a special BIOS for the Xerox 820-II operating under QP/M. The program package consists of QBIOS2, QBOOT2, and SETX2.

QBOOT2 loads the BIOS and sets the system for the type of disk drives (5.25" or 8") connected.

QBIOS2 is the interface between QP/M and the Xerox hardware. It has been designed specifically for the standard Xerox 820-II monitor which supports the physical disk driver concept. It is configured such that the clock MUST be initialized BEFORE the user is allowed to enter QP/M. Essentially, QBIOS2 executes SETX2 so that a date and time may be entered. The auto-boot capability of QP/M is still available to the user (settable through either CONFIGUR.COM or QINSTALL.COM).

QBIOS2 is fully compatible with the Xerox software and utilities. (Note, however, that CONFIGUR.COM must be modified so it does a check for QP/M rather than for CP/M.)

SETX2 is a special program which accepts the current date and time from the user. Once set, the time can only be reset by powering down the system first. The user may optionally display the time in the upper right of the screen.

During execution of QINSTALL, you will need to know the location of the clock routine. The Xerox 820-II has the time/date vector located at 0F039H. Enter this value for option <2> in the QDOS Installation section.

If a hardware clock is installed in your system, you will probably want to execute another program to load the clock; this can be done by either changing the name of your clock program to SETX2 or changing the name of the program to be executed in QBIOS2. At present, MICROCode Consulting supports a limited number of hardware clock loaders. Please contact MICROCode Consulting for specific clock initialization modules available.

I. QP/M Utility Patching Information

This section is included for experienced users who wish to patch the QP/M utilities. It is possible to set all user-definable values to your preferred values by customizing programs QSTAMP(X/V), QPATCH, QSTAT, and QINSTALL. Any user wishing to customize any of the utilities should have working knowledge of DDT or any of the debugging programs.

I-1. QSTAMP/QSTAMPX/QSTAMPV

When QSTAMP(X/V) time/date stamps a disk for use by QP/M, a 2k (4k) directory program (D.COM/DV.COM) is also written to the disk. QSTAMP(X) displays a horizontally sorted directory; QSTAMPV displays a vertically sorted directory. QSTAMP(X/V) can be patched for these conditions:

1. Write another directory program to the disk
2. Don't write any program to the disk
3. Change the name of D(V).COM (e.g., to XD.COM)

In QSTAMP, D.COM may be replaced by any 2k or smaller directory program. When using DDT, I(nsert) the new directory filename.COM, R(ead) with offset 993H, G(o) to location 00H, then SAVE xx pages using the modified QSTAMP filename [SAVE 18 pages for QSTAMP].

In QSTAMPX(V), D.COM may be replaced by any 4k or smaller directory program. When using DDT, I(nsert) the new directory filename.COM, R(ead) with offset 993H, S(usbsitute), change byte 048C from 18 to the number of 128-byte records of your directory program (maximum of 32), then end S(ubstitute) with a period (.) Next, G(o) to location 00H, then SAVE xx pages using the modified QSTAMPX(V) filename [SAVE 26 pages for QSTAMPX(V)].

QSTAMP(X/V) can be modified so that NO directory program is written to disk. Using DDT, change byte 0459H from CD to C3, then SAVE xx filename.COM.

You can change the name of the D(V).COM file that QSTAMP(X/V) writes by changing the filename that is located at 0A72H. The name must be EXACTLY 8 uppercase characters (assumed extension of .COM); and must be padded with blanks to fill the 8 positions if the name is actually shorter.

NOTE: The QPATCH utility allows you to change certain parameters in the D.COM program. If you change the name of the directory program in QSTAMP(X/V) from D.COM to some other name, you should also change the name in QPATCH. Please note that the parameters QPATCH modifies only apply to the original QP/M D(V).COM programs. Any attempt to change D.COM parameters (within QPATCH) of other (non-QP/M) directory programs will have unpredictable results.

I-2. QPATCH

The QPATCH initial settings and "Restore default settings" options set the default values for QBACKUP.COM and all versions of D.COM. The locations of these values and value ranges are included below.

Description Location Range

QBACKUP

Value	**Location**	**Range**
Maximum user number – 1	0131H	0 through 14 (default: 14)
Lines per screen	0132H	16 through 80 (default: 24)

D (2k) or D (4k)

Value	**Location**	**Range**	**Default**
Lines to display before pause	0127H	16 through 80	24
File separator character	0128H	*	:
Width of screen in columns	0129H	03 for <80; 04 for 80+	04
Show user number	012AH	0FFH to show; otherwise 0	0FFH
Show SYStem files	012BH	0FFH to show; otherwise 0	0FFH
Show date/time	012CH	0FFH to show; otherwise 0	0
Reset before directory	012DH	0FFH to reset; otherwise 0	0
Show .LBR files	012EH	0FFH to show; otherwise 0	0

D (4k)

Value	**Location**	**Range**	**Default**
Maximum drive + 1	012FH	1 through 16 (for A - P)	16
Maximum user + 1	0130H	1 through 16 (for 0 - 15)	16

The names of the files that QPATCH modifies can also be altered, if you have changed the names of the original QP/M utilities. The names are EXACTLY 8 uppercase characters (assumed extension of .COM); the names MUST be padded with blanks to fill the 8 positions if the name is actually shorter.

Description	**Location**	**Default**
2k Directory program	0103H	'D '
4k Directory program	010BH	'D '
- Vertical format	0113H	'D '
Backup program	0118H	'QBACKUP '

I-3. QINSTALL

Whenever a system is installed with QINSTALL, default values for the QCP are used unless the user specifically enters a new value. These default values can be customized to suit your tastes; patching locations are listed in the table below. *NOTE that two consecutive locations must be changed as QINSTALL uses the first as the actual value and the second for restoring default settings.* For example, in "Suppress user number ... ", both locations 0104H and 0105H must be changed to the value desired. The only exception is the Time/date vector location where the value is only set once.

Description	**Starting Location**	**Values**	**Default Value**
Suppress user number when user is zero	0103H	CDH: show user 0C4H: do not show user	0C4H
Screen size in lines	0105H	Range from 16 to 80 decimal	24
QCP commands enabled	0107H	0C2H: ALL 0C3H: NONE 000H: Partial (see table on next page)	0C2H
Pause when screen full during TYPE	0109H	028H: pause normally 020H: no pause normally	028H
Top-of-form character	010BH	*	12 dec.
System prompt	010DH	Character + 80H (high bit set)	'>'+80H
Maximum user number	010FH	Range 0 through 15 decimal	15
Show SYStem files in DIR listing	0111H	000H: do not show 080H: show system files	080H
Separator character for DIR and ERA file listing	0113H	*	':'
Initial default drive	0115H	Range 0 through 14 decimal	0
Initial default user	0117H	Range 0 through 14 decimal	0
Max. user number flag #1	0119H	001H: Less than 16 decimal 0CDH: Greater than 15 decimal	001H
Max. user number flag #2	011BH	0C9H: Less than 16 decimal 078H: Greater than 15 decimal	0C9H
Pass control characters	011DH	021H: NO (do not pass) during echo 001H: YES (do pass)	021H
Default drive/user location	011FH	WORD (16-bit) locn (NOTE: 4 bytes to change)	0008H
User number location (> 15)	0132H	WORD (16-bit) locn (NOTE: 4 bytes to change)	000AH
Time/date vector location	0138H	WORD (16-bit) locn (EXCEPTION: ONLY change 2 bytes)	0000H

Included on the next page are the locations which define whether the given QCP command will be enabled or disabled. Note that this table is *only* used if QCP commands are partially enabled (the

value of 0107H is 000H). If the byte at the command location is 000H, the command is enabled; if the byte at the location is 080H, the command is disabled. No other values are allowed.

QCP Command Location (000H enables; 080H disables)

DFD	0123H
DFLT	0124H
DFU	0125H
DIR	0126H
ERA	0127H
GET	0128H
GO	0129H
JUMP	012AH
LIST	012BH
REN	012CH
SAVE	012DH
TIME	012EH
TOF	012FH
TYPE	0130H
USER	0131H

I-4. QSTAT

The logical and physical device names used in QSTAT are those originally defined by Digital Research. However, any of the names can be altered by consulting the table below. A ***maximum*** of 6 characters are allowed for the name which ***must*** end in a colon (:) and be in uppercase. The physical names are grouped with the corresponding logical devices.

Description/Name	**Location**
Logical CON:	0196H
physical TTY:	011EH
physical CRT:	0124H
physical BAT:	012AH
physical UC1:	0130H
Logical RDR:	019CH
physical TTY:	0136H
physical PTR:	013CH
physical UR1:	0142H
physical UR2:	0148H
Logical PUN:	01A2H
physical TTY:	014EH
physical PTP:	0154H
physical UP1:	015AH
physical UP2:	0160H
Logical LST:	01A8H
physical TTY:	0166H
physical CRT:	016CH
physical LPT:	0172H
physical UL1:	0178H

J. Interfacing a real-time clock to QP/M

This section is intended as a supplement for experienced users who are familiar with assembly language. It details the QP/M-to-real-time-clock interface to fully utilize the time/date stamping features of QP/M.

Your system must have a software operated clock (say off an interrupt counter or CTC) or hardware clock or a combination, whereby a software-operated clock is initialized by hardware at power-up. The interface that QP/M works through is independent of the actual clock installed.

When QDOS fetches the current time/date information, it executes a routine (called `TIMDAT`). The address of `TIMDAT` must be specified during QINSTALL via option <2> of the QDOS installation menu. `TIMDAT` can be anywhere in memory (*except* 0000H). As such, QDOS does not just run off and execute it. First, it *checks* for a JMP (0C3H) as the first instruction of this routine. If it is not a jump, QDOS does *not* perform the operation. This is a safety measure since many systems load a clock routine AFTER booting QP/M; the system could crash if QDOS just arbitrarily executed the routine at the assigned address if nothing is there yet.

The first instruction of the `TIMDAT` vector must be a jump instruction to a user routine which returns a pointer in register pair HL to the current time/date information. Thus, it might jump to a monitor routine, a BIOS routine, or a special routine which is loaded into memory after cold boot. This jump must return with the register pair HL pointing to the time/date information, a 6-byte block of memory which is arranged as follows:

```
HL + 00: day (1-31 decimal)
   + 01: month (1-12 decimal)
   + 02: year (0-99 decimal)
   + 03: hour (0-23 decimal, in 24-hour format)
   + 04: minute (0-59 decimal)
   + 05: second (0-59 decimal)
```

There are a few systems which do not store the time/date information in binary format (for example, binary-code decimal or BCD is sometimes used) or that may arrange the bytes differently. Furthermore, some hardware systems do not allocate any memory for a clock. If this is the case, then the `TIMDAT` routine for your system must perform any hardware read required (if any), and convert your specific clock information into the QP/M format somewhere in memory.

In summary, the `TIMDAT` routine for interfacing QP/M to your real-time clock is as follows.

1. Establish a JuMP vector anywhere in memory (the address of which is entered during QINSTALL) which jumps to your system- specific time/date routine.

2. Have your system-specific time/date routine RETurn with register pair HL pointing to the time/date 6-byte block in the QP/M format shown above.

Your assembly code should follow this example:

```
  +--- Address of TIMDAT entered during QINSTALL
  v
TIMDAT:  JMP USERCLK   <--- First instruction MUST be a Jump.
 .
 .             ...  possibly some other code or storage
 .
USERCLK:  ... perform any hardware read, if req'd ...
          ... perform any time/date conversion if necessary ...

     LD HL,USERDT   <---  Return pointer to time/date string
     RET
 .
 .             ...  possibly some other code or storage
 .
USERDT:   DB day         <---  6-byte storage area containing
          DB month the time/date information
          DB year
          DB hour
          DB minute
          DB second
```

If you have trouble installing the TIMDAT vector with your system's time/date information in QP/M format, contact MICROCode Consulting for further assistance.

K. Public Domain Utilities

Included in the remainder of this section is a brief description of five public domain utilities that are included on your QP/M distribution disk. These utilities are the sole work of the original author and are *not connected with MICROCode Consulting in any way*. MICROCode has included these popular utilities and printed the brief descriptions at no charge as these programs may be difficult to obtain (depending on whether you own a modem and where you are located).

[With all legacy Z-80 programs moved into the Public Domain by MICROCode Consulting, several of utilities documented in the original distribution are no longer included as they have been replaced by the corresponding (and better) MICROCode Consulting utility.]

K-1. DCON 1.1

DCON is no longer part of the distribution. Please download and use the vastly more powerful and versatile Debug-Z instead from the MICROCode Consulting website.

K-2. PDLOAD

Public Domain Hex Loader
Created by Mitchell Mlinar

OVERVIEW: PDLOAD is a public domain hex loader with a few extra features not found in the original Digital Research LOAD.COM program. It is *not* restricted to files that can ONLY run at 100H (start of the TPA) and has a loader module for files that need to execute elsewhere in memory.

OPERATION: PDLOAD is invoked by typing

```
d>PDLOAD filename
```

where 'filename' must have an extension of .HEX (a drive specifier is optional). If the file is found, PDLOAD proceeds to load the Intel Hex format file, converting it into executable object code (.COM or .CIM) while verifying its integrity via the checksum.

PDLOAD requires that the .HEX file loads from low to high memory. Although the file does not have to be continuous (there may be a "break" in memory for data storage which is skipped by the .HEX file), it must be consecutive and non-overlapping. When PDLOAD detects the .HEX file "backing up" in memory, it will abort with a message.
After the load is complete, PDLOAD checks the base (ORG) address of the file. If it is 100H (start of the TPA), PDLOAD writes the object file with the same name as the .HEX file, but with an extension of .COM. For example, TEST.HEX will produce TEST.COM.

If the .HEX file origin is less than the base of the TPA (100H), PDLOAD warns the user that this is a non-standard object file and writes the file with an extension of .CIM instead of .COM.

Finally, if the .HEX file origin is greater than 10FH, PDLOAD will give a message and save the object file along with a special loader that automatically moves the object code to its proper location first, and then executes it at its proper address.

If the .HEX file origins between 101H and 10FH, PDLOAD will abort with an error message.

For example, assume that you wrote a transient screen dump program that needs to reside from C000H to C7FFH. Assume also your DOS is at A800H. Write your file to ORG at C000H and assemble it with Z80MR or similar.

After the file is loaded, PDLOAD recognizes that this is not a normal object file. First, it writes a loader which will move the object code from low memory (PDLOAD locates the non-standard object code at 0110H) into high memory (at C000H). Then it writes the screen dump program. Unlike LOAD, PDLOAD does not save all the memory in-between. Rather, it saves only the code written by the user (using a loader) to move the object code where it belongs thereby saving disk space and speeding up load time.

Non-standard object files (except those origined below 100H) *always* have the same format. A memory map is shown on the next page.

```
0100:     LD HL,codend      ;end of relocated code
0103:     LD DE,whend       ;end of where it will go
0106:     LD BC,size        ;code size
0109:     LDDR                ;tail first load
010B:     JP where          ;execute the code
010E:     DB 'MM'           ;guess who!
0110:     <your code goes here>
xxxx:     <end of code>       ;<---  codend points here

where:    <where code belongs>
whend:    <end of relocated code>
```

K-3. NSWEEP

NSWEEP is no longer part of the distribution. Included in the normal QP/M distribution is QSWEEP, a time/date and speedy version of SWEEP from MICROCode Consulting.

K-4. VDO

```
                                VDO

                          A Full Screen Editor
                       Created by James H. Whorton

            Documentation revised 10/85 by MICROCode Consulting
```

Included below is a brief summary of the VDO 2.5b documentation, reduced from the original documentation by James Whorton.

VDO is a memory based-editor. Both the editor and the file being edited reside in memory, so with a 64K QP/M system, you can edit a maximum file size of about 52K. VDO uses a subset of standard WordStar keystroke commands. Where possible the keystrokes to activate a VDO operation are exactly the same as for WordStar.

INSTALLATION

To begin using VDO, follow the instructions below. The distribution package contains a number of files that make up VDO v2.5(b). The subset provided for QP/M users is:

```
VDO.COM VINST.COM VTERM.DAT
```

As distributed, VDO is installed for Kaypro computers. If this installation fits your needs, skip to the OPERATION section. Otherwise, to install VDO.COM, type VINST <RETURN>. This will start the install module written for the editor. You will then be prompted for the source filename (VDO.COM) and the destination (.COM) filename to which the installed version of the editor will be written (e.g. VDONEW.COM). If you enter a <RETURN> for the destination filename, the source file will be overwritten. This is not recommended unless you have a backup of the original source file.

A menu is then displayed, allowing you to either select a terminal to install, or exit the program. After a valid terminal is selected, the selection is confirmed, and the editor is installed with the new terminal control codes. You will see a message to that effect, and the program will terminate.

You should now have a copy of VDO ready to run on your system. Type VDO (or whatever you named the destination file) to begin. Once inside the editor, try various commands, watching the results to ensure that the editor is properly installed. If it doesn't look right, it may not be installed for the proper terminal type. Check your owner’s manual and make sure you have selected the proper terminal.

OPERATION

VDO is started by typing “VDO” or “VDO d:filename”.

In the first version, where no filename is specified, VDO will create a new file when you exit, asking for the filename at that time. If the second version is used, VDO will attempt to open the specified file. If no file of the specified name exists, VDO will display an error message. You may continue from that point, and the name specified will be used to save the new file. If the file specified can be located, VDO will load the file and place the cursor at the start of the file.

VDO's commands are broken into four major groups: Cursor Control, Quick Commands, Block Commands and Help Commands. Several minor commands are also implemented. Any non-control character which can't be interpreted as a command will be entered into the file as text.

All input to the editor, with the exception of pause prompts, is buffered in a 128 character type-ahead buffer. This means that you don't have to wait for the editor or display to catch up to you; just keep typing. Characters may be lost during disk read/write operations, since the buffer routines are not true interrupts.

Cursor Control

The standard WordStar cursor controls ^A, ^S, ^D, ^F, ^E, ^X, ^R and ^C are all implemented.

^S	move cursor a character to the left
^D	move cursor a character to the right
^A	move cursor a word to the left
^F	move cursor a word to the right
^E	move cursor up a line
^X	move cursor down a line
^R	scroll one page up (usually about 18 lines)
^C	scroll one page down

Single Keystroke Commands

TAB	Both are accepted at any time, and entered into the file RETURN as text.
^L	Repeats last Find/Find & Replace command.
^P	Enter a printer (control) code - The next character typed will be embedded into the text as a control character.
^V	Toggle Insert mode on/off
^T	Delete word to the right
^Y	Delete entire line
^G	Delete character right
DEL	Delete character left

Quick Commands

The Quick Commands are accessed by typing ^Q. If in Normal or Novice level, a menu will appear showing Quick options. Most Quick commands rapidly move the cursor to a different position in the text.

^QR Go to beginning of file.
^QC Go to end of file.
^QB Go to beginning of block, if this marker is set (see ^K B).
^QK Go to end of block, if the marker is set (see ^K K).
^QF Find string in file. Options: ignore case (assume upper and lower case are equivalent; search backwards.
^QA Find-and-replace. Works just like ^QA, except that when the string is found, a string of your choice is substituted for the target string.

Note: Strings containing Carriage Returns may not be searched for, but any other control character may be included in the target string (applies to both ^QA and ^QF).

^QT Set tab stops. VDO allows tabs to be set to 8, 2, 4, or 16, with the default at 8. This is a dynamic process, and is used mainly for programmers for variable indentations.

Block Commands

Block commands are accessed by typing ^K. A "block" is the section of a text file found between a Begin Block marker ^KB and an End Block marker ^KK.

^KB Set Begin Block marker.
^KK Set End Block marker.
^KC Copy marked block to the cursor position.
^KV Move marked block to the cursor position.
^KY Delete marked block.
^KW Write marked block into the specified file.
^KR Read disk file into the current file at the cursor position.
^KP Print specified file. You will be asked for any printer setup codes, which are entered via keystrokes.
^KZ Zaps (erases) current file from memory (does not affect any disk versions of the file), and erases the filename.
^KX End of Edit. Saves the file under the current name and creates a backup file if a version exists on disk. If VDO was invoked without a filename, or if you ZAPped the file, VDO will ask for a new filename.
^KQ Abandon file - any changes to the file (unless you did a ^KS) are abandoned. Any disk versions of the current file are not affected.
^KS Save and continue. Saves the file as it appears in memory, and continues the editing session.
^KF Displays a directory of the currently logged drive/user area.

^KL Change logged drive. Upon selection, the currently logged drive is displayed and the user is prompted for the new drive to log.

To change disks, remove the old disk, insert the new one, press ^KL and log the new disk. VDO resets the disk system before writing files, so the disk may changed between reading a file in and saving a file, or in case of system failure (e.g. DISK FULL, BAD SECTOR).

Help commands

The help commands are accessed by typing ^J. If in Normal or Novice level, a menu will appear showing Help options.

^JH Set help level (Novice is default). In this mode all menus, including basic cursor control information, are displayed whenever an extended command is initiated (^K, ^Q, ^J). In Normal mode the basic cursor control information is suppressed. In Expert mode all menus are suppressed. The menus are still available through the help commands, ^J, of course.
^JK Display Block Command menu.
^JQ Display Quick Command menu.
^JM Display basic cursor control menu.
^JJ Display help function menu.

ERROR HANDLING

VDO has nine error conditions. When one occurs, VDO will clear the screen, display the error message, and wait for you to press the ESACPE key to clear the message. Error messages are:

FILE TOO BIG - occurs when loading a file to start the editing session or using ^KR. In the case of ^KR, none of the file is retained: the text remains as it was before you attempted ^KR. This message can also occur when attempting to insert text when your system memory size is exceeded.

INVALID KEY - occurs when an entry contains an "illegal" keystroke. This can happen during text entry when an undefined control character is typed or when entering filenames, if invalid characters are entered as part of the filename.

INPUT/OUTPUT FAILURE - occurs when a requested file can't be found during initial load or ^KR, or when any system failure occurs (e.g. DIRECTORY FULL, BAD SECTOR, R/O). Can also occur when trying to read an empty file.

STRING NOT FOUND - occurs during find or find-and-replace operations if the target string can't be located.

DISK FULL - self explanatory. You may insert another disk and try again.

BLOCK NOT MARKED - occurs when attempting a block operation which requires the block to be marked. Also can occur if both the block start and block end markers aren't marked, or if the end of the block precedes the start of the block.

BLOCK STRADDLES CURSOR - occurs when trying to move a block into itself. The cursor can't be inside the block when copying or moving a block.

ILLEGAL TAB STOP - occurs if attempting to set invalid dynamic tab stops (only 2, 4, 8, 16 are valid).

K-5. Z80MR

```
                              Z80MR

                        A Z80 Macro Assembler
                      Created by Dave Trebrink

          Documentation revised 10/85 by MICROCode Consulting
```

Included below is a brief overview derived from the original Z80MR documentation.

Z80MR is a Z80 macro assembler with syntax closely following RMAC and MAC. It assembles standard Z80 mnemonics into an Intel Hex format. The resulting file (which has a .HEX extension) can be translated to a .COM file with LOADQ.COM if it originates at 100H. If it originates elsewhere, the .HEX file may be read into memory and manipulated with DCON.COM.

FORMAT

Z80MR uses the standard Z80 instruction set as defined by Zilog, Inc. Z80MR is a field-oriented assembler. A field can be anywhere within a line of code. This assembler recognizes four fields in an assembly language source line:

```
label       operation   operand           comment
```

The assembler knows when it has reached the end of a field when it sees a field delimiter. This can be a space or a tab; it is a good habit to always use tab characters as delimiters.

Most assemblers require that any undefined labels be terminated in a colon, but this assembler does not. For example,

```
START JP FINISH
HELLO:    LD HL,1234H
```

are both valid. However, labels must appear in the label field. For example,

```
      START EQU   100H
START:      JP    FINISH
```

will cause an error. The EQU must be in the operation field and the label in the label field.

This assembler only examines the first six characters of any label or symbol. If the following labels were used in the same program,

```
FINISH1     EQU   1000H
FINISH2     EQU   2000H
```

a 'D' (duplicate symbols) error would be generated.

The operation field follows the label field and may either contain a Z80 op code mnemonic, an assembler directive (or pseudo-op), or a macro call.

Comments are not limited to the comment field and can actually be the entire line. This assembler recognizes the semicolon as the beginning of a comment and ignores the rest of the line.

The assembler will accept numbers in HEXADECIMAL (base 16), BINARY (base 2) or DECIMAL. Hex numbers must end with an H and binary numbers must end in a B. Decimal numbers should have no suffix letter. When a HEX digit begins with a letter, the letter should be preceded with a 0. Several examples are shown below:

```
LD    A,0F3H            ;hex format
OR    01001000B         ;binary format
LD    HL,4000H+28       ;combination hex/decimal format
```

SYNTAX

The assembler translates Z80 mnemonics into object code. It also recognizes certain commands and directives that the programmer can use to manipulate the assembler's output. These are referred to as 'pseudo-ops' and must be in upper case.

Psuedo-Ops

ORG <expr>

Sets the origin of the code or section of code. <expr> could be a number of previously-defined symbol (e.g. ORG 0 or ORG START).

END <sym>

Determines the end of an assembly language program. <symb>, if present, describes the first executable instruction of the program.

DEFW wordlist

DW wordlist

Both of these have identical meanings. In DEFW wordlist assembly language programs, 8-bit values are called bytes and 16-bit values are called words. Addresses are assembled with the most significant byte (MSB) following the least significant byte (LSB), since this is how the microprocessor handles these values. If more than one word is to follow a DW, the words should be separated by commas.

DDB wordlist

This pseudo-op is a way of assembling 16-bit values with the MSB first (opposite of DW).

DB bytelist

DEFB bytelist

DEFM bytelist

These pseudo-ops have identical meanings. The bytelist can be one byte or multiple bytes separated with commas and can include any mix of symbols, ascii characters in quotes, or numbers on the same line.

DS n

Reserve data space (n bytes). This is used to DEFS n position, allocate or label data storage space in a program. n is a number describing the number of bytes reserved.

label EQU <expr>

The EQU sets the label equal to the expression. Note that the label should not be terminated with a colon when used with an EQU pseudo-op. The label can be any symbol (byte or word) and the <expr> any number. A label defined with an EQU cannot be redefined later in the program.

label DEFL <expr>

This assigns the value of the <expr> to the label like the EQU pseudo-op, but a label defined with a DEFL can be redefined later in the program.

*INCLUDE <filename>

This pseudo-op causes the assembler to stop assembling lines in the file it is presently in and read in the file <filename>. It then begins assembling lines in this included file until it reaches the end of the file, then returns to the original file and resumes assembling. The <filename> can be any file, although if the extent is left off, it looks for the given filename with an extent of .LIB. The asterisk must appear in column 1 with the word INCLUDE immediately following, with no embedded spaces.

Conditional Pseudo-Ops

IF <expr>

Conditional assembly is a way of writing a single ELSE program so that it can be assembled different ways ENDIF or with different options by only changing a couple lines of code. When the assembler encounters an IF pseudo-op, it evaluates the symbol <expr>. If <expr> is non-zero, it assembles the following lines until it reaches an ELSE or an ENDIF. If <expr> is 0, the lines are ignored until the assembler encounters an ELSE or an ENDIF. When the ELSE or ENDIF is encountered, the assembler resumes operation. Every IF requires an ENDIF.

Operators

Operators allow the programmer to make the assembler do arithmetic and logical operations. They are usually used to manipulate operands or generate symbols. There should be no embedded spaces when using these operators, since the first blank encountered terminates the operand field. The operands may be symbols or numbers in binary, hex, or decimal.

```
Arithmetic
      +           arithmetic addition.
      -           arithmetic subtraction
      *           arithmetic multiplication
      /           arithmetic division (truncating the result)

Logical (Bit_Manipulation)
      & or .AND.  logical AND operation
      ^ or .OR.   logical OR operation
      .XOR.       logical exclusive OR operation
      \ or .NOT.  logical inversion
      .SHR.       shift left operand to right by right operand
      .SHL.       shift left operand to left by right operand
      .HIGH.      byte value is assigned the high byte of a 16-bit value
      .LOW.       byte value is assigned the low byte of a 16-bit value
```

```
Conditional (return TRUE or FALSE to IF )
      = or .EQU.  logical equivalent
      > or .GT.   greater than
      .UGT.       unsigned greater than
      < or .LT.   less than
      .ULT.       unsigned less than
```

Listing Psuedo-Ops

These options only affect the print file (.PRN). The pseudo-ops beginning with an asterisk must begin in column 1.

*EJECT or EJEC
: The next line of the listing should be placed at the top of the next page.

*HEADING
: Place the text (following this command) on the top of each page.

TITLE 'text'
: Place the text in the quotation marks (either double or single) on the top of each page in the listing file.

SPAC n
: Leave n blank lines in the listing.

*LIST ON
*LIST OFF
: Turn the listing on or off. This is usually used to omit long comments or certain sections from the .PRN file.

*MACLIST ON
*MACLIST OFF
: Turn the expansion of macros on or off. Seeing how the macros are being expanded is handy for optimizing code.

LIST options
NLIST options
: These pseudo-ops allow you to turn any of the supported listing file options on (LIST) or off (NLIST) without changing the other options. Both of these pseudo-ops must be followed with one or more of the following option letters. If these pseudo-ops are used, some options (marked with [ON] in the following list) default to on.

A	List all bytes in DB, DW, DDB, etc. Otherwise, only the bytes that can fit in one line are included in the listing.
B	Place symbol table into object file.
G	Place system generated symbols into object file.
I	List lines of conditional code following a false conditional. If off, only the code actually assembled is listed. [ON]
M	Expand macros in listing files. [ON]
O	Produce an object module, showing the bytes being generated by the assembler; otherwise just the source and (optionally) macro expansions. [ON]
R	Use absolute displacement for JR and DJNZ.
S	List source code in listing file. [ON]

T List symbol table in listing file. [ON]
X Generate and list cross references in listing file.
Z Generate an error for Z80-only opcodes. Allows you to write in Z80 mnemonics for an 8080 processor.

ERROR REPORTING

When the assembler is unable to understand what you are instructing it to do, it generates an error message. The console will display one of the error codes shown below, and the line on which the error was found. This will also be noted in the listing file.

D Duplicate symbol definition. You will see this error message if you use the same symbol twice within the six character limit, independent of upper or lower case.
E Relocation error. This occurs if the assembler cannot reassign an address as expected.
F Format error. You will see this if you break any of the rules regarding field use and macro format.
K Keyword error. This means you tried to use one of the assembler's reserved words or pseudo-ops as a symbol.
L Label error. The attempt to assign a value to a label was unsuccessful.
M Missing label. The symbol you are using was never defined.
N Macro nesting error. Macros can be nested (that is, a macro can call another macro), but if the nesting gets too deep the assembler will quit, and give this error. You can only call macros that were previously defined.
O Op code error. If you see this, look in the operation and operand fields and consult the mnemonic table.
P Phase error. A two-pass assembler builds a symbol table on the first pass and generates the object code on the second pass. If a number that it calculates for a symbol on the first pass does not agree with a number it generates in the second pass, this error is shown.
Q Questionable operand. Actually there's no question about it, it is a bad operand.
S Syntax error. You broke one of the syntax rules described above.
T Symbol table full. Not much you can do with this except pare down the code.
U Undefined symbol. You used a symbol but forgot to define it.
V Value error. Usually means you are trying to do an 8-bit operation with a 16-bit number.

MACROS

Macros are a way of writing subroutines in assembly language and then calling the subroutine by entering the 'macro name' into the source. The macro may be called as many times as necessary, anywhere in the program. When the assembler is operated, the lines of source code that make up the macro will be inserted into the file by the assembler. Note that using a macro does not reduce the size of the object code that is produced, since all the lines of code that make up the macro definition are assembled into the object file.

Macros have a form that is unique, and must be followed closely for correct results. The general form of a macro is

```
name  MACRO #parameter1,#parameter2,....
      instruction
      instruction
      instruction
      .
      .
      .
      ENDM
```

The name is the symbol that will be used to invoke the macro. MACRO is a keyword that will indicate to the assembler that a macro is being defined. The parameters always must begin with a '#' sign, and must be separated by commas. The instruction can be a Z80 instruction, or any of the assembler commands listed above, including conditionals. The instruction can also be another macro call (a nested macro), but only if the nested macro has previously been defined. The ENDM keyword tells the assembler that it has reached the end of the code for the macro that was called. Do not use a colon behind the macro name.

Another keyword unique to macros is LOCAL. This makes the assembler generate its own unique label every time the macro is expanded in a program. Following the word LOCAL (which must be on the second line of the macro), are the symbols which require unique labels. These symbols must also be preceded with a '#' sign. An example of LOCAL is included below.

```
AJUMP MACRO
      LOCAL #ADR_Z,#BACK
      OR    A
      JR    Z,#ADR_Z
      LD    A,40H
      JR    #BACK
#ADR_Z:     LD    A,04H
#BACK:      LD    DE,0
      ENDM
```

Each call to AJUMP will generate unique labels for ADR_Z and BACK.