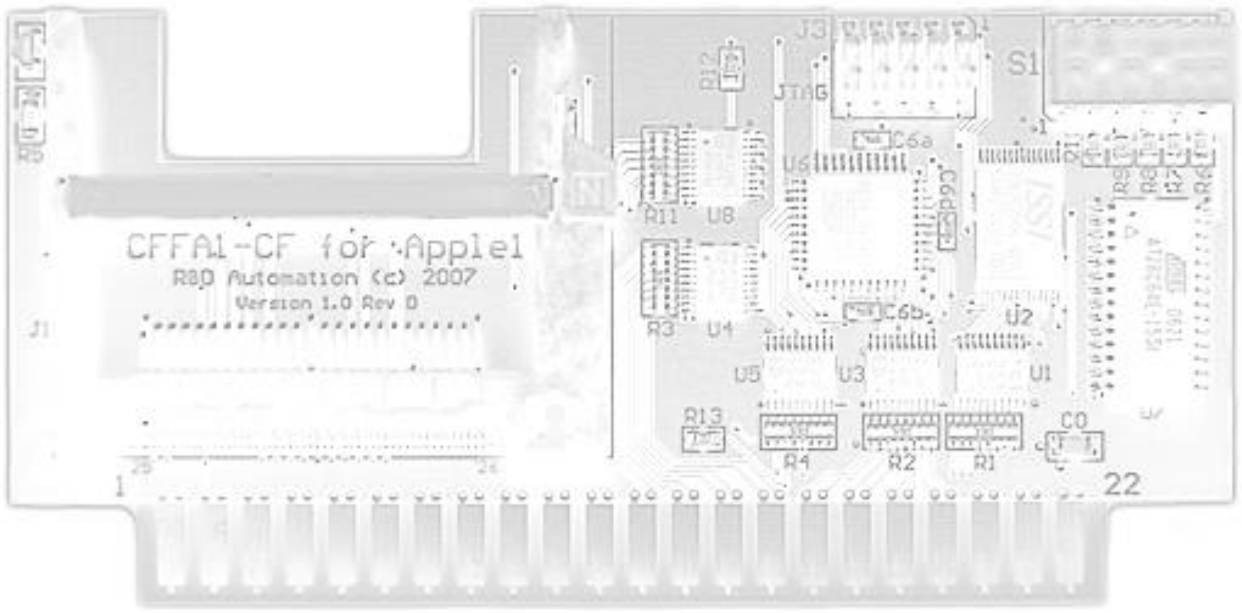
CFFA1 – CompactFlash Interface for Apple 1



Manual v1.1 for CFFA1 v1.0 RevD

Disclaimer of All Liability

Plain English Version:

Do not use this manual or the CFFA1 Interface card for any mission-critical applications, or for any purpose, in which a bug or failure could cause you a financial or material loss. This product was designed to enhance your Apple 1 computing experience, but may contain design flaws that could inhibit its proper operation, or result in a loss of the data recorded on the storage devices attached to it. When using this product you assume all risks associated with operation or data loss. If these terms are not acceptable, you may return the product for a refund.

Legalese Version:

Richard Dreher, doing business as R&D Automation, makes no warranties either expressed or implied with respect to this manual or with respect to the software or firmware described in this manual, its quality, performance, or fitness for any particular purpose. All software and firmware is sold or licensed “as is”. Any risk of incidental or consequential damages resulting from the use of the information in this manual or the software / firmware / hardware described herein, shall be assumed by the user or buyer or licensee. In no event will R&D Automation be liable for direct, indirect, incidental or consequential damages resulting from any defect in the software / firmware / hardware described in this manual.

R&D Automation reserves the right to make changes and improvements to the product described in this manual at any time and without notice.

Contents

Basic Information

Warranty and Return Information

You may return the CFFA1 Interface card for any reason within 30 days of receiving it. This should allow you enough time to evaluate the compatibility with your system. R&D Automation guarantees your CFFA1 Interface card to be free of defects under normal usage for a period of one year from the date you receive the product. If the card fails, and you have treated it properly, R&D Automation will repair, replace, or refund your money at our discretion, to be determined by R&D Automation on a case-by-case basis.

If you want to return the product under warranty, please contact us via E-mail to discuss return arrangements. Include your name and the serial number from the sticker on the back of the card. It is your responsibility to get the product you are returning back to us. We are not responsible for lost return shipments. Please choose shipping methods and insurance, as you deem necessary.

Warnings

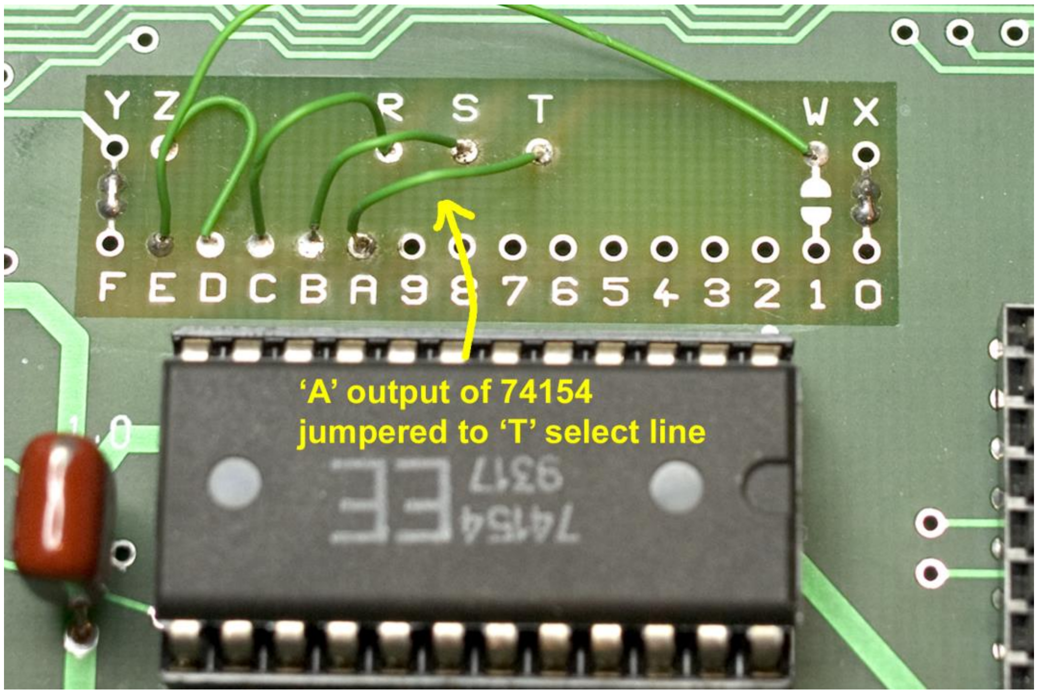
You should avoid electrostatic discharge to the CFFA1 Interface card. Like all electronics devices, static “shock” can destroy or shorten the life span of the CFFA1 Interface card. Avoid touching the CFFA1 Interface card after you have walked across the room, especially over carpet, and especially in dry weather.

You should safely discharge yourself before you handle the CFFA1 Interface card. This can be done by momentarily coming into contact with a grounded piece of metal.

In all cases, please exercise common sense and observe all electrical warnings provided by the manufacturers of the equipment you are using.

Quick Start Instructions

1. Be sure that your Apple1 has a jumper between address select points A and T, as shown in Figure 1. Note: *Replica1 computers have a static memory mapping scheme and need no modifications for the CFFA1.*



Y Z R S T W X
F E D C B A 9 8 7 6 5 4 3 2 1 0
'A' output of 74154 jumpered to 'T' select line

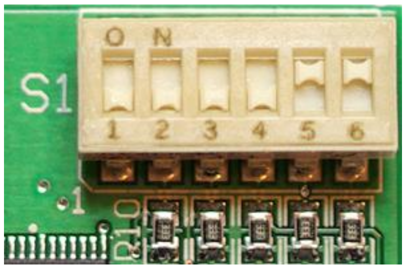
Figure 1: Typical mapping of Apple1 with the additional jumper for CFFA1.

2. Discharge yourself of excess static charge.
3. Remove the CFFA1 Interface from the anti-static bag.
4. Turn off power to your Apple1/Replica1 computer. **Do not skip this step!**
5. Configure dip switch S1 on the CFFA1 (upper right corner) for your computer. Table 1 gives the meaning of each switch. Figure 2 shows the typical configuration for the Apple1 and Replica1 computers. Note: Mimeo1 and Obtronix Apple1 clones should use the Apple1 settings.

Switch	Purpose	Apple1	Replica1
1	Not used	**X**	**X**
2	Disable SRAM decode from $1000 to $7FFF	**OFF**	**ON**
3	Low at CPLD input pin 19 (not used currently)	**X**	**X**
4	EEPROM Write Enable	**OFF**	**OFF**
5	SRAM Enable	**ON**	**ON**
6	Connects Apple1 Reset to CFFA1 CF card socket	**ON**	**ON**

Table 1: CFFA1 connections and jumpers PCB v1.0 Rev D
*NOTE: **ON** = switch up, **OFF** = switch down, and **X** = Don't Care*

Typical S1 settings for Apple1



Typical S1 settings for Replica1

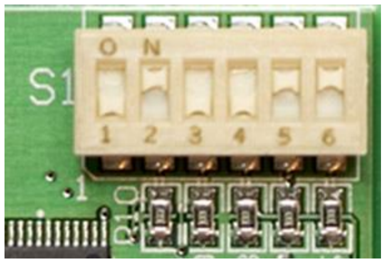


Figure 2: Typical dip switch configuration for the Apple1 and Replica1 computers.

6. Before Inserting the CFFA1 Interface into your computer, determine the proper orientation using the silk-screen labels: '**1**', '**22**', '**A**' and '**Z**' located near the gold edge card fingers. Match these labels up with the labels on the host computer before inserting the CFFA1. See Figure 3 below for a picture of the CFFA1 correctly inserted in the Apple1.

***** Warning: Inserting the CFFA1 card backward will destroy the CFFA1 *****

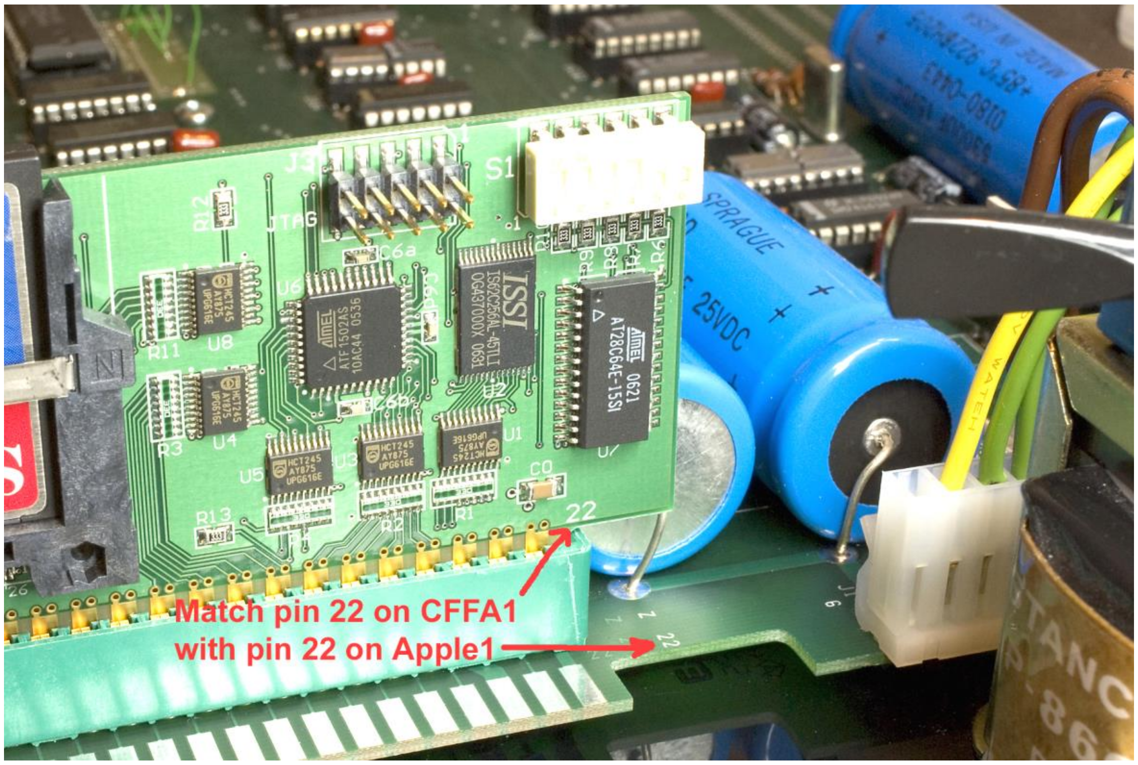


*Figure 3: CFFA1 inserted **correctly** in Obtronix clone of the Apple1.*

7. Insert the CFFA1 Interface into signal slot on the Apple1. The Replica1 SE had no slot required the optional slot expansion board for the CFFA1 to work. All slots are wired in parallel so they should all work the same.
8. Insert a CompactFlash memory card into your CFFA1 interface. You can insert the card anytime, as long as you power cycle your computer after you insert the card. NOTE: Depending on the brand of CF card, you will either have to power cycle or press reset after each insertion of a CF card into the CFFA1.
9. Power up your Apple1 computer.

10. Optionally, press the clear screen button on the Apple1.
11. Press the Reset button on the Apple1.
12. Type 9000R to access the CFFA1's interactive menu. The firmware for the CFFA1 is located at $9000 to $AFFF.

Note to Replica1 SE Users:

You may need to connect a short ground wire from the Replica1 to its expansion board to get reliable operation from the CFFA1 (see Figure 4). The symptom you may notice is an unexpected I/O error when accessing the CF card from a simple Catalog menu command. If you get I/O errors, try adding the ground wire. The best connection will be a soldered connection, but other temporary connections may work also.

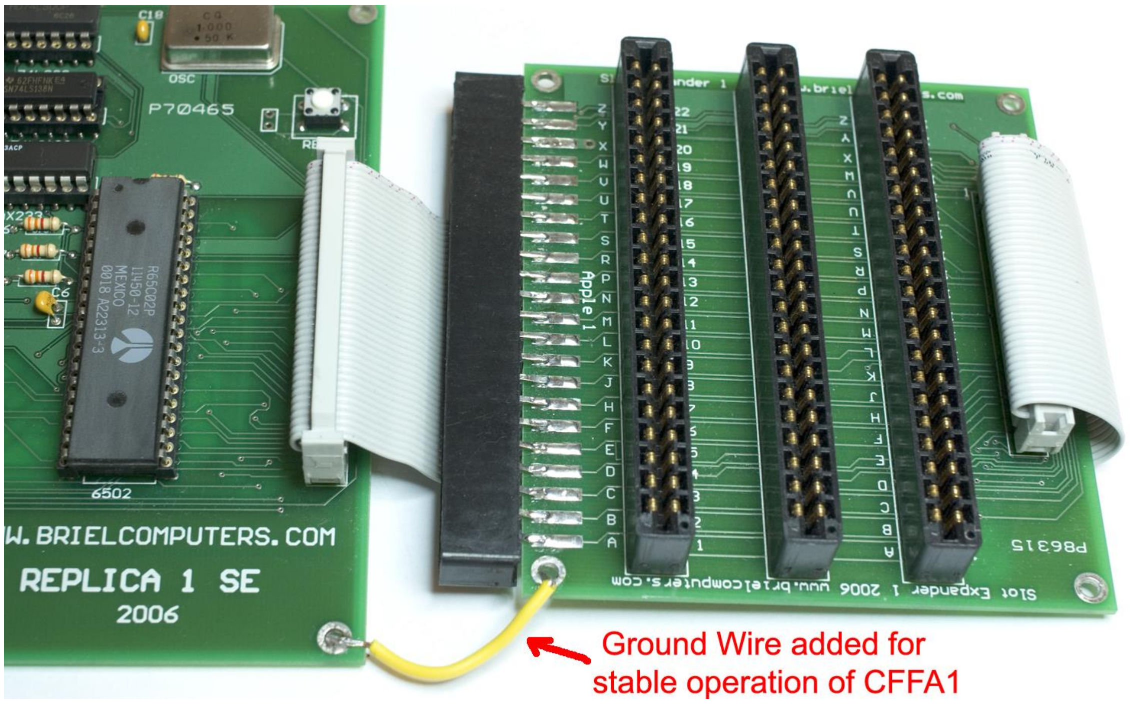


Figure 4: Replica1 SE and expansion board with ground wire added

Installation Details

The CFFA1 Interface card comes in an anti-static bag. This bag is a good place to store the card at times when the card is not installed in a computer. Before opening the zip-top bag, be sure you do not have a static charge built up in your body. The CFFA1 Interface card can be installed in the Apple1's expansion slot. To determine the correct cabling for the Replica1 SE and its expansion board, look on the bottom of the boards to find the square pin that denotes pin 1 of the connectors. The Replica 1 TE has a built-in slot like the Apple 1.

The following figure and tables define the location and meaning of all of the connectors and switches on the CFFA1 card. Use these tables to set the switches for CFFA1 operation in your Apple1 computer.

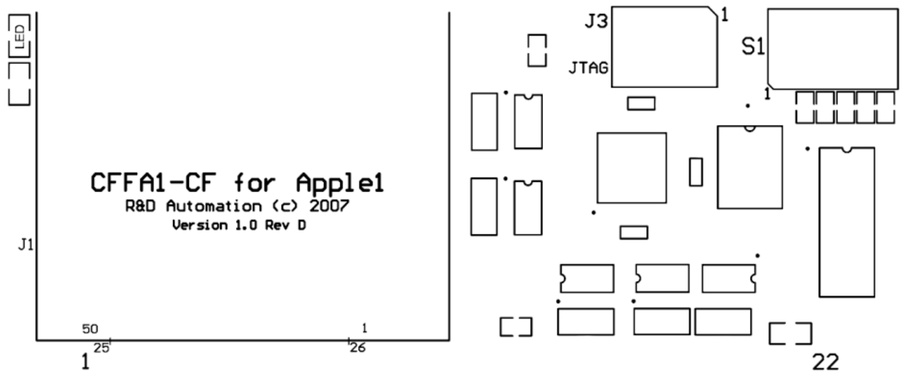


Figure 5: CFFA1 switches and connectors on PCB v1.0 RevD

Name	Purpose
J1	CompactFlash Socket for CF card or Microdrive.
J2 (not shown)	CFFA1 edge card connector. Plugs into an Apple 1 expansion slot. Note: the pin number designations **1** on the left and **22** on the right. On the back of the board you will find **A** and **Z.** Be sure these pin labels correspond when inserting your CFFA card into your Apple1 expansion slot.
J3	JTAG programming header for CPLD U6. Note: Power must be supplied to board during programming with this port. In other words, plug CFFA1 into an Apple 1 while using this port.

Table 2: CFFA1 headers, sockets, and edge connectors

Name	Purpose
S1-1	Unused. Can be in any position
S1-2	Set switch to ON to disable response from the onboard SRAM in the address range of $1000 to $7FFF. This is useful if you are using a Replica1 computer, which already provides SRAM in this address range. Leaving this switch OFF on a Replica1 or any host system with RAM in this address range will result in a bus fight and unpredictable computer operation.
S1-3	Set switch to ON to set pin19 of the CPLD to a logic 0. This pin is currently unused. But may be used in future release of CPLD logic firmware.
S1-4	EEPROM Write Enable. When set to OFF, this switch prevents accidental and intentional writes to the EEPROM. Always leave this switch in the OFF position until you want to change the EEPROM contents.
S1-5	Set switch to ON to enable the 32KB SRAM. This SRAM is mapped into the host computer at $1000 to $8FFF. This cannot be changed without reprogramming the CPLD. You can however prevent the SRAM from responding to addresses $1000 to $7FFF using S1-2. This is useful if you are using a Replica1 computer. Note: the Menu firmware developed for the CFFA1 needs RAM to exist from addresses $8000 to $8FFF on the host computer. If your host computer already has this RAM then disable the CFFA1's onboard SRAM by switching this switch OFF.
S1-6	Reset Enable. Factory Default: jumper installed. Connects Apple bus reset signal to CF device reset. I recommend leaving this jumper in.

Table 3: CFFA1 dip switch meaning

CFFA1 Partition Scheme

Both v1.0 and v1.1 firmware use the same fixed partition scheme supported by the CFFA for Apple *II*. The v1.1 firmware for the CFFA1 only supports access to Drive 0 the first 32MB* partition on your CF card. Future updates to the CFFA1 firmware will support access to FAT16 formatted CF cards.

** Please note that the marketing departments of most device manufactures now define "MB" to mean 1,000,000 bytes instead of the more traditional meaning of 1024 x 1024 = 1,048,576 bytes which the CFFA1 uses.*

Preparing the Storage Device

The menu firmware programmed on the CFFA1's EEPROM has support to read and write a CF card with the ProDOS file system. This means that you can take CF cards from your Apple *II* CFFA and read and write them in your Apple1. Please keep in mind that the CFFA1's firmware is not running ProDOS or any code from ProDOS. It was developed by Dave Lyons to parse the ProDOS file structures. The firmware can load and save Apple1 binary files and Woz Basic files. It can also format the first 32MB partition on your CF card. If you have access to an Apple *II* and a CFFA card (for Apple*II*), you may want to format your CF card in your Apple *II* using a ProDOS formatting tool, like Copy II+ or Davex. This will make your CF card bootable in your Apple II and accessible in your Apple1. If you format your CF card in your Apple1, the CF will not be bootable in your Apple *II*, because the CFFA1 format command does not write the ProDOS boot block on to the CF card.

CF Removability

Although most CF cards are used in a "removable" sense, the CFFA1 interface card does not treat a CF card as a removable device. In other words, if you want to remove the CF card from the CFFA1 interface card, shut down your computer first. Removing the card with the computer's power on will not hurt the CF card, but if you plug the card back in, you will not be able to access the data until you do a complete power cycle of the computer.

Note: Most newer CF cards will be useable again with only a reset* of the Apple1 instead of a power cycle.

**Reset of the CF card will occur when the Apple1 performs a reset, as long as the CFFA1's reset enable switch S1-6 is up (on). A reset only works to reinitialize SOME brands of CF card back into TrueIDE mode. Some brands will only work after a power off/on cycle.*

Devices Compatible with CFFA1

The CFFA1 Interface card was developed using SanDisk CompactFlash cards. A wide variety of brands work with the card, but I can't guarantee compatibility.

Auto-Booting from CF cards?

NO! There is no support in a real Apple1, or authentic clone, to support booting an OS. Remember this was year 1976 and the dawn of the Microcomputer revolution. The Apple1 had 256 bytes of ROM held on two 4bit x 256 one-time PROMs (Programmable Read-Only Memory). This provided just enough space to hold a small monitor and not much else. Only through custom monitor modifications could you create an auto-booting Apple1.

CFFA1 Built-in Menu Details

This section describes in detail the features of the firmware available onboard the CFFA1. The firmware provides both the low-level block access to the cards and a high level menu to allow you to easily access the ProDOS file system. The CFFA1 firmware does not attempt to provide all ProDOS services and should not be thought of as an operating system. It is a menu driven program that can read and write the ProDOS file system.

For more information about the firmware's entry points, see Firmware section in the Advanced Information portion of this manual.

To access the interactive menu on the CFFA1 from Woz monitor simply type:

```
9000R
```

Alternately, if you are mainly using Woz BASIC and you want the Quit 'Q' command to always exit to Woz BASIC, then enter the CFFA1 menu by typing:

```
9003R
```

In either case this will display a menu similar to the one shown below.

```
CFFA1 MENU (1.1)
----------
C - CATALOG      P - PREFIX
L - LOAD         N - NEW DIRECTORY
S - SAVE (BASIC) W - WRITE FILE
R - RENAME       D - DELETE
! - FORMAT       T - TERSE
B - READ BLOCK   M - MEMORY DISPLAY
Q – QUIT
```

Note: While being prompted for input from the various menu commands, the backspace key will work. You do not need to use the "_" (underscore) character like you would in the resident (Woz) monitor.

The current active directory can be changed using the "P" – Prefix command. You may notice a new file type BA1 when viewing a catalog. This has been defined as an Apple1 Basic or "WOZ" Basic file. It is not compatible with Applesoft or Integer basic files on the Apple II. The hex value for this type is: $F1.

Command	Key	API Index	Description
Catalog	**C**	**N/A**	The Catalog command will display the contents of the current directory in a column format reminiscent of the ProDOS "CAT" command.
Details			
After each page of files, the catalog command will pause and wait for the user to enter <SPACE> or <CR> to continue the catalog listing. Alternately you can press <ESC> after each page to abort the catalog command.			

Command	Key	API Index	Description
Load	**L**	**$22, $26**	The Load command is used for loading any type of file into the Apple1 memory.

Details

The firmware handles Woz BASIC (BA1) files specially. All other file types are loaded into RAM in a single block. If the file you are BA1 type file, the load command will automatically load both the main program and the zero page block at: $4A to $FF. Load will prompt you for the file name,

`LOAD FILE:` (enter a standard ProDOS compatible file name)

Next you will be prompted for the starting address to load the file at. If the file has a load address already defined, it will be displayed in the $xxxx field shown below. If that "default" address is satisfactory, press <ENTER> to accept it. Note: if you are loading a BA1 basic file, you will not be prompted for the ADDR, as it is already known.

```
ADDR ($xxxx):
```

If the file loads is successful, the message `00 SUCCESS` will be displayed.

Important: Version 1.0 CFFA firmware loads data in multiples of blocks (512 bytes). So, even for a 1 byte file, 512 bytes are loaded. This limitation may be removed in future version of firmware.

Command	Key	API Index	Description
Save (BASIC)	**S**	**$24**	The save command is used specifically to save WOZ basic programs you have created or modified using the WOZ basic typically located at: $E000.

Details

When this command executes, it will save the two ranges of memory. The first range is the zero page locations used by Basic ($4A to $FF) and the BASIC program defined by HIMEM (4A-4B) and LOMEM (4C-4D). You do not typically need to set HIMEM or LOMEM manually to use this command.

You will be prompted to enter a name for the WOZ Basic program you are trying to save:

```
SAVE:
```

Enter the name you would like to use, if the save operation is successful the message `00 SUCCESS` will be displayed.

IMPORTANT: The Save command will not warn you if you are about to over-write an existing file on the CF card.

Command	Key	API Index	Description
Write	**W**	**$20**	The Write File command is used for saving all file types to the CF card except Basic files.

Details

The write command will prompt you for the starting address and the length of the data you want to write to the CF card. Then it will prompt you for a file name. This file must be a ProDOS compatible name of 15 characters or less.

Enter the starting address to start writing from:

```
WRITE FROM: $
```

Enter the length in bytes in hexadecimal

```
    LENGTH: $
```

Enter the file type. This is a 2 digit hex value of ProDOS file types.

`TYPE (BIN): $` (press <ENTER> to accept BIN type)

Enter the name you want the file stored under in the CF card. Standard ProDOS naming rules apply.

```
      NAME:
```

If the write operation is successful the message `00 SUCCESS` will be displayed.

IMPORTANT: The Write command will not warn you if you are about to over-write an existing file on the CF card.

Command	Key	API Index	Description
Prefix	**P**	**N/A**	Prefix (Change Directory) The Prefix command is the ProDOS way of saying change directory.

Details

To change to the top level (root) directory simply press <enter> when prompted for the directory name. To select a subdirectory enter the name of that directory.

```
PREFIX DIR:
```

Enter the name of the new directory you want to change to.

IMPORTANT*: *CFFA1 firmware v1.0 allows a single level of directories. You can create directories within the root directory, but you can't have directories inside those directories. However, each directory can hold as many files as available disk space allows. The root directory can only hold 51 files.

Command	Key	API Index	Description
New Directory	**N**	**$26**	The New Directory command allows you to create a new directory in the file system.

Details

The command will prompt you for the name of the new directory.

```
NEW DIRECTORY:
```

Enter the name of the directory you want to create.

IMPORTANT: Currently there is a nesting limit of one directory level so all new directories must be created in the root (top level) directory.

Command	Key	API Index	Description
Rename	**R**	**$28**	The Rename command allows you to change the name of a single file or directory.

Details

The command will prompt you for the name of the file to rename. The new name must not already exist in the current directory.

```
RENAME:
```

Enter the name of the file you want to rename.

```
    TO:
```

Enter the new name you would like the file to be called.

Command	Key	API Index	Description
Delete	**D**	**$2A**	The Delete command allows you to delete a file or empty directory from the ProDOS file system.

Details

The command will prompt you for the name of the file or empty directory to be removed. Although the file is removed from the file system, the data blocks that stored the file's content are not overwritten.

```
DELETE:
```

Enter the name of the file you wish to delete.

Command	Key	API Index	Description
Format	**!**	**$2E**	The Format command writes the ProDOS file system to the CF card.

Details

The Format command allows you to initialize the CF card for operation in the CFFA1. Format performs the following actions:

1) Write Zeros to every byte in CF blocks 0 to 64. This will complete erase any existing file system and master boot record on the CF card.
2) Constructs and writes block 2, the main ProDOS directory block.
3) Constructs and writes blocks 3, 4, and 5, which just contains links to other blocks.
4) Constructs and writes the volume bit map starting at block 6. Adding 1 block for every 2MB (or fraction thereof) of volume size. For a 32MB volume, the bitmap takes 16 blocks, from 6 to 21.

```
DESTROY DATA ON DRIVE 0

DESTROY VOLUME: <Volume Name>

   BLOCKS USED: <Volume Size>

ARE YOU SURE? ('YES')
```

Enter “YES” here if you want to format the CF card’s first partition. Entering anything else will abort the format command.

IMPORTANT: If you are planning to use your CF card in your Apple II and your Apple1, I recommend that you format your CF cards in your Apple II. By formatting the card in your Apple II you will be able to boot your CF in your Apple II if you chose. If you only use your CF card in your Apple1, then the menu format command will work fine.

Command	Key	API Index	Description
Terse	**T**	**N/A**	The Terse command simply toggles the menu display on and off.

Details

This is useful due to the slow rate at which characters display on the Apple1 and can save you a lot of time. While in terse mode, if you press a key that is not recognized by the menu firmware, the menu will be displayed once, but you are still in terse mode.

When you press “T” the message **`TERSE ON`** or **`TERSE OFF`** will be displayed to show the new mode. As of firmware v1.1 the terse mode is on by default.

Command	Key	API Index	Description
Read Block	**B**	**N/A**	The Read Block command allows you to read any 512-byte block on the CF card without regard to the type of data it is.

Details

The data block will be displayed in hexadecimal and ASCII format. Once the command is complete the data will be left at address range: $8A00 - $8BFF, until that space is used by another command.

Note: this address is subject to change in future releases of firmware.

Command	Key	API Index	Description
Display Memory	**M**	**N/A**	The Display Memory command allows you to view the content of any memory in the Apple1.

Details

The data is displayed in hexadecimal and ASCII format.

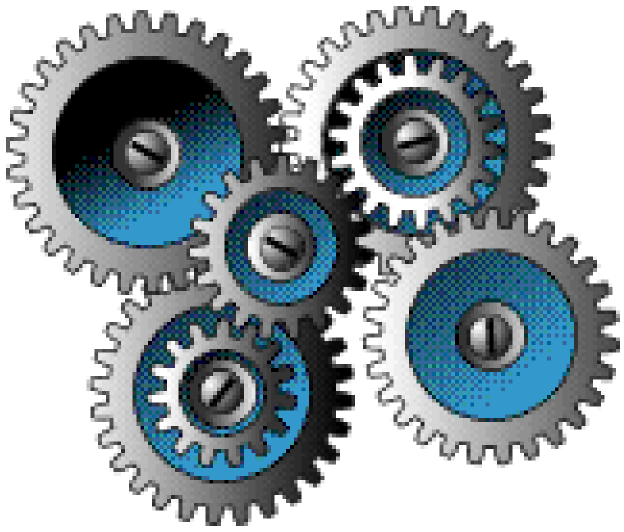
Command	Key	API Index	Description
Quit	**Q**	**N/A**	The Quit command will cause the menu firmware to stop executing.
Details			
Which memory address the Quit command returns to is dependent on which entry point you used to start the menu command. For more information about the firmware's entry points, see *Table 5: CFFA1 Firmware Entry points.*			

Command	Key	API Index	Description
Toggle Serial Mode	**^A**	**N/A**	The serial on / serial off command
Details			
This command toggles the serial mode of the CFFA1 firmware. When the serial mode is enabled, the CFFA1 firmware will look for a 6551 ACIA located on the Replica1 Multi I/O board and use it for basic input and output. When enabled the ACIA is initialized to operate at 19200 baud.			

Command	Key	API Index	Description
Toggle debug Mode	**^D**	**N/A**	Enable/Disable debug output mode
Details			
This command toggles the state of the internal debug enable flag in the firmware. When enabled, the firmware prints information about the blocks written and read from the CF card.			

Command	Key	API Index	Description
Exit	*****	**N/A**	Exit directly to Woz monitor at $FF1F
Details			
This command exits directly to monitor regardless of what entry point was used to enter the menu. Use the Quit command to return to the calling program.			

Advanced Information



Hardware

This section gives detailed information about the hardware used on CFFA1 Interface card.

EEPROM

The CFFA1 firmware is stored on an EEPROM (Electrically Erasable Programmable Read Only Memory) with the designator U7 on the right side of the PCB. The run #1 board used an AT28C64E which had enhanced endurance of 100,000 writes. This EEPROM supported byte mode programming with write times around 200uS/B. The run #2 PCBs use an AT28C64B-15SU. This device uses page mode programming with a page size of 64 bytes. Writing 64 bytes to the EEPROM requires about 10ms. The Flash programming tools have been updated to automatically detect which EEPROM is installed and program it correctly.

EEPROM Address Mapping

Logic on the CPLD places the EEPROM into the address map at $9000 to $AFFF. The last 32 bytes of this range are used to map the CF card registers into making those EEPROM locations inaccessible. Even though this address mapping is statically defined in the CPLD, the CFFA1 also requires that the Apple1's slots 'T' select line be connected to the "A" output of the Apple1 74154 decoder chip.

To make decoding easier, the two halves of the 8K address space were mapped into the Apple1 address space in reverse order. This has no effect when using or re-programming the EEPROM while it is in circuit. The reverse mapping is completely transparent to the user. However, if you were going to program the EEPROM in an external programmer like I was doing early in the project development, you would need to be aware that physical EEPROM addresses $0000 to $0FFF map to $A000 and $1000 to $1FFF map to $9000.

CPLD

The chip U6 is an Atmel ATF1502AS-10AC44 part, which is compatible with an Altera EPM7032STC44-10 CPLD (Complex Programmable Logic Device). It is flash based and can be reprogrammed via the JTAG Port J3. For this you will need Altera compatible programming cable and Atmel's ATMISP program for Windows. When programming the CFFA1 card's CPLD via the JTAG header, the CFFA1 card must be plugged into an Apple 1 bus to supply power to the card. I do not recommend programming the CPLD with a CF card installed.

Programming cables are available from many places, including eBay, for as little as US$15. Search for "ByteBlaster Cable". The older 5 volt cables will work fine. The more expensive low voltage cable is not needed but may also work.

Atmel CPLD Pinout

Figure 6 provides the signal names for version 1.0 of the CFFA1 CPLD firmware (here “firmware” refers to the AHDL logic files used to program the CPLD).

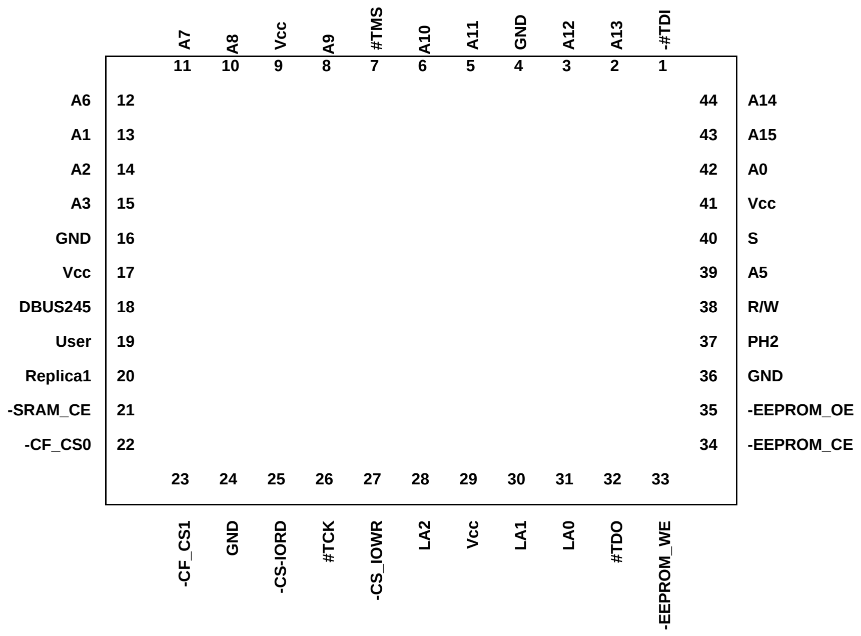


Figure 6: Atmel’s ATF1502AS-10AC44
NOTE: Signal names are specific to CFFA1 logic

CPLD Memory Mapping

The logic resident on the CPLD creates the memory map shown in Table 4. This mapping was programmed into CPLD after the CFFA1 was assembled. The exact logic file used to create this mapping is shown in the next section. Changing this mapping will require reprogramming the CPLD via J3 and a JTAG programmer like an Altera Byte Blaster.

NOTE: Changes to this layout may require changes to the CFFA1's EEPROM menu firmware also.

Address	CFFA1 Memory Mapping	
\$1000 \$7FFF	**U2 - 32KB SRAM**	**Switch S1-2:** enable/disable the region \$1000 - \$7FFF region **Switch S1-5:** enable/disable the entire SRAM chip.
\$8000 \$8FFF		
\$9000 \$9FFF	**U7 – EEPROM**	Always mapped in, when CFFA1 is inserted
\$A000 \$AFFF		Enabled during accesses to \$A000 range as long as the slot signal "T" (memory select line on pin "L") is low. Mapping T to any other address will cause this 4KB portion of firmware to disappear.

Table 4: CPLD Memory Map

CPLD Logic

Listing 1 is the ADHL source file used to create the programmer-ready .jed file needed to program U6. Comments in the file start and end with the % character. This file was compiled using Altera's Quartus 6.0 sp1 Web edition development software and then converted to a .jed file using the Pof2Jed utility. This software is free and can be downloaded from Altera's and Atmel web sites. A free license from Altera is needed to use the Quartus software.

Listing 1: CPLD Logic - AHDL Source

```
%------------------------------------------------------------------------------
 CompactFlash Interface for the Apple 1 computer
 Project Home: http://dreher.net/CFforApple1/
 Project Version 1.0   April, 2007

 Version 0.8  -  Prototype 3: CF card in TrueIDE mode
 Version 0.9  -  Production RevD: EEPROM was being selected at same time
                 as CF card for CF addresses
 Version 1.0  -  Initial Release
------------------------------------------------------------------------------%

SUBDESIGN CFLogic
(
  A0, A1, A2, A3, A5, A6, A7, A8, A9    :INPUT;
  A10, A11, A12, A13, A14, A15          :INPUT;
  PH2, /EN_T, R/W, User, /Replica1      :INPUT;

  /SRAM_CE                              :OUTPUT;
  /CF_CS0, /CF_CS1, /CF_IORD, /CF_IOWR  :OUTPUT;
  /EEPROM_WE, /EEPROM_CE, /EEPROM_OE    :OUTPUT;
LA0, LA1, LA2, /DBUS245                 :OUTPUT;
)

VARIABLE
/EEAddr, /EEAuxAddr                     :NODE;
/SRAMMainAddr, /SRAMAuxAddr             :NODE;
/CFAddresses                            :NODE;
/RESET_MASK, /SET_MASK, CS_MASK         :NODE;
LA3                                     :NODE;

BEGIN

%------------------------ 8K EEPROM Control logic -------------------------------%

% EEPROM Address: 4K direct mapped to $9000 to $9FFF %
 /EEAddr = !A15 # A14 # A13 # !A12 # !/CFAddresses;

% EEPROM Address: 4K mapped to T select line at $A000 to $AFDF as long as Select
  line T is connected to A on Apple1%
% Note: The CF card is mapped into $AFF0-$AFFF and $AFE0-$AFEF (It is duplicated
  twice because A4 is not available for address decode) %
 /EEAuxAddr = /EN_T # (!A15 # A14 # !A13 # A12) # !/CFAddresses;

% EEPROM Chip Enable %
 /EEPROM_CE = (/EEAddr !$ /EEAuxAddr) # !PH2;


% EEPROM Output Enable %
 /EEPROM_OE = !R/W # /EEPROM_CE;

% EEPROM Write Enable %
 /EEPROM_WE  = R/W # /EEPROM_CE;

%------------------------- 32K SRAM Control logic -------------------------------%

% 1) Decode SRAM address at: $1000 to $7FFF. Assumes there is 4K DRAM already on
     board Apple 1 at $0000%
% /Replica1 =0 to disable SRAM on CFFA1.%
 /SRAMMainAddr = A15 # (!A14 & !A13 & !A12);

% 2) Decode SRAM address at $8000 to $8FFF %
```

```
 /SRAMAuxAddr = !A15 # A14 # A13 # A12;

% If Replica1 then disable SRAM from $1000 to $7FFF. %
% If Apple1   then  enable SRAM from $1000 to $7FFF. %
% In both Apple1 and Replica1 always map 4K sram at $8000 to $8FFF %
 /SRAM_CE = ((/SRAMMainAddr  # !/Replica1) !$ /SRAMAuxAddr) # !PH2;

% Pass Address line through, for now, unlatched %
  LA0 = A0;
  LA1 = A1;
  LA2 = A2;
  LA3 = A3;

%------------------------------------------------------------------------------%
% Fix for SanDisk Family of CompactFlash drives. True IDEmode is not quite     %
% True! The idea here is to mask the read cycle that proceeds all write cycles %
% because the read cycle was confusing the Sandisk                             %
%------------------------------------------------------------------------------%
% SetMask = $AFF1   %
 /SET_MASK = /CFAddresses # (LA3 # LA2 # LA1 # !LA0);
% ResetMask = $AFF2 %
 /RESET_MASK = /CFAddresses # (LA3 # LA2 # !LA1 # LA0);
  CS_MASK  = SRFF(!/SET_MASK, !/RESET_MASK, PH2, VCC, VCC);

%------------------------------ CF Control logic ---------------------------------%
% decode CF card address range at $nFF0 to $nFFF. Because Address lines A4 is not
  available to this logic, the CF address space will repeat from nFE0-nFEF, where
  n is the 4K space mapped to select line S  %
 /CFAddresses = /EN_T  # !A11 # !A10 # !A9 # !A8 # !A7 # !A6 # !A5;

% CF Chip Select0 line: addresses nFF8 to nFFF %
 /CF_CS0 = /CFAddresses # !LA3 # (CS_MASK & R/W);
% CF Chip Select1 line: addresses nFF6 to nFF7 %
 /CF_CS1 = /CFAddresses # (LA3 # !(LA1 & LA2)) # (CS_MASK & R/W);

 /CF_IORD  = !R/W # /CFAddresses # !PH2;
 /CF_IOWR  =  R/W # /CFAddresses # !PH2;

 /DBUS245 = /EEPROM_CE & /SRAM_CE & /CF_CS0 & /CF_CS1;

END;
```

Firmware

This section gives detailed information about the EEPROM based code (firmware) shipped with the CFFA1 Interface card.

User Accessible Firmware Entry Points

The firmware resident on the CFFA1 EEPROM provides a variety of firmware entry points for the user, as shown in Table 5. The first two entry addresses, $9000 and $9003 will be the most useful to a typical CFFA1 user. The key difference between these entry points is in what happens when you select the Q – Quit option on the menu.

Address	Purpose
$9000	Standard Menu Firmware entry point. Menu option 'Q' (Quit) exits back to Woz monitor at $FF1F.
$9003	Alternate Menu Firmware entry point. Menu option 'Q' (Quit) exits back to Woz BASIC (if loaded) at address $E2B3. If basic is not resident the Apple1 will likely crash. Note: The standard Woz BASIC entry point is $E000, which clears out any existing BASIC program. The alternate BASIC entry point $E3B3 is useful if you want to return to Woz BASIC and not erase any existing BASIC program. If writing a Woz BASIC program from scratch (not loading one from the CF card) you should enter BASIC using E000R entry point. Then use CALL –24000 to reach the CFFA menu.
$9006	Alternate Menu Firmware entry point. Q (Quit) will RTS back to caller. This entry point is designed to be used via JSR $9006.
$9009	Entry point for low-level CF block driver code. This code must be called with a JSR $9009, and specific Zero-page locations must be set up before being called. See firmware source for additional information
$900C	Entry point for access to CFFA1 firmware Application Programming Interface (API). A simple set of functions used by the CFFA firmware is exposed via this entry point. Prior to calling this entry point, various zero page locations must be initialized and the X register must be loaded with the requested API function to be executed.
$9012	Alternate Menu Firmware entry point. Enter Menu using Serial IO on Replica 1 multi-I/O board, 19200 baud, ACIA at $C300. entry point requires v1.1 or later firmware.
$A240	Jumps to the $9003 entry point. This entry point can be used from Woz BASIC by typing CALL –24000. This is easy to remember and when you quit the menu, you will return to Woz BASIC via the $E3B3 BASIC entry point.
$AF80	"Bootstrap" entry point: Reads block 0 from the CF card to SRAM starting at $0E00. If successful, then jumps to $E00. If the load fails jumps back to Woz Monitor at $FF1F. This entry point requires v1.1 or later firmware.

Table 5: CFFA1 Firmware Entry points.

Firmware Updates

All new firmware updates will be made available at the CFFA1 web site: <http://dreher.net/CFforApple1/> in the "downloads" section. They will be available as both ASCII and binary files. Andy McFadden's CiderPress program for Windows makes copying files to your ProDOS formatted CF card easy. In the rare case that your CFFA1 firmware is corrupted you will either need to find the corrupt bytes and fix them, or use a serial port to your Apple1 (if available) to send the firmware .HEX file to the Woz monitor as if you typing the code in byte by byte.

Below are detailed instructions describing how to load v1.1 firmware into the CFFA1's EEPROM using a FLASHERV1.1.BIN utility. These instructions assume you have a working copy of Andy McFadden's CiderPress for Windows on a PC with a CF card reader recognized by Windows.

Loading CFFA1 firmware using CiderPress on Windows

This section describes how to load two files from your PC onto a ProDOS formatted CF card using CiderPress for Windows. This assumes you already have a ProDOS formatted CF card ready to go.

- Insert a ProDOS formatted CF card into your Windows PC.

*Note: Windows may offer to format your card. Do **NOT** format the CF card but instead just cancel the format window.*

- Run Andy McFadden's CiderPress v3.0.1. Older version will probably work too.

- Click File -> Open Volume

- Under the "Show" pull-down select "Physical Disks"

- Uncheck the "Open as Read-Only" check box to allow writing.

- Select the physical disk that corresponds to the CF card.

Note: Be careful here! Don't select the wrong disk. Look closely at the size of the media and make sure to select the correct physical disk.

- Click OK. The disk should be opened and the contents of the ProDOS volume should be displayed.

- Right click on the disk volume label at the top of the content list and select "Add Files". The volume label usually starts with a "**:**".

Note: If "Add Files" is grayed-out then you forgot to uncheck the "Open as Read-Only" check box above.Close the volume and start over.

- From the "Add Files…" dialog, find and select the file FLASHERV1.1.BIN from your hard drive. The default settings for the other options on this dialog will work fine. Click Accept.

- Now that the file is on the CF card, edit its attributes by right clicking on FLASHERV1.1.BIN and selecting "Edit Attributes...".

- On the "Edit Attributes" dialog, change the "Aux Type" (hex):" entry from 0000 to 1000. Change the file type to be $06, if it is not already. This updates the Aux type information in the files directory entry. Now the firmware will know where to load the file.

- From the "Add Files…" dialog, find and select the file CFFAROMV1.1.BIN from your hard drive. The default settings for the other options on this dialog will work fine. Click Accept.

- Now that the file is on the CF card, edit its attributes by right clicking on CFFA1ROMV1.1.BIN and selecting "Edit Attributes...".

- On the "Edit Attributes" dialog, change the "Aux Type" (hex):" entry from 0000 to 2000. Change the file type to be $06, if it is not already. This updates the Aux type information in the files directory entry. Now the firmware will know where to load the file.

- Close the volume by selecting "File -> Close" from the menu.

- Remove the CF card from your PC.

Programming the EEPROM on the CFFA1

This section describes how to program EEPROM U7 with new firmware. This assumes your CFFA1 has working firmware on it already and you can access the menu features of that firmware.

- Turn off your Apple1 computer.

- Insert the CFFA1 card into your Apple1 computer.

- Insert CF card into the CFFA1.

- Power on your Apple1 computer.

- Run the existing CFFA1 menu firmware:

```
 9000R
```

- If the flasher and firmware image are located in a subdirectory on the CF card use the Prefix command to change to that directory: Example: **P**refix: **UTILITIES** <enter>. If the files are in the CF card's root directory, then skip this step.

- Select the **L**oad option to load the flashing utility: **FLASHERV1.1.BIN**

- Accept the default address by pressing <Enter> key if it is ($1000). If it is not, enter **1000** <Enter>.

- Select the Load option to load the firmware image: **CFFAROMV1.1.BIN**

- Accept the default address by pressing <Enter> key if it is ($2000). If it is not, enter 2**000** <Enter>.

- Press **Q** to quite the CFFA1 firmware back to Apple1 monitor.

- Type: **1000R** (to run the FLASHER program)

- Follow instructions on screen. They will instruct you to flip switch S1-4 up, and press <space> to begin programming and then flip switch S1-4 back down. The entire programing process takes about 3 seconds.

If all has gone well, the FLASHER program should program and verify the new firmware. If there is a verify failure, check the dip switch S1-4 to be sure it is set to enable writes to the EEPROM.

- Flip switch S1-4 down to write protect the EEPROM when programming is done.

- Your CFFA1 should be ready to use

Optionally, instead of loading the two files FLASHERV1.1.BIN and CFFAROMV1.1.BIN you can use a file that combines both of those files into a single file called: UPDATERV1.1.BIN. This file should be loaded at $E00, and executed at $1000 with 1000R. In either case the same data ends up being flashed onto the EEPROM.

Contributing Firmware to the CFFA1 Project

There are two ways in which you can contribute to the firmware portion of this project.

1. You can send me your ideas for improvements that I will consider integrating into a future firmware version.
2. Write your own firmware/driver for my hardware and send me the working source code and binary, and I may post it on my web site under a contributors' section.

CFFA1 CompactFlash Memory Mapped I/O

Table 6 shows all of the CF I/O addresses decoded by the CFFA1 Interface card. These addresses are used to interface a CF card's task register file to the Apple's bus. There are also two special soft switches used to inhibit CPU read cycles from confusing CF cards during block write routines. These were added because some CF cards are not tolerant of read-cycles during PIO block writes in "TrueIDE" mode.

NOTE: Because the CPLD on the CFFA1 doesn't have address line A4 connected to it, the memory map in table 6 below repeats at addresses $AFE0 to $AFEF also. To maintain compatibility with any future version of the CFFA1 don't use this mirrored range of addresses.

Apple 1 Address	Name Used in Source Code	Read/ Write	Description
$AFF0	ATAData	R/W	This register is used to transfer data between the host and the attached device.
$AFF1	SetCSMask	R/W	Special soft switch to **disable** CS0 & CS1 signaling to attached device during 65C02 read cycles that always precede write cycles
$AFF2	ClearCSMask	R/W	Special soft switch to **enable** CS0 & CS1 signaling to attached device during 65C02 read cycles that always precede write cycles
$AFF3			Unused
$AFF4			Unused
$AFF5			Unused
$AFF6	ATADevCtrl	W	This register is used to control the device's interrupt request line and to issue an ATA soft reset to the device.
$AFF6	ATAAltStatus	R	This register returns the device status when read by the host. Reading the ATAAltStatus register does NOT clear a pending interrupt. (NOTE: CFFA1 does not use interrupts)
$AFF7			Unused
$AFF8			Unused
$AFF9	ATAError or ATAFeature	R	This register contains additional information about the source of an error when an error is indicated in bit 0 of the ATAStatus register.
$AFFA	ATASectorCnt	R/W	This register contains the number of blocks of data requested to be transferred on a read or write operation between the host and the device. If the value in this register is zero, a count of 256 sectors is specified.
$AFFB	ATA_LBA07_00	R/W	This register contains the starting sector number or bits 7-0 of the Logical Block Address (LBA) for any device access for the subsequent command.
$AFFC	ATA_LBA15_08	R/W	This register contains the low order 8 bits of the starting cylinder address or bits 15-8 of the Logical Block Address.
$AFFD	ATA_LBA23_16	R/W	This register contains the high order bits of the starting cylinder address or bits 23-16 of the Logical Block Address.
$AFFE	ATA_LBA27_24	R/W	This register is used to select LBA addressing mode or cylinder/head/sector addressing mode. Also bits 27-24 of the Logical Block Address.
$AFFF	ATACommand	W	A write to this register will issue an ATA command to the device.
$AFFF	ATAStatus	R	This register returns the device status when read by the host. Reading the Status register does clear a pending interrupt. (NOTE: CFFA1 does not use interrupts)

Table 6: CF card address space defined by the CFFA1 Interface card

Application Programming Interface (API)

A file called CFFA1_API.s is available to programmers who would like to gain access to the CFFA1 firmware from their own programs. The file contains equates a programmer will need, including the entry point addresses for the various CFFA1 firmware routines.

Listing 2: Programmers API Info

```
;-----------------------------------------------------------------------------
; CFFA1_API.s    10-Feb-2013
; Released with CFFA1 firmware v1.1.
;
; Equates for calling the CFFA1 API.
;-----------------------------------------------------------------------------

CFFA1_ID1           = $AFDC   ; contains $CF when CFFA1 card is present
CFFA1_ID2           = $AFDD   ; contains $FA when CFFA1 card is present

;
; FirmwareVersion $01 = CFFA1 firmware 1.0
; FirmwareVersion $02 = CFFA1 firmware 1.1
;
FirmwareVersion     = $02

;-----------------------------------------------------------------------------
; Entry points to the CFFA1 firmware:
;
; MenuExitToMonitor
;   JMP here to display the CFFA1 menu.
;   Quit puts the user into the monitor.
;
; MenuExitToBASIC
;   JMP here to display the CFFA1 menu.
;   Quit puts the user into BASIC.
;
; Menu
;   JSR here to display the CFFA1 menu.
;   Quit returns control to your code.
;
; CFBlockDriver
;   JSR here to read or write a block, after setting up pdCommandCode
;   and other inputs (see below).
;   Result:  CLC, A = 0
;            SEC, A = error code
;
; CFFA1_API
;   JSR here to call one of many functions provided by the firmware.
;   See "Function selectors for CFFA1_API" below.
;
;-----------------------------------------------------------------------------
MenuExitToMonitor   = $9000
MenuExitToBASIC     = $9003
Menu                = $9006
CFBlockDriver       = $9009
CFFA1_API           = $900C
MenuWithSerialIO    = $9012
```

```
;-------------------------------------------------------------------------------
; Inputs for CFBlockDriver - ProDOS block interface locations
;-------------------------------------------------------------------------------
pdCommandCode       = $42     ; see below
pdUnitNumber        = $43     ; always set this to 0 for firmware 1.0
pdIOBufferLow       = $44
pdIOBufferHigh      = $45
pdBlockNumberLow    = $46
pdBlockNumberHigh   = $47

;
; Values for pdCommandCode
;
PRODOS_STATUS       = $00
PRODOS_READ         = $01
PRODOS_WRITE        = $02
PRODOS_FORMAT       = $03


;-------------------------------------------------------------------------------
; Function selectors for CFFA1_API.
;
; Load one of these values into X:
;
;    ldx #CFFA1_xxxxx
;    jsr CFFA1_API
;
; Result:  CLC, A = 0
;          SEC, A = error code
;
; Certain functions have additional outputs, as described below.
;
;-------------------------------------------------------------------------------
;
; CFFA1_Version:
;   Output: X = current firmware version
;           Y = oldest compatible firmware version
;
; CFFA1_Menu:
;   Result: Runs the CFFA1 menu and returns when the user chooses Quit.
;
; CFFA1_DisplayError:
;   Input:  A = an error code
;   Result: Prints out a carriage return, the 2-digit hex error code,
;           and a description of that error, if available.
;
; CFFA1_OpenDir:
;   Input:  None (operates on the current prefix directory)
;   Result: Prepares for one or more calls to ReadDir.
;
; CFFA1_ReadDir:
;   Setup:  You have to call OpenDir before calling ReadDir.
;   Result: If no error, EntryPtr points to the next occupied directory entry.
;
; CFFA1_FindDirEntry:
;   Input:   Filename = name to search for
;   Result:  If no error, EntryPtr points at the found item's directory entry.
;
; CFFA1_WriteFile:
```

```
;   Input:  Filename = name for new file (will be replaced if it already exists)
;           Destination = starting address
;           FileSize = number of bytes to write
;           Filetype = type for new file
;           Auxtype = auxiliary type for new file
;
; CFFA1_ReadFile:
;   Input:  Filename = file to read into memory
;           Destination = starting address ($0000 to use the file's Auxtype value)
;
; CFFA1_SaveBASICFile:
;   Input:  Filename
;
; CFFA1_LoadBASICFile:
;   Input:  Filename
;
; CFFA1_Rename:
;   Input:  OldFilename = original name
;           Filename = new name
;
; CFFA1_Delete:
;   Input:  Filename = file or empty directory to delete
;
; CFFA1_NewDirectoryAtRoot:
;   Input:  Filename = name for new directory
;
; CFFA1_FormatDrive:
;   Input:  Filename = name for new volume
;           A = drive number (always set to 0 for firmware 1.0)
;           Y = $77 (just to help avoid accidental formatting)
;   Result: Disk volume is erased and given the specified name.
;
;------------------------------------------------------------------------------
CFFA1_Version            = $00
CFFA1_Menu               = $02
CFFA1_DisplayError       = $04

CFFA1_OpenDir            = $10
CFFA1_ReadDir            = $12
CFFA1_FindDirEntry       = $14

CFFA1_WriteFile          = $20
CFFA1_ReadFile           = $22
CFFA1_SaveBASICFile      = $24
CFFA1_LoadBASICFile      = $26
CFFA1_Rename             = $28
CFFA1_Delete             = $2A
CFFA1_NewDirectoryAtRoot = $2C
CFFA1_FormatDrive        = $2E


;------------------------------------------------------------------------------
; Zero-page inputs and results for API functions
;
; Filename and OldFilename point to strings that begin with a length byte (from
; 1 to 15), and each character must have its high bit off.  For example:
;
;    Filename   = $80     $280: 05  48  45  4C  4C  4F
;    Filename+1 = $02               'H'  'E' 'L' 'L' 'O'
```

```
;-------------------------------------------------------------------------------
Destination         = $00                   ; 2 bytes
Filename            = Destination+2         ; 2 bytes
OldFilename         = Filename+2            ; 2 bytes
Filetype            = OldFilename+2         ; 1 byte
Auxtype             = Filetype+1            ; 2 bytes
FileSize            = Auxtype+2             ; 2 bytes
EntryPtr            = FileSize+2            ; 2 bytes


;-------------------------------------------------------------------------------
;
; ProDOS low-level return codes
;
;-------------------------------------------------------------------------------
PRODOS_NO_ERROR       = $00    ; No error
PRODOS_BADCMD         = $01    ; Bad Command (not implemented)
PRODOS_IO_ERROR       = $27    ; I/O error
PRODOS_NO_DEVICE      = $28    ; No Device Connected
PRODOS_WRITE_PROTECT  = $2B    ; Write Protected
PRODOS_BADBLOCK       = $2D    ; Invalid block number requested
PRODOS_OFFLINE        = $2F    ; Device off-line
;
; High-level return codes
;
eBadPathSyntax        = $40
eDirNotFound          = $44
eFileNotFound         = $46
eDuplicateFile        = $47
eVolumeFull           = $48
eDirectoryFull        = $49
eFileFormat           = $4A
eBadStrgType          = $4B
eFileLocked           = $4E
eNotProDOS            = $52
eBadBufferAddr        = $56
eBakedBitmap          = $5A
eUnknownBASICFormat   = $FE
eUnimplemented        = $FF


;-------------------------------------------------------------------------------
; ProDOS directory entry structure offsets
;-------------------------------------------------------------------------------
oFiletype           = $10
oKeyBlock           = $11
oBlockCount         = $13
oFileSize           = $15
oCreateDateTime     = $18
oVersion            = $1C
oMinVersion         = $1D
oAccess             = $1E
oAuxtype            = $1F
oModDateTime        = $21
oHeaderPointer      = $25

oDirLinkPrevious    = $00
oDirLinkNext        = $02
oVolStorageType     = $04
```

```
oVolVersion          = $20
oVolAccess           = $22
oVolEntryLength      = $23
oVolEntriesPerBlock  = $24
oVolFileCount        = $25
oVolBitmapNumber     = $27
oVolTotalBlocks      = $29

;
; ProDOS Storage types
;
kSeedling            = $10
kSapling             = $20
kTree                = $30
kExtended            = $50
kDirectory           = $D0
kSubdirHeader        = $E0
kVolume              = $F0
kStorageTypeMask     = $F0

;
; Filetypes
;
kFiletypeText        = $04
kFiletypeBinary      = $06
kFiletypeDirectory   = $0F
kFiletypeBASIC1      = $F1
kFiletypeBAS         = $FC
kFiletypeSYS         = $FF

;------------------------------------------------------------------------------
; end of CFFA1_API.s
;------------------------------------------------------------------------------
```

Contact Information

The following is a list of information resources for the CFFA1 Interface Card project. If you have questions, comments, or problems to report, please contact me using one of the methods listed below.

CFFA1 Web Site

The CFFA1 web site is located at: <http://dreher.net/CFforApple1/>. There you will find any new firmware revisions, project revisions, and general project status information.

Internet E-Mail

I can be reached via E-mail at: rich@dreher.net.

If you are reporting a problem, please use “CFFA1 problem” or similar as your subject line. In your E-mail you should include the firmware version number, which can be determined by looking right after the words “CFFA MENU” when you run the menu firmware at $9000. Also, if possible, describe the conditions necessary to cause the problem.

CFFA1 Message Web Forum

To post a message in the forum, simply browse to: <http://dreher.net/phpBB/>. The CFFA1 message forum is a good place to post technical problems and solutions. Other users may have had similar problems and know of a possible solution. You will have to register before you can post to the forum the first time. I will have to manually approve your registration request. This is because of all the spam bots on the net. If you don’t get registered within a day, email me and ask to approve you.

Acknowledgements

I would like to thank the following people for providing help on this project:

Sherry Dreher

Dave Lyons

Vince Briel

Philip Lord