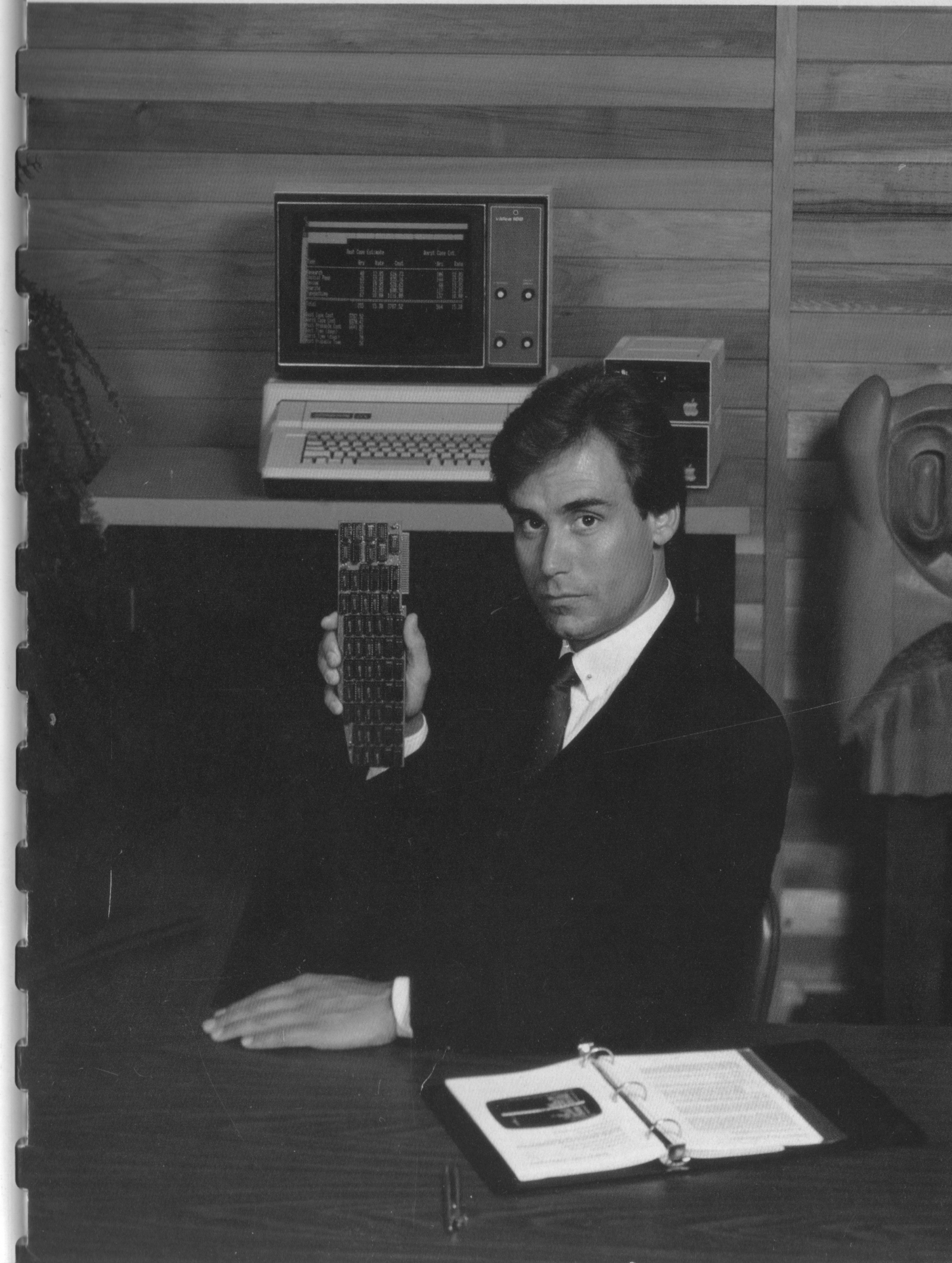
flashcard T.M.

OWNER'S MANUAL
FOR MODEL 2201 (147K)
AND MODEL 2202 (294K)



Flashcard Solid State Disk Emulator

Users' Manual

Synetix, Inc.
15050 N.E. 95th Street
Redmond, Wa 98052

For product support, contact:

Synetix, Inc.
15050 N.E. 95th Street
Redmond, Wa 98052
Attn: Flashcard Product Support

(206) 828-4884
(800) 426-7412

TABLE OF CONTENTS

1. Introduction

Congratulations on buying the Synetix Solid State Disk. This card is the finest of its sort on the market today, and will speed up your applications and/or program development so much that we feel you will agree with its nickname "The Flashcard."

Even though you are probably anxious to install your new purchase, we encourage you to read through this manual carefully, even if you are an experienced Apple owner.

We will be providing information and recommendations throughout the manual that will enhance your use of The Flashcard, and enable you to use it productively right away.

Some of the information contained in this manual is not obvious, even to the most experienced Apple owner, as we have found out from the questions of current owners. Please be sure to read as much of the manual as pertains to your application.

1.1. Scope of This Manual

This manual will present complete information on the Flashcard, including a general overview, installation instructions, applications programming (Applesoft, Pascal, CP/M) instructions, and advanced programming (machine language) protocols. Appendices at the end of this document are contain application notes. These notes will cover some useful hints on using the Flashcard with various application (purchased software) programs. As revisions or updates to these notes become available, they will be provided to owners of the Flashcard along with the new software at $10.00 per update.

1.2. Conventions Used in This Manual

Some computer jargon will be used in various places in this manual. The first time such a word is used, it will be underlined and defined. Please pay particular attention to the way these terms are defined.

There are currently three major revisions of the Apple // computer:

o The Apple][(or Integer Apple)
o The Apple][-Plus
o The Apple //e

The Flashcard functions identically in all three machines. For the purposes of this manual, the notation Apple // or simply Apple will be used to denote the family of Apple // computers, and one of the above three notations will be used when referring to any specific computer.

The Flashcard comes in two models: model 2201, which is capable of storing 147K bytes of data, and model 2202, which is capable of storing 294K bytes of data. Model 2201 will be referred to as a single Flashcard, and model 2202 as a dual Flashcard.

1.3. What Is the Flashcard?

The Flashcard, or Solid State Disk, or is a peripheral card for the Apple][, Apple][-Plus, and Apple //e. It fits inside the Apple // in one of the peripheral expansion slots (commonly called slots).

The section of this manual on installation covers which slot you should put your Flashcard in.

Floppy disk drives are the normal mechanical devices used for mass storage of data on the Apple //. They allow reasonably speedy transfer of data from the Apple's memory to magnetic media (the floppy disk). This data remains on the floppy disk until removed, much as music remains on a magnetic tape until erased.

The Flashcard is popularly known as a disk emulator. That is, the Flashcard emulates, or looks like, a floppy disk drive to the programs running on the Apple //, except that in place of the magnetic media, the Flashcard uses random access memory (RAM) to store the data. Data may be erased from RAM either by specifically removing it, changing it, or by turning off the power to the computer. You may use all the commands you normally use to manipulate data on disk to do the same with the data in the memory on the Flashcard. The only difference is that the data on the Flashcard is a working, rather than a permanent copy.

There are several problems associated with the use of normal floppy disk drives that make disk emulation desirable:

- Slow Data transfer rate (the speed at which data can be placed on or retrieved from the floppy disk). Large quantities of data take quite some time to transfer either to or from a floppy disk because of the mechanism involved in getting to the data.
- Insufficient data storage on a disk. The maximum amount of data that can be written to a typical Apple // floppy diskette is approximately 143,000 characters (more often referred to as 143K bytes). These floppies fill up quickly.

The Flashcard was designed to alleviate, or at least temper, these problems by using RAM in place of the mechanical parts and magnetic media of the conventional floppy disk drive.

This approach combines high data transfer rate with the improved reliability of RAM to circumvent many of the problems with the mechanical devices.

With a single Flashcard, you may effectively add the capacity of an additional disk drive, and with a dual Flashcard, two additional drives. Under several certain environments, a dual Flashcard can be formatted to look like a double sized floppy, giving you a great deal of room on a single volume.

2. Installation

Proper installation of the Flashcard is of paramount importance. Please follow these directions carefully.

1. Turn off **all** power to your Apple //, and all peripherals (printers, etc.). Failure to shut off the power before attempting to install the Flashcard could result in damage to the Flashcard and your Apple.

2. Clear everything off of the top of your Apple //, and remove the lid by lifting up at the back, then sliding the lid away from you.

3. Put the lid aside.

Near the rear of the computer you will see eight expansion slots (seven if you have an Apple //e). These slots are numbered from left to right starting with 0 (1, if you have an Apple //e). You probably have several of these filled already, so you must determine which slot to use for the Flashcard. Although the Flashcard will work in any slot, our recommendations are as follows:

If installing one Flashcard, use slot 6 for the floppy disk controller, and slot 5 for the Flashcard.

If installing two Flashcards, use slot 6 for the floppy disk controller, slot 5 for the first Flashcard, and slot 4 for the second Flashcard. If one of the Flashcards is a single, and the other a dual, place the dual Flashcard in slot 5.

The reason for these particular recommendations is for compatibility with Pascal and CP/M, both of which expect Flashcards to be located in slots 4 and 5. For more detailed information on the slot dependencies of any of the operating systems, please refer to the section pertaining to the operating system in question.

4. Place your hand on the Apple's power supply (located on the left of the computer) for a moment to discharge any static electricity you might have accumulated.

5. Unwrap the Flashcard from its protective wrapping and place it in the appropriate slot in your Apple (see above for slot determination). Rock the Flashcard gently until it is completely settled in the slot. Be absolutely certain that the Flashcard is firmly seated in the slot, but do not force it or treat it roughly.

Note: It is impossible to insert the Flashcard backwards.

If you are installing more than one Flashcard, repeat step 5 for the additional Flashcards.

6. Replace the lid on the Apple.

3. Disk Emulation Software

The Apple // normally uses <u>DOS</u> (Disk Operating System) to control the floppy disk drives. You probably received a DOS 3.3 System Master diskette, either when you bought your disk drive, or for people who have older Apples, when you upgraded to release 3.3 of DOS.

There are, however, several other principal operating systems that can be used on the Apple // to control the disk drives:

Apple Pascal
CP/M (if you have a Z-80 card)

For a complete list of operating and applications software for the Flashcard, please see the Flashcard Products List.

The presence of more than one operating system complicates the question of disk emulation software. Each operating system requires its own disk emulation software (or <u>drivers</u>).

Driver software is available for each of the operating systems, and as new operating systems are released for the Apple //, driver software will be made available, if possible.

The use of each set of drivers is discussed in detail in the next section of this manual.

4. DOS 3.3 Emulation Software

DOS 3.3 is the most popular operating system available on the Apple // today. It is the most current release of the very first disk operating system Apple Computer ever produced, and it still stands up well.

Although DOS 3.3 does have an elegant simplicity about it, there are certain dependencies built into DOS 3.3 that can make it less attractive than certain other operating systems. These dependencies all revolve around the concept of the mechanical disk drive.

Unlike Pascal and CP/M, DOS is "hard-wired" to believe it is running a 5 1/4" single sided/single density disk drive. Some assumptions have been made in the Apple DOS program which rely on the characteristics of the mechanical disk drive. These assumptions make it difficult to take full advantage of alternative mass storage devices such as the Flashcard without a few compromises. We have provided DOS 3.3 drivers that are as flexible as possible with regard to which of these compromises you have to endure.

The approach used in the DOS 3.3 drivers was to "fool" the DOS program into redirecting its data transfer to the Flashcard. None of the inherent weaknesses of DOS have been fixed, and no attempt has been made to speed up DOS for use on a high-speed device such as the Flashcard. The reason we chose this approach is two-fold:

1. Some programs rely heavily on DOS being absolutely intact with no modifications whatsoever (they use locations in the DOS program directly).

2. Several companies have produced relatively low priced DOS enhancement software that is compatible with the Flashcard. Providing DOS enhancement software in the drivers would be somewhat redundant, and would limit the the user's choice as to the method and extent of DOS enhancement.

4.1. Where to Realize Speed Increase

On the subject of DOS enhancement, perhaps a brief word is in order on the areas most commonly improved in DOS.

DOS Operation	Comments
BLOAD, BSAVE	The speed of binary loads and saves is reasonably easy to improve through a variety of means, and can be increased dramatically. Unfortunately, most disk overhead is not binary file operation.

DOS Operation	Comments
LOAD, SAVE	LOADing and SAVEing Applesoft and Integer BASIC programs can be sped up by the same techniques used for binary files. While such a speedup is extremely desirable during program development, it loses a certain amount of impact in the finished application since most BASIC programs tend to be "single-load" applications.
READ, WRITE	Text file manipulation has long been the bane of the Applesoft programmer's existence. Since the speed of text file I/O is considerably slower than I/O to binary and program files, text file handling routines are ideal candidates for enhancement.
All Other Commands	The speed of the other commands remain unchanged under the most popular DOS enhancers, although some features may be added to the CATALOG command, and the INIT command is often removed.

Even without the DOS enhancements, the Flashcard will improve disk I/O speed by a variable factor ranging from 100 to 500 percent, (depending on the particular operation being performed) while adding the element of solid state reliability.

We have determined that it is possible to load high resolution graphics pictures from the Flashcard at a rate of up to 6.2 per second (using the Fastload^{tm} DOS enhancement program from Microseeds). This amounts to a bit better than 50,000 bytes per second.

4.2. DOS 3.3 Distribution Diskette

With your Flashcard, you will receive a diskette marked:

DOS 3.3 Flashcard Drivers

This disk contains all the programs necessary to enable the Flashcard to emulate a disk drive under DOS 3.3.

4.3. Purpose of the Software

The purpose of the DOS 3.3 software is to allow the Flashcard to emulate a 5 $^1/_4$ inch floppy disk drive (model 2201), or two 5 $^1/_4$ inch floppy disk drives (model 2202).

Note: More than one Flashcard may be used in a given Apple.

Since the Flashcard so closely emulates Apple DOS, you will have to treat it just the same way you would treat a brand new floppy diskette after installing the DOS 3.3 driver software. When you install the drivers, you have what amounts to an unformatted disk. You must either **INIT** the disk, or let the COPY program format it. The Flashcard supports volume assignments.

When you boot the DOS driver diskette, you will be allowed to install the drivers, or configure them from a menu. You should read the appropriate sections on the drivers, and configuration to understand the function of each.

4.4. Files on the Disk

The files contained on the distribution diskette are:

SSD DRIVER/STD	Standard DOS 3.3 drivers for the Flashcard
SSD DRIVER/ALT	Alternate DOS 3.3 drivers fo the Flashcard
CONFIGURE SSD	Program to install special configuration
COPY	Fast copy program
AUTOCOPY	Program to install drivers and do automatic copy

4.4.1. SSD DRIVER/STD

This program installs a driver in DOS 3.3, allowing that operating system to access the Synetix Flashcard (models 2201 and 2202) as a floppy disk.

To use this program, simply boot a normal DOS 3.3 diskette, insert your DOS 3.3 driver diskette, and type:

BRUN SSD DRIVER/STD <RETURN>

(you do not type the <RETURN>, you press the key marked RETURN on your Apple)

This set of drivers may be run from within a program to allow automatic installation, and create "turnkey" applications.

Technical Information

The load address of the /STD driver is $4000 (16384). The program first examines each slot, looking for Flashcards. It builds a table containing a list of slots which do not contain Flashcards, then moves that table (as well as a relocated version of the driver) into the highest DOS buffer (usually at $9AA6).

The current MAXFILES value is reduced by 1, and the default MAXFILES value is changed from 3 to 2.

The net effect is that although DOS is larger, HIMEM will remain the same; this allows most programs to work with this driver, with the exception of those who use all three file buffers (these programs should issue a MAXFILES 3 command).

This program also installs a JMP instruction at the start of the RWTS routine (at $BD00).

The INIT Command

The DOS INIT command will format a new disk, as documented in the DOS 3.3 manual (Apple Computer, Inc.). The image of DOS written to the newly formatted disk will **not** contain the Flashcard driver routines, but will have a MAXFILES default of 2, rather than the normal 3.

You must initialize the Flashcard before attempting to write data to it. This may be done using the DOS INIT command (see the DOS Manual for more information), or with the COPY program.

Compatibility

This driver should work with most BASIC programs, and most machine language programs which do not alter the RWTS routine, or move the DOS buffers downward in memory. The latter case includes such programs as GPLE, which installs itself between DOS and the file buffers. These programs will function correctly if they are run before the SSD DRIVER/STD program is run.

4.4.2. SSD DRIVER/ALT

SSD DRIVER/ALT is the alternate DOS 3.3 driver for the Flashcard. Like SSD DRIVER/STD, SSD DRIVER/ALT installs a disk emulation driver in DOS 3.3.

It is intended for use with programs which require a standard size DOS ($9600-$BFFF with MAXFILES set to 3). After the /ALT driver is installed, DOS will not be able to format blank disks (see discussion under INIT Command, below).

To use this program, simply follow the instructions above for installing SSD DRIVER/STD, except substitute the file name SSD DRIVER/ALT. This program may be run from within an Applesoft or machine language program.

Technical Information

SSD DRIVER/ALT loads at $4000. Like the /STD program, the /ALT driver builds a table containing a list of slots containing a list of slots containing Flashcards. The table, and the driver program are moved into the space in memory usually occupied by the "16-sector disk format" routine (used by the INIT command to format blank disks). This means that DOS will not get larger (as with the /STD driver), and that the current MAXFILES value will not be altered.

The INIT Command

You will not be able to format blank disks with the INIT command once the /ALT driver is installed, but you will be able to create a new catalog, and write a copy of DOS on a pre-formatted disk. The DOS written will contain the /ALT driver.

As noted under the section on the /STD drivers, you must initialize the Flashcard before attempting to use it to store or retrieve data.

Compatibility

The /ALT program will not be compatible with any other program which tries to install itself in the area of memory that is normally reserved for DOS's disk formatter code (this includes such programs as FASTLOAD by Microseeds).

It may not work with other programs which patch the RWTS routine, such as ScreenWriter^{tm} and some hard disk operating systems.

Other than the above restrictions, the /ALT program should work with just about everything, since it does not affect the file buffers, or any other part of DOS.

4.4.3. CONFIGURE SSD

Both versions of the Flashcard driver (/STD and /ALT) examine each Apple slot to see which contain Flashcards. This examination has no harmful effect on the Flashcards (all data is preserved during the test), but while examining a slot containing a peripheral card other than a Flashcard, unexpected, and often unpleasant side effects have been known to occur (a co-processor card, for example, may decide to turn the Apple's processor off!).

To avoid this problem, the CONFIGURE SSD program may be used to specify which slots contain these types of cards.

To use this program, type:

RUN CONFIGURE SSD <RETURN>

You will then be asked which driver (/STD or /ALT) you wish to configure.

Press **S** to configure the /STD drivers, or **A** for the /ALT drivers.

The selected driver will then be loaded, and its configuration displayed. The default configuration looks like this:

```
SLOT 1 MAY CONTAIN A RAM DISK.

SLOT 2 MAY CONTAIN A RAM DISK.

SLOT 3 MAY CONTAIN A RAM DISK.

SLOT 4 MAY CONTAIN A RAM DISK.

SLOT 5 MAY CONTAIN A RAM DISK.

SLOT 6 MAY CONTAIN A RAM DISK.

SLOT 7 MAY CONTAIN A RAM DISK.
```

To lock out a slot, enter the slot number, and the word "MAY" will change to "CANNOT". If you select a slot in error, choosing the slot number again will change the "CANNOT" back to "MAY".

Continue with this process until the configuration is complete. When asked if the configuration is OK, press **Y** (for yes), and the driver program you chose will be updated. If the file is locked, or the disk is write-protected, this program will not work.

You will then be asked if you want to configure the other driver too. If you press Y, it will be configured automatically according to the specifications you have already entered.

4.4.4. COPY

As the name implies, this program is used to copy a disk, and will copy any DOS 3.3 disk (it will not copy CP/M, Pascal, or any protected disks). The COPY program can be called from an Applesoft program to create a "turnkey" system (see AUTOCOPY, below).

This program is extremely quick because it only copies the sectors in use, rather than the entire disk.

To run the COPY program, type:

BRUN COPY

Enter the slot and drive numbers of the original disk (the one from which to copy), then enter the slot and drive numbers of the copy disk (the drive to which you wish to copy).

You will be asked: ANY CHANGES?

Press **Y** if you have made an error in you specifications, otherwise, type **N**.

If you are using two drives, you will be prompted to insert both disks and press **RETURN**. Once you press the RETURN key, the copy will take place.

If you are using only one disk drive, you will periodically be prompted to switch disks.

This copy program is very handy for backing up floppies, as well as executing quick data copies to the Flashcard.

When the copy is complete, you will be asked if you want to make another copy of the same disk. Answer this by typing **Y** if you wish to make additional copies of the same disk, otherwise type **N**.

You will now be presented with the option of copying another disk. If you would like to copy another disk, you may do that by typing **Y**, otherwise type **N**.

Technical Information

The copy program loads at $C00, allowing a small Applesoft program (such as AUTOCOPY, below) to reside below it. The COPY program uses all memory from the end of the code up to HIMEM as a buffer, so the previous contents of that memory are destroyed when the program is run.

This program only copies those sectors marked as used in the disk's VTOC. Since Pascal and CP/M do not store their volume table of contents in the same format, COPY will not work correctly on such diskettes.

If you choose to repeat the copy procedure, and the disk was only partially full, the next copy may not require the program to re-read the original disk.

The following table will help you in advanced use of the COPY program:

Location	Contents
$C00-$C02	JMP instruction to the start of the copy routine
$C03-$C04	Pointer to the parameters (usually at $C05)
$C05	Slot from which to copy
$C06	Drive from which to copy
$C07	Slot to which to copy
$C08	Drive to which to copy
$C09	Contains $FF if copy should be automatic (no human intervention), otherwise $00
$C0A	Format flag (only used if $C09 contains $FF) $0 does not format a copy disk, $FF does format the copy disk.

4.4.5. AUTOCOPY

This program is used to load the SSD DRIVER/STD program and copy disks (usually to the Flashcard) automatically. It is easy to use, and can easily be modified for just about any application. As delivered, it will copy the disk in slot 6, drive 1 to slot 6, drive 2.

To use AUTOCOPY, type:

RUN AUTOCOPY

Customization

You will need to change the program to copy from and to different drives. There are only five variables that need to be changed. These variables are:

OS -- Original Slot (1..6)

OD -- Original Drive (1..2)

CS -- Copy Slot (1..6)

CD -- Copy Drive (1..2)

FMT -- Format flag. 0 = do not initialize copy diskette

1 = initialize copy diskette

After these variables have been initialized, **GOSUB 100** to actually perform the copy. When done, control will return to the statement following the **GOSUB 100.** The variables may then be changed, and another copy made.

Prompts may be added, instructing the operator to insert certain disks in the drives, but the program must be short. When all copies have been made, the AUTOCOPY program should then RUN another program.

To find the end of an Applesoft program, type:

PRINT PEEK(105) + 256 * PEEK(106)

If the value displayed is greater than 3040, the program is probably too large to fit under the COPY program in memory.

Hint: removal of REM statements will save program space.

Technical Notes

Since the copy program loads at $C00 (3072), the AUTOBOOT program must remain below that address.

4.4.6. DOS 3.3 - Flashcard and DiversiDOS^{tm}

Both the /STD and the /ALT drivers are compatible with Bill Basham's DiversiDOS (Note: you must use the first of the POKE's listed in the DiversiDOS manual to use the /ALT drivers).

The DiversiDOS disk includes several programs and utilities, some of which conflict with the drivers. These are described below:

DDMOVER

This program moves DiversiDOS into the upper 16K region of the Apple (the "language card" space), leaving more room free for user programs.

Using DDMOVER with the /STD Driver

There is no compatibility between these two programs.

Using DDMOVER with the /ALT Driver

You must install the /ALT drivers before BRUNing DDMOVER. As documented in the DiversiDOS manual, disks initialized after the DDMOVER program has been run will be data disks, and will not boot.

BUFFER

The BUFFER program installs a keyboard and printer buffer in DiversiDOS.

You may use the BUFFER program with the /STD drivers, and it will function correctly.

The BUFFER program is incompatible with the /ALT drivers.

5. Apple Pascal Disk Emulation Software

Apple Pascal provides a friendly environment for the development of large systems. One of the primary reasons the Pascal environment is so friendly is the capability of the operating system to retrieve code from disk, as needed for execution.

Unfortunately, this disk swapping capability implies considerable system overhead because of the reasonably slow speed of floppy disk drives. The Flashcard addresses this problem, and provides almost transparent disk accessing.

Note: Under Pascal, you will achieve a far greater increase in productivity than under DOS 3.3 because Pascal utilizes the block devices (disk drives) far more efficiently.

5.1. Installation of the Pascal Drivers

Under Pascal, device drivers are normally stored a special space reserved for them in the operating system called the BIOS (Basic Input/Output System). These drivers are installed in an extremely flexible manner by an Exec Processor. The exec processor allows you to configure the Flashcard, and the Pascal system according to your own individual preferences.

The following discussion assumes that you have used the Pascal Operating System, and have a certain amount of familiarity with the various functions being discussed. If you have not used the system before, or are unfamiliar with some of the terminology used, please take the time to read through the appropriate sections of Apple Pascal Language Reference Manual, and Apple Pascal Operating System Reference Manual (Apple Computer, Inc.).

5.2. Installation of the Flashcard for Pascal

Under Pascal, you **must** install your first Flashcard in slot 5 of your Apple (see the previous discussion for more information on installing your Flashcard). One additional Flashcard may be installed in slot 4, and will be recognized under Pascal. If you have a single Flashcard and a dual Flashcard, the single must be located in slot 4, and the dual in slot 5. The reason for these assignments is for compatibility with the normal Pascal disk drive slot assignments.

Once properly installed, your Flashcard may be configured to appear as any volume number in the range 4..5, 9..12 (normally the floppy disk drives are volume numbers 4, and 5). The following table illustrates the sizes of various configurations:

Slot 5	Slot 4	Volume Size
Single		288 blocks
Dual		576 blocks
Dual	Single	864 blocks
Dual	Dual	1152 blocks

5.3. The Exec Processor

The Exec Processor does more than installing the Flashcard drivers -- it is a configuration processor; that is it will configure the Pascal Operating System the way you want it automatically by executing a series of commands you supply. These commands are contained in a file called

PROFILE.TEXT

which is created in the normal Pascal Editor. Upon power-up, or cold-boot (RESET, or H(alt command), this file is executed. The purpose of the file is to do the following:

1. Initialize the Flashcard
2. Transfer some system files to the Flashcard
3. Instruct the operating system to look on the Flashcard for those system files.
4. Other user-definable operations (performed at your discretion).

If we look at this sequence carefully, it may be apparent that we would prefer to leave the Flashcard alone on a RESET or H(alt cold-boot so no data is lost. This involves not initializing the Flashcard every time the file PROFILE.TEXT is executed.

Fortunately, there is a very easy way to make this happen. All the commands to the Exec Processor have the following syntax:

[c]

where c is a command character, and the left and right square brackets ([and]) enclose the command. These command characters will be defined below.

Obviously, the first step is not to initialize the Flashcard, but rather to see whether the Flashcard has already been initialized (query the Flashcard). If this query finds out that the Flashcard has, indeed been initialized, a flag will be set. If the flag is set, only commands in upper case will be executed. If the flag is not set, all commands will be executed.

Two example EXEC's are included on the distribution disk to illustrate the use of this option.

Note: If you simply X(ecute EXEC.CODE, the Exec Processor is invoked, but rather than executing the commands in PROFILE.TEXT, it executes the commands in ACCESS.TEXT. In this way, you can have one exec for cold-starts, and another for general utilities.

Boot-Up Sequence

When you boot the Flashcard driver diskette, several things happen before control is returned to you:

- the file **SYSTEM.ATTACH** is executed, which loads in the drivers for whatever non-standard devices you may have configured.
- the file **SYSTEM.STARTUP** is executed, which starts the Exec Processor, and instructs it to perform the commands contained in **PROFILE.TEXT** (according to the upper case/lower case rule discussed above).
- the Exec Processor performs all the commands you have supplied it, then returns control to you in the familiar UCSD Pascal Operating System shell.

5.4. A Sample Exec

Let's look at a sample exec -- what the commands are, and what they do. We will assume, for this example that there is a dual Flashcard installed (in slot 5).

As you will recall from the above discussion, the first thing we want to do is to query the Flashcard to see whether it contains valid data. The exec command for this is:

[Q]SSD-200:
[Q]

The purpose of this command is to query the existence of the volume SSD-200:. If SSD-200: is found, the flag is set true. Now we execute the [Q] one more time to flip the flag. The reason this is done is as follows: On a power-up start, SSD-200: will not be found (no data on the Flashcard), so the flag will be false. Lower case commands execute only if the flag is true, so we flip the flag to make all commands execute if SSD-200: is not found. On a RESET or H(alt, SSD-200: will be found, and the flag will be true. We do not want to execute all the commands if SSD-200: is online, so we flip the flag to false to exclude lower case commands.

[i]128,576,9

This is the initialization command for the Flashcard. This command initializes the Flashcard as user unit 128 (see the documentation for the Attach BIOS for more information on user units).

The second parameter specifies that there are 576 blocks on the volume. This number was obtained from the table above.

The third parameter is the Pascal unit number you wish to use to refer to the Flashcard. In this case, we are configuring the system to refer to the Flashcard as volume #9:.

Note that the command is in lower case. From the above discussion, we know that this command will only execute if there is no volume named SSD-200: online (in other words, if the Flashcard has not yet been initialized).

Now, let's zero the directory of the Flashcard.

[z]SSD-200:,576,9

Again, we zero the directory only if this is a new initialization. We are calling the volume SSD-200:, it has 576 blocks, and looks like volume #9: to Pascal.

These are all the steps necessary to initialize the Flashcard for Pascal! Normally, however, you will want to exercise a number of other options in the Exec Processor. The complete command syntax is detailed below.

5.5. Exec Processor Command Syntax

[*]VOLNAME	change root volume to VOLNAME. This is handy for eliminating lengthy search sequences for system files.
[:]VOLNAME	change prefix name to VOLNAME. You may change the prefix volume to any name you choose with this command.
[:]	set prefix from a displayed menu. If you do not know which volumes will be online before you start the exec, this is a handy option.
[D]VOLNAME	set date on VOLNAME. For updating date stamps on files and volumes.
[D]	reads date from external DATE interface. The supplied interface requests the date from the user, and accepts it in the same format as the Filer D(ate command.

[J]FILENAME	jump to FILENAME for more orders (3 levels max.). If another diskette is mounted, you may wish to jump to an exec file on that volume for more orders.
[B]	jump back to caller or terminate execution. This command must be at the end of every file that is [j]umped to.
[T]FILENAME1,FILENAME2	transfer FILENAME1 to FILENAME2. This is the exec version of the Filer T(ransfer command. It should be used to transfer often-used files to the Flashcard upon boot.
[M]PARM	call external MOUNT interface with PARM. For example, if you needed the volume APPLE2: placed in one of the disk drives, you would issue the command: [m]APPLE2: The Exec Processor waits for the requested volume to come online, then proceeds.
[P]VOLNAME	get SYSTEM.PASCAL and SYSTEM.LIBRARY from volume. This command is particularly useful, as SYSTEM.PASCAL and SYSTEM.LIBRARY are two of the most frequently used files. This command allows them to reside on a volume other than the boot volume, if required.

The next several commands set the default filename for the Editor, Filer, Compiler, Assembler, and Linker. This feature allows you to specify the volume as well as the file name, thus eliminating the search sequence Pascal would normally go through. You may also use the system commands to execute the code file of your choice in place of the Editor, Filer, etc. simply by specifying that name in the appropriate command.

[E]FILENAME	use FILENAME for SYSTEM.EDITOR
[F]FILENAME	use FILENAME for SYSTEM.FILER
[C]FILENAME	use FILENAME for SYSTEM.COMPILER
[A]FILENAME	use FILENAME for SYSTEM.ASSMBLER
[L]FILENAME	use FILENAME for SYSTEM.LINKER

[X]FILENAME,PARM	load FILENAME into the execution queue, with Cval = PARM. At the end of your exec, the Exec Processor will perform a Chain to the specified file name. Note that the file is not immediately executed, but is deferred until the exec process terminates.
[G]FILENAME	do F(iler G(et for FILENAME.
[G]	display list of text files and select one from a menu.
[K]VOLNAME	do F(iler K(runch on VOLNAME.
[N]	New screen.
[H]VOLNAME,FN, ..	hold onto FN, .. on VOLNAME. An inverse R(emove.
[H]FN,FN,FN, ..	continuation of the hold file list
[H]	causes the execution of the hold command.
[S]CLEAR	do F(iler S(ave and clear work file.
[S]	do F(iler S(ave.
[R]x..x	read a character in [x..x] and execute exec commands at the label corresponding to the character entered. For example, if your labels are a, b, c, and d, the [R] command would read: [R]abcd
<x>	a label, used with [R], where x is a character in the acceptable set (as defined in the [R] command).
[W]X,Y,MESSAGE	write MESSAGE at X, Y on screen.
[O]N	output to console a control character with an ASCII value of N.
[Q]FILENAME	query the existence of FILENAME. If FILENAME ends in a colon (:), the Q command will search for the existence of a volume with the specified name. If the file or volume queried is found, a FLAG is set to true, otherwise the flag is set to false. This flag controls the conditional execution of lower case commands.

[Q]	change FLAG status (true to false, or vice versa).
[V]VOLNAME,SPACE	set FLAG to true if less than SPACE blocks remain on VOLNAME. This command is used to prevent an exec command from attempting to transfer data to a full volume.
[I]PARM1,PARM2,PARM3	call external INIT interface with PARM. The Flashcard initialization routines are invoked using this command. The parameters are:

Unit Number	128
Number of blocks	From table above
Volume Number	4,5,9..12

[Z]NAME,SIZE,UNIT	zero disk directory, set volume name, size, and unit number.
[U]PARM	call external USER function with PARM. No external user functions are supplied with the distribution copy of the Flashcard drivers.
[Y]	call V.N.A. menu driven file handler.

Blanks are ignored by the Exec Processor except in the [W] command.

6. CP/M Disk Emulation Software

6.1. Introduction & Summary

This section describes interface software which allows the user to configure one or two Flashcards to emulate various 5 1/4" and 8" floppy disks on an Apple // computer equipped with a Microsoft Softcard Z-80 adapter and the Microsoft adaptations 2.20B or 2.23 of the Digital Research Inc. CP/M operating system.

The Flashcard CP/M software can emulate drives of 144k, 288k, 432K, and 576K, with the size and number of emulated drives being dependent on the number and capacity of Flashcards available.

Turnkey systems can be easily created and used, thanks to the menu driven choice of a wide range of options.

6.2. CP/M Flashcard Software

There are two versions of the CP/M Flashcard software for use with the Microsoft implementations of CP/M. These two versions are contained on the following files:

SSDFLOP	Configure the Flashcard to look exactly like a floppy diskette. This includes the allocation of system tracks.
SSDRAM	Configure the Flashcard to use all the RAM available. This option is <u>not</u> compatible with standard floppy disk formats, and in some circumstances may not be desirable.

The Flashcard software is intended to accomodate all possible combinations of 1 or 2 2201 (single) or 2202 (dual) Flashcards, configuring them in several ways:

6.3. Possible Configurations

Flashcards	memory blocks	possible configurations
2 duals	8x64K + 4x16K	576K or 432K + 144K or 2x288K or 288K + 2x144K
1 dual, 1 single	6x64K + 3x16K	432K or 288K + 144K or 3x144K
1 dual or 2 singles	4x64K + 2x16K	288K or 2x144K
1 single	2x64K + 16K	144K

6.4. Configuration Notes

As was mentioned above, there are two versions of the CP/M software. The effect this has is as follows:

Configuration	Notes
144K	Under the SSDFLOP drivers, the 144K configuration provides you with approximately 128K of usable space. Under the SSDRAM drivers, the 144K configuration provides you with approximately 144K of usable space.
288K	Under the SSDFLOP drivers, the 288K configuration provides you with a 256K (8" Single Sided/Single Density) disk emulation. Obviously, this does not make use of the full capacity of a model 2202 (dual) Flashcard. The reason for this is twofold: 1. 256K is the standard size for 8" disks. 2. When CP/M needs to "grow" a file (that is, make it bigger), it uses what is known as an <u>extent</u>. These extents are normally 1K in length, but if disk size gets larger than 256K (as in the case of the SSDRAM drivers), they must be allocated in segments of 2K. This means that for many small files, the allocation of 2K extents does not utilize storage efficiently.
432K	The 432K configuration is incompatible with most floppy disk formats, and there has been no attempt made to place system tracks on the Flashcard under this configuration. Both the SSDRAM and SSDFLOP drivers function identically in this configuration.
576K	The 576K configuration under SSDFLOP provides for an emulation of a Single Sided/Dual Density 8" disk. Under this configuration, approximately 512K are available for data. Under SSDRAM, the full 576K of RAM is available for data storage.

6.5. Why Emulate 8" Drives?

For use with the Apple's "native" operating systems the standard configuration of the Flashcard is entirely satisfactory.

Apple software, whether under DOS or Pascal, is universally written for use on systems with 5 1/4" floppy disks. If you want to use the CP/M operating system, however, you may sometimes encounter a complication--CP/M was originally written for systems with 8" disk drives.

Such drives have approximately twice the data capacity of the Apple's disks. Programmers who may never have dreamed that their programs would be run on an Apple have sometimes created programs that assume that a lot of disk space is available.

The disk space requirements of a program are not always obvious. For example, a compiler may create and then erase files that are much longer than the source file or the final object file, and the user of the compiler may never know how much disk space is required until the system fails to run.

To solve this problem the programs SSDFLOP and SSDRAM are provided. These programs will initialize single or dual Flashcards so they look to CP/M like disks of 144Kbytes, 288 Kbytes, 432 Kbytes or 576 Kbytes capacity, depending upon system configuration, number, and type of Flashcards installed.

The Flashcard system can exactly emulate 8" single sided single density CP/M disk format, or can take better advantage of the storage space available to it, as detailed above in "configuration notes".

Since 8" single density disks are the common medium for program exchange among CP/M users, this means that the Flashcard will allow you to run virtually any CP/M program.

Note: It is possible for advanced applications to use drives larger than their emulated floppy equivalents. A CP/M 8" single sided single density floppy is 256K, but an emulated 8" SSSD drive can have up to 288K storage at your discretion.

6.6. Files Included on the Distribution Disk

The following files are included on this disk:

1. SSDFLOP.COM and SSDRAM.COM are the standard versions of the Flashcard CP/M disk emulation software.

Note: For the remainder of this document, the notation SSDxxx will be used to designate the files SSDFLOP.COM and SSDRAM.COM.

The SSDxxx files are menu driven, and are used to configure the Flashcard system. They may be used to create turnkey installation files for different configurations.

The "turnkey" file is named SSDINIT.COM, and will be described later. The patch code can be installed in the "I/O Configuration Block". This is a part of the CP/M memory reserved for custom interface programs in the Microsoft implementations of CP/M.

Installation of SSDxxx in the I/O Block does not change the amount of memory available to your programs. As a menu option, the code can alternately be installed in the high end of the "Transient Program Area", reducing the amount of memory available to your programs. The reduction in available memory will be about 2.5K bytes.

2. UNBUILD.COM copies a large file to a series of smaller files. It is meant to be used for downloading from a Flashcard or a high capacity drive to smaller drives.

   The complement to UNBUILD is BUILD, which recreates the original file. UNBUILD creates a header to identify the source and the subfiles, and writes this header at the beginning of the first subfile. A subfile is named with the same filename as the source file with an extension formed from the first letter of the source file extension and a decimal number 00..99.

3. BUILD.COM undoes the work of UNBUILD, uploading a series of small files to a single file on a large capacity drive. BUILD uses the header written by UNBUILD to recreate the file originally segmented.

4. FREE.COM is a utility that reports the upper limit of memory available to your programs.

5. FLASHDOC.TXT is a CP/M text file copy of this documentation.

6.7. Configuration & Installation

The following section describes the use of SSDxxx. Before you start, you should make a copy of the distribution disk and put the original away in a safe place. You might also want to put the CP/M system on the copy.

Note: there is **no copy of the CP/M operating system on the disk we provide.**

SSDxxx supports one or two Flashcards. A standard Apple disk controller must be in slot 6. The one or two Flashcards can be in any peripheral slot(s), but you should observe the CP/M system's protocol for slot use of 80 column cards (slot 3), printer cards (slot 1) and the like.

The SSDxxx interface is written in such a way that the Flashcard can be used with some other storage devices, such as 8" floppy disk drives, hard disks, or bubble memory devices if certain conditions are met. There cannot be more than six disk drives (including Flashcards) under CP/M 2.20B (44K or 56K) or more than four drives under CP/M 2.23 (60K). Additional considerations for systems with multiple storage devices are discussed in the section on hardware and software compatibility.

Installation is simple. Boot up your system and run SSDxxx when you get the CP/M 'A>' prompt:

```
        A>SSDFLOP
```

or

```
        A>SSDRAM
```

```
Options: A>SSDxxx <option>
               menu driven configuration, queries if it is to
               save SSDINIT and/or to proceed with installation

         or

         A>SSDINIT <option>
               install previous configured Flashcard drivers;
               turnkey file

               (A note on SSDINIT will follow later.)
```

The <option> may be one of the following (the first letter, either uppercase or lowercase, suffices):

RECON permits reconfiguration of previously configured version. (normally used with SSDINIT) The defaults will be those of the SSDINIT configuration.

OLD the OLD option prevents erasure of the Flashcard directory, allows reinstallation with preservation of Flashcard contents. Overrules the Autoload flag.

LOAD forces an Autoload from the current default disk to the first Flashcard of compatible format (5-1/4" disk to 144K Flashcard and 8" single sided single density disk to 288k Flashcard). Overrules a no-autoload configuration.

(a note on autoloading follows later.)

CLEAR overrules the Autoload flag. Leaves a completely erased directory.

Note: if you initialize with the OLD option at power-up time, when in fact the directory area of the Flashcard is full of garbage, you probably will not be able to erase the directory completely with an ERA *.* command. You will have to do a cold boot again because SSDxxx will refuse to initialize an already-initialized Flashcard.

Once the program is running, you will be prompted by the program for various option choices. Here is a sample of what you might see. On the left will be an example of an actual (possible) configuration. On the right will be some additional information and commentary. Bold type in the simulated configuration represents user responses to the program.

```
On screen:                                            Commentary:

  FLASHCARD Installation Program
              by
   *** Woodhull and Gilbert ***
              for
Synetix, Inc., Redmond, Washington

     Version 2.0 rev. 08/09/83
   Copyright 1983,  A. S. Woodhull
     CP/M version: 2.20B, 56K                          -Note CP/M version
                                                       found.

4 more drive(s) may be installed                       -Program uses
  ...press any key to continue                         version to
                                                       determine number
                                                       of drives.

Default slot for 1st FLASHCARD is 5                    -Program allows
Press CR to retain, or enter slot                      your choice of
number (1-7)  --> <RETURN>                             slots for one
                                                       or two cards.
The 1st FLASHCARD type is: DUAL
Is this correct? (Y/N) --> Y

Default slot for 2nd FLASHCARD is 4
Press RETURN to retain, or
enter new slot number, 1-7  --> <RETURN>

The 2nd FLASHCARD type is: DUAL
Is this correct? (Y/N) --> Y

Current configuration:                                 -Program recalls
1 576 Kbyte Capacity Simulated Disk                    current or
Do you want to keep this configuration?                default
(Y/N) --> N                                            configuration.
```

```
Possible configurations:                          -Program shows
1 576 Kbyte Capacity Simulated Disk               all possible
or                                                configurations
1 432 Kbyte Capacity Simulated Disk               with number &
1 144K Kbyte Capacity Simulated Disk(s)           type of
or                                                Flashcards in
2 288K Kbyte Capacity Simulated Disk(s)           system, and
or                                                remaining number
1 288K Kbyte Capacity Simulated Disk(s)           of drives
2 144K Kbyte Capacity Simulated Disk(s)           unallocated.
or
4 144K Kbyte Capacity Simulated Disk(s)

Allocate a 576K SSD? (Y/N) --> N                  -Program prompts
Allocate a 432K SSD? (Y/N) --> Y                  for your choice
Remainder will be 1 144K Kbyte Capacity           of setup.
Simulated Disk(s)

Current configuration:                            -Program shows
1 432 Kbyte Capacity Simulated Disk               current setup &
1 144K Kbyte Capacity Simulated Disk(s)           again offers the
Do you want to keep this configuration?           choice to keep or
(Y/N) --> Y                                       change it.

Current code location is the                      -Program offers
Transient Program Area                            choice of TPA or
Do you want to keep it there? (Y/N) --> Y         I/O Block
                                                  installation.

Do you want autoloading of an SSD                 -Program offers
with the contents of the default                  autoload ONLY if
disk? (Y/N) --> N                                 appropriate
                                                  formats match.
                                                  (See Notes on
                                                  Autoloading.)

Save new SSDINIT? (Y/N) --> Y                     -Program offers
                                                  option to save
                                                  new turnkey file
                                                  and option to
                                                  install the
                                                  configuration.
```

If all has gone well you are done with the installation procedure now.

You can now use PIP to move the contents of your floppy drives to each of the Flashcards you have installed, and proceed as if you have that many more disks on-line.

You may have already used the autoload feature to accomplish part of this. You might want to write a submit file, to automate the process if you use the Flashcard(s) in much the same way each time. Just don't forget that your Flashcards are not real floppy disks, and you need to transfer anything you really want to save back to a floppy.

6.8. A Note About SSDINIT and Turnkey Systems

SSDINIT.COM is not on the distribution disk; it is created by saving a turnkey configuration.

Whenever a new turnkey version is saved any previous SSDINIT version will be erased.

Once SSDINIT has been created it is completely independent and self-sufficient, and is in fact an exact backup of SSDxxx.COM except that it contains a customized table of configuration parameters and a flag that causes the initial dialogue to be skipped unless it is invoked with the RECON option.

The rationale for this process is that the distribution version should not be subject to automatic erasure. Multiple turnkey files for different configurations can co-exist: each time you create a new SSDINIT file simply rename it, preferably to a filename that indicates its function. Example: SSD576.

6.9. A Note About Autoload

The autoload feature is a convenient way of copying the contents of the default drive at bootup onto the Flashcard systems. It has some necessary limitations: it can only autoload 5 1/4" disks to 144K Flashcard drives or 8" single sided single density disks to 288K Flashcard drives. This is because the formats must match between the floppies and the emulated drives. PIP alone or in combination with submit files can be used for easy transfers of mixed or other formats. Note: for special applications it is possible to Autoload 576K Flashcard configurations. This is an assembly option of the SSDxxx file, or patchable with DDT. Please write Synetix for details.

If you use SSDRAM, autoloading will normally not be possible, because your emulation disk will be a different size than a standard floppy disk. SSDFLOP will solve this problem.

6.10. Error Messages

SSDxxx may not work properly on the first try--if instead of the lines quoted above you received a message indicating an error, you can't proceed immediately.

Several error messages are possible in attempting to install the SSDxxx interface software:

"Flashcard installation failed"

This means the installation process was terminated and no code was patched in. This message is usually preceded with one of the other messages, giving some possible indication of reason for the failure.

"No space for code in i/o configuration block"

You must either configure for the Transient Program Area or modify the system to free the lower 300 bytes (approximately) of the I/O configuration block for use by the SSDxxx routines. To prevent a system crash, SSDxxx will issue this message if it finds any value other than 0 in the portion of memory it uses.

"Flashcard card already initialized, or not found"

You will get this message if a Flashcard card is not present at all, or if you have already run SSDxxx since the last time you did a cold boot.

Note: if you tell the program you have a 2201 (single), it will not check to see if you indeed have a 2202 (dual) in the indicated slot. This is not necessarily an error, but a valuable information resource.

"CP/M version: cannot be verified"

SSDxxx refers to a few addresses within CP/M which are "magic numbers" that depend on the version of CP/M in use.

The installation programs look for a particular pattern in memory which identifies the version. The pattern is actually a part of the sign-on message that CP/M displays when cold-booted, and it can be overwritten when no longer needed.

Normally you should run the SSDxxx program when you initialize the system to avoid this problem. You will also receive this message if you have a new version of CP/M, or if you have a version that has been radically customized, perhaps with code from another alternative peripheral driver. In the first case we will attempt to provide the information you need to accomodate a new CP/M version as soon as possible. In the second case you may be on your own, but we have tried to make modifications to custom versions as easy as possible, and you should contact Synetix for help.

"Adequate Flashcard not found in slot x"

You may have specified (incorrectly) that there was a 2202 in slot x and the program found a 2201.

"Warning: Too many drives requested" or "Too many drives already installed"

The program knows what version of CP/M it is working with. If you ask it to install more drives than the version can accomodate, it will ask you to modify your demand.

As mentioned earlier, CP/M version 2.20B can accomodate six drives, but CP/M version 2.23 can handle only four. If you want to use two Apple drives and two Flashcards and an 8" floppy disk drive you won't be able to use the 2.23 version (60K). You should also note that CP/M will think there are at least two Apple drives--the program senses the controller, not the individual drives.

A case related to the message above will arise if you have two Flashcards, but CP/M can only handle one of them within the four or six drive limit. No error message will be generated, but the program will stop after installing the first Flashcard.

"Error writing file"

This error relates to SSDxxx attempting to write a turnkey file SSDINIT and not being able to. Possible reasons: the default disk is write protected; the disk is full; or the disk contains an R/O (read only) copy of SSDINIT.

"No format match or erase prevented"

This occurs when an Autoload can't be done either because an emulated disk cannot be found to match the default disk's format or because Autoload was explicitly overwritten.

6.11. Utilities

6.11.1. Unbuild

UNBUILD copies a large file to a series of smaller files. It meant to be used for downloading from a Flashcard or a high capacity drive to smaller drives. The complement to UNBUILD is BUILD, which recreates the original file. UNBUILD creates a header to identify the source and the subfiles, and writes this header at the beginning of the first subfile. A subfile is named with the same filename as the source file with an extension formed from the first letter of the source file extension and a decimal number 00..99. The usage:

```
A>UNBUILD <destination>:=<source>:<unambiguous filename>
```

where <destination> and <source> are valid drive designations A..P and the filename is the name of the large file to be divided.

```
EXAMPLE:  A:UNBUILD B:=C:BIGFILE.TXT
```

A series of subfiles BIGFILE.T00, BIGFILE.T01, etc. will be written to drive B:. Before attempting to write each subfile UNBUILD will verify that there is space on B: and will allow you to change disks if there is not enough room. Up to 12 subfiles of 64K length may be created, allowing a single file occupying all the space on a 576K Flashcard emulated disk to be stored on 6 standard 5 1/4" floppy disks. You should note carefully which subfiles are on each floppy disk, to avoid confusion when you want to recreate the big file.

Note: UNBUILD was designed to work on files created in sequential mode; do not use it for random-access files which may have missing records.

6.11.2. Build

BUILD undoes the work of UNBUILD, uploading a series of small files to a single file on a large capacity drive. UNBUILD creates a header to identify the source and the subfiles, and writes this header at the beginning of the first subfile. A subfile is named with the same filename as the source file with an extension formed from the first letter of the source file extension and a decimal number 00..99. The usage:

```
     A>UNBUILD <destination>:<unambiguous filename>=<source>:
```

where <destination> and <source> are valid drive designations A..P and the file name is the name of the large file to be created.

```
     EXAMPLE:A>UNBUILD C:BIGFILE.TXT=B:
```

It is your responsibility to check that you have enough room on the destination disk before you start. You must supply disks containing the required subfiles in order; although you will be prompted for which file comes next you ought to look at the directories of the subfile disks and place them in order before you start. If a required file is not found when needed BUILD does not abort unless you request it to, so you must insure that files are where they ought to be, aided by prompting.

6.12. Hardware and Software Compatibilities

It would be impossible to test all possible interactions between all possible hardware and software products available for the Apple under CP/M.

While CP/M is designed to work with many combinations of devices and programs, we cannot predict **all** interactions between the Flashcard or its driver software and other hardware or software.

Prudence dictates that you should test out a new combination of devices thoroughly using disks that do not hold any irreplaceable data or programs. The SSDxxx installation program checks many things, but there is no program that can't be fooled or accidentally altered after installation.

Some possible problem areas are:

1. Some programs, like Microshell or the CP/M Users Group UNSPOOL utility, alter the CP/M system when loaded. SSDxxx is compatible with these, but the SSDxxx program should be run before installing one of these other programs.

2. Other programs may alter the CP/M system in ways that interfere with the SSDxxx drivers. The most likely source of problems will be hard disks or other kinds of RAM disk simulators, because these devices have to alter the same parts of the CP/M system that are altered by SSDxxx. In general, such routines must alter certain JMP or CALL instructions in order to ensure that any calls to, say, a general purpose routine to read data from a disk are diverted to the new subroutines. The added subroutines must then pass control back to the old routines if, for example, the request for data calls for reading one of the original Apple disk drives. SSDxxx handles this in such a way that additional modifications of this type can be accomodated either before or after installing the Flashcard code, but we can't guarantee that other programs will fit themselves in as well as we think ours does.

3. If other peripheral devices use the I/O Configuration Block and are installed following installation of SSDxxx the SSDxxx code could be overwritten. If you write any programs that relocate code into the I/O Configuration Block you should include a loop that verifies that the portion of the block to be used is filled with zeroes. This is the technique used by SSDxxx to verify that the area it needs is free. The Flashcard software uses only as much of the I/O Configuration Block as it needs.

   Using a single format takes up less space than using mixed formats (emulated drive sizes), and each emulated drive takes up additional storage.

4. Memory space is scarce within the CP/M system. System customizers are tempted to use data areas that "wouldn't be missed", such as the areas reserved for use by additional disk drives. All such problems can in principle be overcome, but a general purpose program cannot foresee them. If you have an unusual combination of peripheral devices you may need to do, or to have someone else do, some custom programming. In many cases of apparent incompatibility the problem will be found to be that two different programmers tried to use a common area of memory for different purposes.

Finally, you should note that many programs that have been designed particularly for certain hardware configurations will not work with the Flashcard. Beware of COPY or FORMAT or any programs with similar names--they are likely to have been written for use with a particular kind of disk drive. You don't need to format the Flashcard anyway, and you are probably safest using PIP to copy programs to or from the Flashcard. It is a good idea to have a system experiment disk, with expendable data, to use while you are doing anything doubtful.

One caveat: while testing, we noted that the current version of VISTA CP/M for its VISTA 8" A800 controller and subsystems is not compatible with the Flashcard software.

6.13. Disk Format Details

Programs that run under the CP/M operating system do not usually care about the actual format of disks, but once in a while there is an exception. To handle even the exceptions, the Flashcard (configured as a 288K drive) should be made to resemble a standard single density format disk in detail, using SSDFLOP. This emulation includes:

simulation of two system "tracks" preceding the area used for the directory and files

simulation of 26 "sectors" of 128 bytes each per "track"

There is one way the simulated disk does not resemble a real one, however. A full 256K bytes is available for storage, rather than the 240K available on a real 8" floppy, so the Flashcard appears to have 81 tracks rather than 77. One of the 16K blocks of memory on the Flashcard board is not currently used. This is not an oversight, but a deliberate decision. Because of the way CP/M allocates disk space, a disk with less than 256K of storage can be divided up into blocks of 1K, but a larger disk must be divided into 2K blocks. For most users the available storage would be less if we had tried to make the additional memory available. If it is desirable to utilize the extra memory, use the SSDRAM drivers.

A similar parallel exists between the 144K format and a standard Apple 5 1/4" CP/M diskette.

7. Advanced Programming Techniques

This section is intended for those who wish to use the Flashcard for other purposes than disk emulation, and for those who are simply curious about how the Flashcard is accessed. Some of the information in this section will be in "computerese", and some assumptions have been made about the target audience. If you find some of the terms used in this section unfamiliar, several excellent reference pieces are available to clear up most of these definitions:

Leventhal, Lance <u>6502 Assembly Language Programming</u>, Osborne/McGraw-Hill, Berkeley, 1979

<u>Apple][Reference Manual</u>, Apple Computer, Inc.

<u>Apple //e Reference Manual</u>, Apple Computer, Inc.

7.1. Address Locations

Unlike some large RAM boards for Apple computers, the Flashcard's memory is not bank-switched into the upper 16K region of the Apple address space. Instead, the memory is serially addressable.

It is important to understand the logical design of the Flashcard before going on.

The Flashcard is divided into three or six banks of memory, depending upon whether you have a model 2201 (147K) or 2202 (294K).

These banks are not all the same, and are detailed in the table below:

Bank Number	Size	End Address	Model
0	64K	$FFFF (65535)	2201, 2202
1	64K	$FFFF (65535)	2201, 2202
2	16K	$3FFF (16383)	2201, 2202
3	64K	$FFFF (65535)	2202
4	64K	$FFFF (65535)	2202
5	16K	$3FFF (16383)	2202

Note that banks 3, 4, and 5 are not present on the model 2201 (147K) Flashcard.

Before entering into a discussion of addressing protocols, it is important to understand how the Flashcard utilizes the Apple's I/O address space:

We will use the letter "n" to designate slot number for the following table of I/O address space utilization.

$C0n0	Least significant byte of bank-relative address
$C0n1	Most significant byte of bank-relative address
$C0n2	Bank 0 data location
$C0n3	Bank 1 data location
$C0n4	Bank 2 data location
$C0n5	Bank 3 data location
$C0n6	Bank 4 data location
$C0n7	Bank 5 data location

The phrase "bank-relative" was used above. What this means is the address relative to the <u>beginning</u> <u>of</u> <u>the</u> <u>bank</u> -- not the beginning of the Flashcard memory.

Note: The "n" <u>must</u> have the high bit set in order to properly specify the locations above. For example:

Slot 1 addresses are in the $C09x range.

Slot 2 addresses are in the $C0Ax range.

Slot 3 addresses are in the $C0Bx range.

Slot 4 addresses are in the $C0Cx range.

Slot 5 addresses are in the $C0Dx range.

Slot 6 addresses are in the $C0Ex range.

Slot 7 addresses are in the $C0Fx range.

Where x is 0 through 7 (for the Flashcard).

7.2. Addressing Protocol

In order to address the Flashcard, several steps must be taken:

1. Determine the bank in which the data you wish to write or read resides.

2. Determine the byte relative to the beginning of the bank to be read or written.

3. Store the bank relative address in the first two address locations in the I/O space for the slot the Flashcard is in ($C0n0, $C0n1). This is done in the normal Apple manner with the least significant byte first.

   <u>Important</u> <u>Note:</u>

   The 16K banks are addressed somewhat differently from the 64K banks. The difference is as follows:

   64K banks require a 16-bit address consisting of an 8-bit "least significant byte", and an 8-bit "most significant byte" written to addresses $C0n0 and $C0n1, respectively.

   16K banks require a 14-bit address consisting of a 7-bit "least significant byte", and a 7-bit "most significant byte" written to addresses $C0n0 and $C0n1, respectively.

4. Read from, or write to the address specified by $C0nx, where "n" is the slot number + $80 (high bit set), and x is the Flashcard bank number + 2.

Note: If you attempt to read from, or write to unpopulated banks of a Flashcard, you will get random data, but no indication of error. The burden is upon you to check the persistence of the RAM to determine whether you are dealing with a 147K or 294K Flashcard.

8. Appendix 1 -- DOS 3.3 Applications Notes

Many DOS 3.3 applications programs are compatible with the Flashcard. The primary criterion for determining whether or not a program is compatible with the Flashcard is to try to copy it using COPYA.

If it will not copy, try to CATALOG it. If you can obtain a catalog of the disk, there is a good chance that you can use it with the Flashcard.

The normal procedure for starting an applications program is to:

1. Install the Flashcard drivers by either booting the Flashcard driver disk, or by placing the driver disk in the drive and typing:

   BRUN SSD DRIVER/STD

   or

   BRUN SSD DRIVER/ALT

   depending upon which set of drivers best suits your application. In many cases, the version of the drivers you use will not matter. For more information on selection of the correct driver, see the section on the DOS 3.3 Disk Emulation Software, and the Application Notes that follow.

2. RUN or BRUN the greeting program (often titled "HELLO" or something similar).

The following is a list of some of the programs we have found compatible with the Flashcard under DOS 3.3.

Ascii Express "The Professional" -- Southwestern Data Systems

This popular telecommunications program is extraordinarily effective in combination with the Flashcard. Ascii Express Pro keeps the editor on disk, and most of the telecommunications program in memory. When you want to edit a file, it takes quite some time to bring the editor in from the disk.

One way to run Ascii Express Pro with the Flashcard is to boot the Flashcard drivers using the /ALT version, or type:

BRUN SSD DRIVER/ALT

Then BRUN COPY to copy the Ascii Express Pro disk to the Flashcard.

For this example, assume that the Flashcard is in slot 5. To start Ascii Express Pro, type:

BRUN AE

Ascii Express Pro will then be run from the Flashcard.

One of the major advantages of the use of the Flashcard is that saving and loading files is nearly instantaneous, so connect time to expensive timesharing services is held to a minimum.

If you are after astonishing performance, try using Ascii Express Pro with DiversiDOS, and the DDMOVER, as documented in the section entitled DOS 3.3 - Flashcard and DiversiDOS.

Data Capture 4.0 -- Southern Systems

This is another popular telecommunications program in the DOS 3.3 environment. This program may be run from the Flashcard in much the same way as Ascii Express Pro.

Simply BRUN SSD DRIVER/ALT, then BRUN COPY to move the Data Capture 4.0 program to the Flashcard. Finally type:

RUN DATA CAPTURE 4.0

TASC (The Applesoft Compiler) -- Microsoft

TASC is an excellent compiler, but it tends to be extremely slow because of the great amount of disk access it requires.

Fortunately, TASC is compatible with the Flashcard, which can speed up its effective execution time immensely.

To run TASC on the Flashcard:

BRUN SSD DRIVER/ALT

then

BRUN COPY

to copy the compiler programs to the Flashcard. You should also copy the Applesoft source code to the Flashcard. Finally, type:

BRUN TASC

and TASC will run from the Flashcard.

The pattern for running DOS 3.3 applications software on the flashcard should begin to become apparent. You must take several steps before starting your applications program:

1. Install the appropriate driver software -- experimentation should show which version (SSD DRIVER/STD or SSD DRIVER/ALT) works best with your particular application. In some cases there will be now difference.

2. Install a DOS enhancer, if you want to use one.

3. Start your application program by RUNning or BRUNning the "hello" program.

Several other applications that run well with the Flashcard in the DOS 3.3 environment are:

Complete Graphics System -- Penguin Software

EPF IV -- Sierra Online

Hayden Compiler -- Hayden Software

Merlin Assembler -- Southwestern Data Systems

MagiCalc -- Artsci, Inc.

Magic Window][-- Artsci, Inc.

Magic Words -- Artsci, Inc.

ORCA/M Macro Assembler -- Hayden Software*

PIE Writer -- Hayden Software

ScreenWriter][-- Sierra Online*

The General Manager -- Sierra Online

THE Spreadsheet -- A.P.P.L.E.

The software packages listed above represent a sampling of applications packages than may be run in the DOS 3.3 environment using the Flashcard. There has been no attempt made here to list all compatible software.

*Compatibility is not yet in place for this piece of software, but Synetix is looking into this problem.

9. Appendix 2 -- CP/M Applications Notes

The CP/M environment is ideal for the Flashcard for several reasons:

- The Flashcard more closely approximates the disk environment for which most CP/M programs were originally conceived.

- The Flashcard takes excellent advantage of CP/M's already enhanced disk I/O routines. This makes it inherently quicker than DOS 3.3.

- Virtually **no** CP/M software is copy protected, so almost all of it will run on the Flashcard.

In order to run a CP/M application, you must first install the Flashcard drivers, as detailed in the section entitled "CP/M Disk Emulation Software".

Next, you should use the PIP program (which is supplied with your copy of CP/M) to copy the program and data files to your Flashcard.

Now you may run your application from the Flashcard.

Several examples are provided below of a normal startup procedure for a system with a dual Flashcard.

WordStar -- MicroPro International

Boot your system from a CP/M master disk.

Place your driver diskette in one of your floppy disk drives (B: for this example).

Type: **B:SSDINIT**

Note: SSDINIT is the "turnkey" file generated by one of the two CP/M driver programs, SSDFLOP and SSDRAM.

When the initialization program is done, it will inform you of the configuration (288K Flashcard Configured as Drive C: for this example), and prompt you to place a system disk in drive A:

Once this process is complete, you may place your installed WordStar disk in one of your floppy disk drives (B: for this example), and transfer the program to the Flashcard as follows:

PIP C:=B:WS*.*

Now, you should change the logged drive to C: by typing:

C:

At this point you may run WordStar by typing:

WS

WordStar has two overlay files. It keeps track of where to look for these files internally. It first searches the "logged" disk drive, then the default disk drive. The logged disk drive in the example above was C:. If, however, you change the logged disk drive from within WordStar to, for example B: (using the L command from the OPENING MENU, or the ^KL command from edit mode), WordStar will not find the overlay files on the logged disk drive, and will be forced to look at the default disk drive. Normally, the default disk drive is A:. To speed up overlay swapping, it is desirable to modify this so that WordStar's default disk drive is C: (the Flashcard). This is done as follows:

Versions 3.01p and earlier:

Re-install WordStar, and do not change anything. When the install program asks whether you are done, answer N (no). This will place you in the "patcher". Complete documentation on the patcher is contained in the WordStar manual in the Installation section.

The patcher will prompt you for the address or label to modify.

You want to modify DEFDSK:

Note: The colon at the end of the label is necessary.

The value should be 1 now. Type the number 3 to specify drive C:.

Now exit the patcher by typing a zero in response to the "address to modify" prompt. This will exit the patcher.

Save your newly installed version, and it will execute very quickly from the Flashcard.

Versions 3.30 and later:

Versions 3.30 and later allow you to specify certain WordStar features. One of these features is the "System Disk Drive" (option R).

This should be set to C:.

dBase II -- Ashton-Tate

dBase II is a popular database management program that performs much better when run from the Flashcard. There are two reasons for this:

- dBase II is really a system of programs, and depending on the command being executed at the moment, it may need to read in an overlay (program) file. This overlaying process is almost transparent to the user when dBase II is being run from the Flashcard.
- dBase II maintains files that are larger than the amount of RAM available in the Apple. For this reason, dBase must spend a great deal of time performing disk I/O. When index files are being used, the amount of disk I/O increases. The Flashcard improves the speed of these I/O's, thus improving the overall performance of the dBase program.

To install dBase II on the Flashcard, follow the instructions above on hooking up the Flashcard drivers. Next use PIP to move the dBase programs to the Flashcard as follows:

PIP C:=B:DB*.*

At the completion of this file transfer process, you may change logged drives to C:, and start dBase as you normally would.

Note: In most circumstances, you will also wish to PIP some data files to the Flashcard.

Some other CP/M applications that function extremely well with the Flashcard are:

A.L.D.S. -- Microsoft

InfoStar -- MicroPro

MailMerge -- MicroPro

Multiplan -- Microsoft

Punctuation and Style -- Oasis Systems

SpellStar -- MicroPro

SuperSort -- MicroPro

SuperCalc -- Sorcim

The Final Word -- Mark of the Unicorn

The Word Plus -- Oasis Systems

10. Appendix 3 -- Additional Driver Software

The following disk emulation (driver) software is packaged with your Flashcard:

- o DOS 3.3 Disk Emulation Software
- o Apple Pascal (Pascal 1.1) Disk Emulation Software
- o CP/M (Microsoft implementation) Disk Emulation Software

You may elect to purchase additional disk emulation software from Synetix. The following products are available:

- o Pascal IV.1 (Softech) Disk Emulation Software
- o Modula-2 Disk Emulation Software
- o CP/M (Appli-Card or StarCard implementation) Disk Emulation Software
- o CP/M+ (version 3.0) Disk Emulation Software*
- o CP/M (Microsoft Softcard //e implementation Disk Emulation Software*

*Currently under development

11. Index