88/HD
User's Manual

PolyMorphic
Systems

460 Ward Drive Santa Barbara California 93111 (805) 967-2351

This manual is PolyMorphic Systems part number 81Ø198.
Copyright (C) 1981, Interactive Products Corporation. This manual was written by Frank Anderson and Brian Smith.

Section 1

1.Ø Introduction

The System 88/HD is PolyMorphic Systems' Winchester Technology disk system. The disk drive and intelligent controller are manufactured by PRIAM. The host interface S-1ØØ adapter card is manufactured by PolyMorphic Systems. The host interface card consists of data latches and address decoding circuitry to provide communication to the intelligent controller mounted on the disk drive. See the PRIAM documentation for information about the drive and controller.

1.1 Installation

If you have purchased an entire system with hard disk, skip to the section on cable connections.

1.1.1 Host Adapter Card

Turn power off on 8813, remove cover and card hold-downs, install host adapter card in an unused slot of the backplane. Connect cable from card to the empty slot labeled AUX1 in 8813 back panel and using supplied hardware, mount the connector in 8813 back panel. If there is a System 88 8" floppy disk controller included in your upgrade see the 88 MS manual for card installation. Re-install the 8813 hold-downs and cover.

1.1.2 Cable Connections

Connect the supplied 25 conductor cable from AUX1 to the 25 pin connector on the 88/HD. Connect the AUX DISK cable to the 37 pin connector on the hard disk chassis. If you already have an MS, there should be a new cable for the floppy in the hard disk chassis and your MS. Remove the old cable from the 8813 to the MS and connect the new cable from the AUX DISK connector to the hard disk chassis to your old MS. The MS chassis with drives 4 and 5 should be at the end of the cable for proper termination. The new floppy is device # 6. The hard disk interface connector is the opposite polarity of the keyboard connectors on the host system, so connect the hard disk end first and you can't go wrong. These cables will only go on one way so if you can get them to stay they're right.

1.1.3 Power Connections

DO NOT PLUG THE HARD DISK INTO THE UTILITY OUTLETS ON THE BACK OF THE HOST SYSTEM UNLESS THEY ARE RATED GREATER THEN 4 AMPS.

Plug the female end of the power cable into the side of the black box on the back of the 88/HD chassis, make sure the power switch is turned off, and plug the other end into a standard grounded wall outlet. Turn power on.

Section 2

2.Ø Volume Manager

This is a description of what the volume manager is, how it works, and how the system communicates with it. The volume manager allows the System 88 to better allocate its non-volatile storage resources (e.g. floppy disks, hard disks, magnetic tape, bubble memory).

2.1 Definitions

The **volume manager** is a system of flexible mapping of logical devices (units) to physical devices or portions of physical devices. The volume manager deals with units, volumes and devices, so some definitions are in order for these new terms.

A **unit** is the logical equivalent of a drive. A unit is the first thing specified in a pathname and anything in the System 88 that used to refer to a drive, with the volume manager, now refers to a unit.

A **device** is a physical mechanism used for storage of data. There are currently 4 types of devices supported on the System 88. They are 5" single sided single density floppy disks, 5" double sided double density floppy disks, 8" floppy disks, and hard disks. There is a table below listing all the devices on the System 88. Devices are independent of units and any device may be connected to any unit.

A **volume** is a logical division of a device. In the case of a floppy, a volume is defined by the storage capacity of the medium in the device. Floppy devices may have only one volume and the volume is the entire device. Volumes on a hard disk are physical divisions of the device. Volumes may be any size from 1ØØ sectors to 65536 sectors and up to 19 volumes may be defined on one device. Each volume on a configurable device is named.

2.2 Theory of Operation

The volume manager connects volumes to units. For example, connecting device 5 to unit 3 will cause all references of the system to drive 3 to go to device 5. When you type:

```
LIST 3
```

the activity led on drive five will come on and the directory on the diskette in drive five will be listed on the screen.

You can think of the file specifier (e.g. <4<HELLO) as a logical "unit" number followed by a path name rather than a physical drive number followed by a path name. The physical drive "4" may correspond to the logical unit <4<, but does not necessarily have to.

Physical devices may contain more than one volume. For example, we may decide to set up a hard disk so that different areas on it appear to the system as units <6<, <7<, and <8<.

In the system 88, the volume manager resides as a filter on the input of the DIO function. Instead of having the system communicate directly to the physical device drivers, it communicates with the volume manager. The volume manager then translates the parameters and calls the actual device drivers.

The volume manager is actually five programs. The first program is the filter on the input of DIO which we just discussed. The second program is the "Vmgr" overlay which allows you to connect and disconnect both units and physical devices to the system. The third program is called "CONFIGURE". It allows a physical device to be broken up into logical volumes of any size, and then encodes that information onto the device. The fourth program is called "V-SETUP" and is used to configure the default setup of your system. The last program is called "CONNECT" and is used to change the configuration of the units from the keyboard.

2.3 Resident Code

The volume manager is essentially a table driven device. When the system (with the volume manager connected) is passed a request for DIO, the unit control block for the unit specified is indexed. Information from the block is used to translate the parameters passed to DIO to the

parameters which the device drivers need. There is a unit control block for each of the nine (seven on single user) units which the system can access. This resident code is contained in the file Driver.DD on the single user system. On the TwinSystem it is built into the Boot.2U file.

2.4 Vmgr Overlay

The function of the "Vmgr" overlay is to set up the tables in the resident portion of the code correctly. There are five function codes which can be passed to the overlay. They are:

Code	Function
ØH	Initialize I/O drivers and UCB tables
1ØH	Return a pointer to the specified UCB in HL
11H	Connect a volume to a logical unit.
12H	Get device definition block
13H	Disconnect a volume from a logical unit.

Commands are passed to the overlay in the A register.

2.4.1 Initialize Function

The initialize function (command Ø) is called by the System 88 executive when the system is reset. The overlay must first initialize the device driver(s) and then the default hookups for the unit control table. The initialization of the device drivers is done differently for single and twin systems.

2.4.1.1 Single User Driver Initialization

The first thing the overlay does is look for a file on the disk called "Driver.DD". This file contains the hard disk driver and the volume manager filter. If the file does not exist, "Vmgr" will return with an error code of Ø1D5H (No device driver). No other initialization will be done in this case.

If the Driver.DD exists it will be loaded just below the top of RAM and executed. This will initialize the device driver itself. As part of this initialization, the driver code also changes MEMTOP to reflect the loss of the memory used by the resident portion of the volume manager.

After initialization on the single user system, Exec will display it's sign on message:

Exec xx (xx/xx/xx) - Top of RAM is nnFF

nn above will be 600H bytes less than it would be if the volume manager were not present. After the device initialization is performed the program jumps to the unit control block initialization code.

2.4.1.2 Twin System Driver Initialization

In the twin system initialization, the overlay also looks for a file called "Driver.DD". THIS IS NOT THE SAME AS A SINGLE USER Driver.DD FILE. There is no Driver.DD normally on the TwinSystem disk. It is reserved for use for other device drivers for tape cartridge or other possible future devices. When the driver code returns, the routine will jump to the unit control block initialization code.

If no "Driver.DD" file exists, device initialization will be skipped and it will jump directly to the unit control block initialization code. No error code will be returned in this case.

2.4.1.3 Unit Control Block Initialization

The last thing the initialization function does is set up the unit control blocks with their initial values. It does this by repetitively calling the connect function with the default values that the user has set up for his nine (or seven on single user) units. Any errors encountered during these hookups will be reported back to the Exec.

2.4.2 Return a Pointer Function

The return a pointer function (command 10H) takes the unit number which is passed in register C and points HL to that unit control block. The unit control block contents are defined in the System Programmer's Guide.

2.4.3 Connect Function

The connect function (command 11H) can be invoked in two ways. It can either be called to hook up a named volume, or it can be called to hook up the next available volume on a physical device. To invoke the command the overlay is called with the unit number in C, and the

physical device number in B. Optionally, a string of 8 characters may be pointed to by HL when hooking up a named device. If HL = Ø (no name pointed to) then it will be assumed that the user wants to connect the next available volume on that physical device. HL must contain Ø if no name is specified to be connected.

There are five possible errors that the "Vmgr" can return when trying to hook up a device. They are:

ØlDØH: That unit is already connected.
ØlDlH: That volume is already connected.
ØlD2H: I can't find that volume. (for named volumes)
ØlD3H: No volumes available. (for unnamed volumes)
ØlD4H: That device has no volumes defined.
ØlD5H: No device driver.

The volume manager will not connect a new volume while an old volume is connected to the requested unit number. In addition, it will not duplicate any volumes. For example, units <4< and <5< would not be allowed to point at the same spot on one physical device. If the volume manager allowed this the system interlocks would not work, and files would be scrambled.

In order to allow different volume sizes on a physical device and not foul up the pointers to different volumes on that device, each device with multi-volume capability has information regarding its configuration encoded on the device. This information is stored on the last physical sector of the device. This sector is not accessible via system calls to DIO. The volume manager uses this information when connecting a volume to assure that the pointers to different areas of the physical device remain secure.

2.4.4 Read Device Definition Block Function

The read device definition block command (12H) makes this information available to the user. The register setup for this command is: B contains the physical device number of the device being interogated; HL points to the memory area where the device definition block is to be placed. The format of the device definition block is as follows:

# Bytes	Value	Definition
1	1-19	Number of volumes contained on this device.

* The following is a sample entry. This entry is repeated for each volume contained here.

2	100-65535	Volume size
3	0-(dsize-100)	Offset from physical 00
8	ASCII chars.	Volume name

2.4.5 Disconnect Function

The disconnect unit (13H) function is passed the unit number in C. Before the unit is disconnected, it checks to see if the unit is busy. This will assure that one user does not disconnect a unit while the other user is using it. The only error that this function can return is 01C2H: That device is busy.

2.5 CONFIGURE Program

> WARNING:
> The configure program will destroy all the data on the entire hard disk when it is run. It is provided to the user to enable him to configure the disk as necessary for operation. The configure program should not be available to the casual user because of the data destroying possibilites. The configure program tries to insure that the user is aware of the consequences of its use but it cannot prevent malicious use. The configure program should be deleted from the disks that are in everyday use and only be on disks that are kept in a physically secure place.

The purpose of the device configuration program is to set up the device definition block on a physical device. The device definition block was discussed in the previous section. When the program is run it first comes up and warns the user that it is about to zero the data on the entire device. It then asks him for a device number. The

selected device is then formated with error mapping enabled.

After the initialization is complete the program begins asking the user for volume names and sizes. This process is terminated in one of three ways. It will terminate when there is no more room for a volume, when the maximum number of volumes (19) has been reached, or when the user answers the size question with a null (just hits return). At that point the program will construct the device definition block and write it to the last physical sector of the device.

2.6 V-SETUP Program

The V-SETUP program sets up the default definitions for unit connection which the "Vmgr" overlay uses on initialization.

> NOTE: This program modifies the "Vmgr" overlay to accomplish its task. This means that each system disk which is used on the system can have a different set of default values for unit connection. This should not be confused with the CONFIGURE program which permanently sets up a physical device. Care should be taken that you make all system disks with units defaulted compatable with the physical device configuration which the program will be connected to.

When the program is invoked it will ask you if you want to define a new block or modify an old one. If you are modifying an old one it will ask from which disk to read it, and then display the default setup and ask which volume to change. If you are defining a new block, the program will ask you, for each unit number 1-9, what physical device you want connected to that unit. If that device has more than one volume on it, the program will then display a list of the volume names for your selection. You must select the volume by name.

After you have made your selections for all the unit mapping the program will ask you which unit you wish your system to "boot" from. When the program has all this information it will re-write the last sector of the "Vmgr" overlay back out to the disk with the new information in it. This works in much the same way as the printer driver's "Setup" works.

2.7 CONNECT Program

The connect program is used to change units from the

keyboard. For instance, if there are more than nine volumes defined in the system, you may want to change which nine you have connected from time to time without having to change the default setup and reboot. This is done by using the CONNECT program to disconnect the volumes not needed and connect others in their place. CONNECT accomplishes this by calling the Vmgr overlay with the connect and disconnect functions. The format of the CONNECT command is:

CONNECT {*}unit,device,name

The asterisk is optional (indicated by {}) and indicates to the program that you want the current volume on the unit specified disconnected. If the asterisk is not used and the unit specified is already connected, an error (That unit already connected.) will be reported. The unit is a number (1-9 in Twin, 1-7 in single) indicating the unit to be connected. The device is the physical device number to be connected. The device numbers are as follows:

Device #	Device
Ø	No connection
1	*
2	*
3	*
4	8" #1
5	8" #2
6	8" #3
7	8" #4
8	**
9	**
1Ø	**
11	HARD DISK #1
12	HARD DISK #2
13	HARD DISK #3
14	HARD DISK #4
15- 2Ø	UNUSED

* These drives are the same type as previously existed in your system. They are determined by the ROMS. If your ROMS are Part #'s Ø35113, Ø35114 and Ø35115 these drives are 5" Double Sided Double Density. If your ROMS are Part #'s Ø35116, Ø35117 and Ø35118, these drives are 5" Single Sided Single Density.

** These drives are the other five inch type, but no driver is provided.

The name is the name of the volume to be connected. If the device cannot have named volumes (a floppy), then the name is ignored. If device Ø is specified then the unit specified will be disconnected.

2.8 VLIST Program

The VLIST program displays the units in the system and the physical devices and volume names to which they are connected. VLIST is invoked by typing:

```
VLIST
```

The screen display will then be as follows.

```
Vlist/1.Ø (Ø2/23/81)

Unit Device Size  Name
  1    1     Ø578
  2    2     Ø578
  3    3     Ø578
  4    4     134Ø
  5    5     134Ø
  6    6     ØØØØ
  7    7     ØØØØ
  8    11    4356  first
  9    11    3419  second
```

The size displayed by VLIST is taken from the UCB if it is present (hard disk only), otherwise the Dsize function of Dio is called to determine the size of the device connected. Double sided 8" floppies will return the correct size for the diskette currently inserted if a diskette is present. If no diskette is inserted the size will be listed as if they were single sided since the diskette must be inserted to create the hardware indication that the drive is double sided.

Section 3

3.Ø First Time Through

The standard system disk as shipped by PolyMorphic Systems has the necessary software to control the 88/HD but is not defaulted to automatically use the hard disk. The hard disk is configured at the factory with as many 16 megabyte volumes as will fit on the disk ordered. The last volume is defined as the remainder of the disk. For the smaller drives this means that they are configured as one drive. The volumes are named first, second, etc (not very original, but easy to remember). To connect the hard disk, turn on the system and after the Exec '$' prompt, type:

```
CONNECT *7,11,first
```

This tells the system to connect as unit 7, device 11 (the first hard disk), the volume named 'first'. The asterisk '*' tells the system to disconnect the current drive 7 before connecting the new one. Without the asterisk, the system would report 'That unit already connected.'(See the section on the CONNECT program.) Now the hard disk has effectively become drive 7 in the system and can be treated as any other drive.

Section 4

4.0 BACKUP Program

Provided on the system disk is a program called BACKUP. In the unlikely event of a disaster, (Someone tests their new crowbar on the hard disk.) it would be convienent if all the data on the hard disk were also somewhere else. With the System 88 that somewhere else is floppy disks. The BACKUP program provides the ability to put everything new on the hard disk onto floppies for safekeeping.

4.1 Theory of Operation

Each time the System 88 creates a new file on the disk or modifies an old one, it sets a flag in the directory that indicates that the file has been changed (See section 4.3). The BACKUP program looks at these flags and copies all the files it finds with the flag indicating they are 'NEW'. The flag is called the 'NEW' bit (See system programmer's guide). The BACKUP program looks in the directory specified by the user and in all directories that it finds in that directory or its subdirectories. Thus by specifying a unit number only, the entire unit will be backed up because the main directory and all subdirectories will be checked. The BACKUP program resets the 'NEW' bit after it copies the file so that the file will not be backed up again unless it is changed.

4.2 How to Use BACKUP

BACKUP is invoked as follows:

```
BACKUP dirspec1 dirspec2 {*}
```

Dirspec1 is the directory in which to start searching for 'NEW' bits (source) and dirspec2 is the directory (which need not exist yet) to which to copy the files (destination). The directory structure will remain the same on the new volume. For instance, if you specify

```
BACKUP 9 6
```

and on unit 9 there is a sub-directory CLIENTS with new files on it, those files will be copied into the

subdirectory CLIENTS on unit 6. If the subdirectory CLIENTS does not exist on unit 6 it will be created.

The optional '*' indicates to the program that if a file on the destination disk already exists you want it replaced with the new file. If the new file is exactly the same size as the old one, BACKUP will then copy the new one directly over the old one in the same spot on the disk. If the file is not the same size, it will be deleted and a new file written on the destination disk. If the '*' is not specified, the program will ask what to do if it finds a duplicate file already exists.

The BACKUP program resets the 'NEW' bit only after the file has successfully been copied. Therefore, if there are any errors in the backup process, it can be restarted and it will automatically pick up where it left off. On an error that indicates that the disk is full the BACKUP program will tell you that the disk is full and then wait for the destination disk to be changed, and continue.

4.3 SETNEW and CLEARNEW

In order to preserve the flexibility of the TwinSystem, it was not possible to have the 'NEW' bit set when an INOUT file was modified in BASIC. In order to make a modified INOUT file eligible for backup, the NEW bit must be set using SETNEW. The form of the command is:

```
SETNEW filename
```

Once this is done BACKUP will then automatically backup the file the next time BACKUP is run. CLEARNEW is provided as a convenience to reset the 'NEW' bit without backing up a file. Once CLEARNEW is run on a file, that file will not be backed up automatically by BACKUP.

4.4 SetSys

It should be noted that the SetSys command (see System Programmer's Guide) will clear the 'NEW' bit on all files as it sets the 'SYSTEM' bit. Therefore, make sure that any files to be backed up have been backed up before you SetSys.