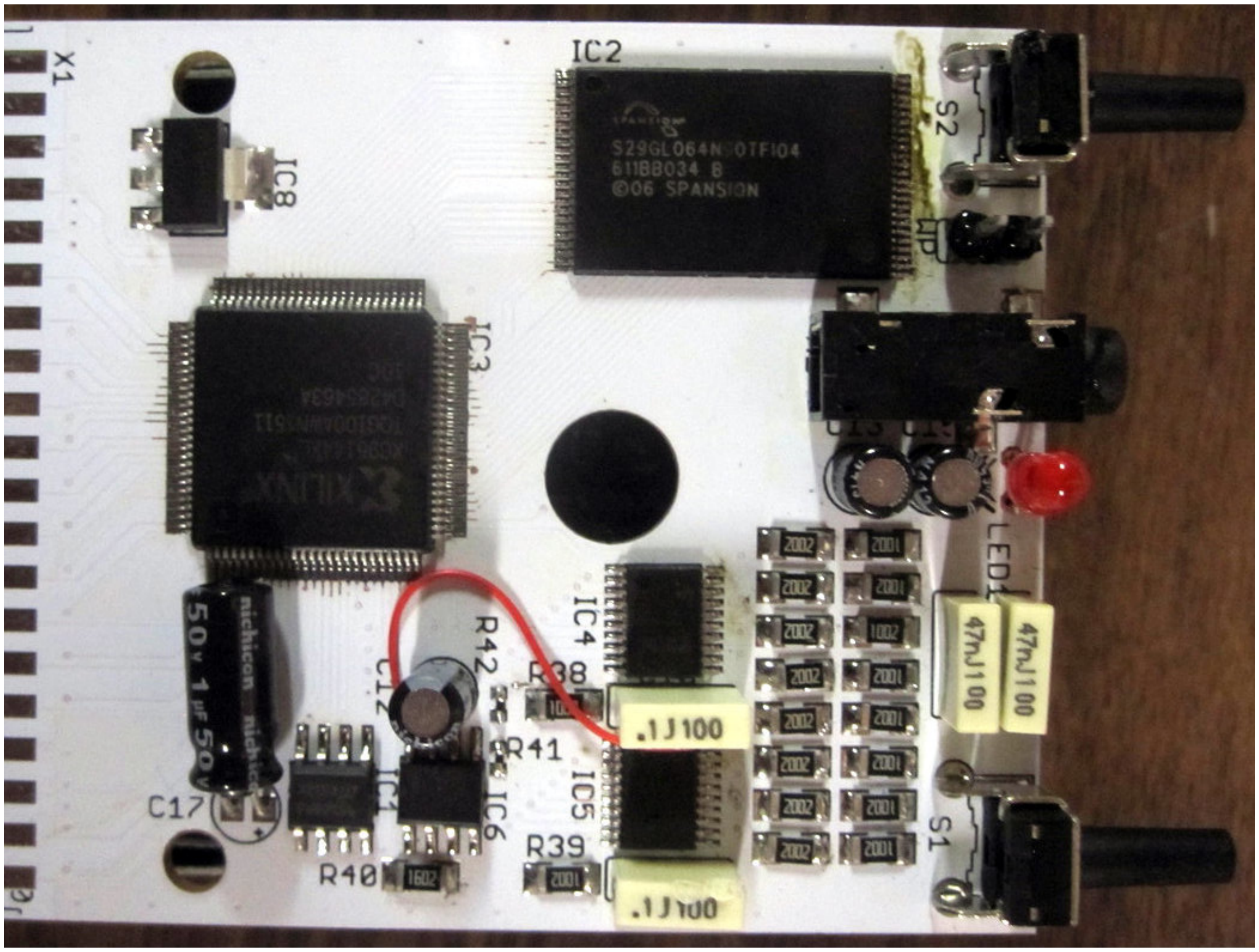
RETRO INNOVATIONS

Coco Flash User Guide


IC2
S29GL064N90TFI04
611BB034 B
©06 SPANSION
IC8
IC3
XILINX
IC4
IC5
IC6
C17
R38
R39
R40
R41
R42
.1J100
47nJ100
LED1
S1
S2

RETRO INNOVATIONS

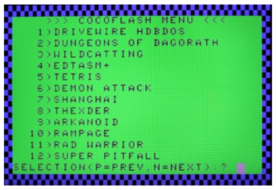
CoCo Flash User Guide

Table of Contents

Description of the CoCo Flash Cartridge

This section describes what the capabilities of the CoCo Flash are and how it can be used.

The CoCo Flash is a programmable flash ROM cartridge. This cartridge can be used to store multiple game and DOS cartridge ROMs which can then be selected from the menu to be used. The Coco Flash can contain up to 2,048 4k ROMs at once. The sizes of these ROMs can range from 2k to 256k. Banked ROMs larger than 32k games such as Robo Cop and Predator are supported. 32k ROMs are capable of being installed by either banking or the CoCo 3 only method of mapping a full 32k ROM. The CoCo Flash is capable of being used as an Orchestra90 sound cartridge and is fully compatible with the original Orchestra 90 Cartridge, which was manufactured by Radio Shack. The CoCo Flash includes a user customizable menu which is written in BASIC. This allows the menu to be edited to list your own choice of ROMs. The CoCo Flash is also pre-loaded with 5 ROMs. The Coco Flash has a copy of HDBDOS, which is configured to auto detect a CoCo 1/2 or CoCo 3 and is configured to use a DriveWire server. The Orchestra 90 ROM is pre-loaded into the CoCo Flash as well. The CoCo Flash also contains the games “Doodle Bug”, “Buzzard Bait”, and “Temple of ROM”.



```
    >>> COCOFLASH MENU <<<
  1)DRIVEWIRE HDBDOS
  2)DUNGEONS OF DAGORATH
  3)WILDCATTING
  4)EDTASM+
  5)TETRIS
  6)DEMON ATTACK
  7)SHANGHAI
  8)THEXDER
  9)ARKANOID
 10)RAMPAGE
 11)RAD WARRIOR
 12)SUPER PITFALL
SELECTION(P=PREV,N=NEXT):?
```

How to Customize The CoCo Flash Menu

The CoCo Flash Menu is written in basic. The menu is customizable to users by loading the menu through the command:

```
LOAD "MENU.BAS"
```

After you have done this you can edit data statements from lines 10000 - 10998 to add your program info into the menu. The info you add for your program will be in this format:

"NAME OF PROGRAM",*bank#*,*type#*

- **NAME OF PROGRAM** represents the title or name of the added program as it will appear on the menu screen.
- ***bank#*** represents the first bank number used for the program you are adding. If you are adding a 32k program the number should be the first bank being used in the second chunk of 16k.
- ***type#*** represents the type of program you are adding. The three common options that can be used are 2, 0, and 34. The number 2 means that a game program which uses the IRQ to start is being added to the menu. If 0 has been entered then a DOS program has most likely been added. DOS programs have DK in the first two bytes of the ROM image. When the color computer starts it checks the cartridge rom memory location to see if the first two characters are DK and if they are it launches the contents of the cartridge as a basic DOS extension which adds commands to basic. Lastly, if the value is 34 then the cartridge has been split into multiple 16k cartridges. In other words, programs that are 128k, 64k, or some 32k cartridges use the 34 value since the largest CoCo ROM bank on a CoCo 1 or 2 that can be used is 16k at a time.

Once you have finished adding the entry to the menu, save the program using the command:

```
SAVE "MENU.BAS"
```

Now, because this is a basic program and ROMs only allow machine code programs to be stored in them, you must convert the basic program to machine code. Fortunately, the menu program has a routine built into it to do this. To convert the menu to machine code use the command:

```
RUN 60000
```

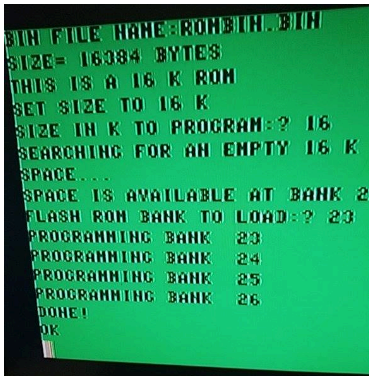
This will create a file named “BASROM.BIN” that can be programmed into bank 0 of the CoCo Flash. Please note that banks 0 and 1 are both reserved for the menu. The next section will discuss how to program files into the CoCo Flash.

NOTE: when programming “BASROM.BIN” the programming software will warn you that you will be overwriting bank 0 and ask you if you are sure you want to erase banks 0 and 1, type yes for the programming to continue.

Programming the CoCo Flash

When programming the CoCo Flash in a system using a Multi-Pak interface, you should insert the CoCo Flash into slot 1 and if you are using a floppy controller or CoCo SDC, that should be inserted into slot 4 of the Multi-Pak interface. If you do not have a Multi-Pak, you may use the built in HDBDOS with DriveWire to access the programming software. To program in a new ROM into the CoCo Flash, use the command:

RUN "PRGFLASH.BAS"



```
BIN FILE NAME:ROMBIN.BIN
SIZE= 16384 BYTES
THIS IS A 16 K ROM
SET SIZE TO 16 K
SIZE IN K TO PROGRAM:? 16
SEARCHING FOR AN EMPTY 16 K
SPACE....
SPACE IS AVAILABLE AT BANK 2
FLASH ROM BANK TO LOAD:? 23
PROGRAMMING BANK  23
PROGRAMMING BANK  24
PROGRAMMING BANK  25
PROGRAMMING BANK  26
DONE!
OK
```

The first thing the program will ask for is the name of the ROM image you are trying to add to the CoCo Flash. This image must be a .bin file that loads at hex address &H4000. If the file is not in this format it must be converted. This will be discussed in the next section. The next thing the program will do is show the size of the ROM in bytes, after doing this it will calculate what the size of the ROM is in kilobytes and ask you what the size in kilobytes to program. The program will then search for an empty space the size you entered. The bank in which the space is available will be displayed and you will then be asked where you wish to load the ROM. Normally you will enter the number that that is displayed. However, there are a few exceptions to this. If you are rewriting the menu then you will want to enter 0. Another exception is if you are overwriting an existing ROM then you might want to enter the value that was used before. Once a bank value has been entered the program will proceed to program the ROM into the declared bank and if the ROM is larger than 4 k it will use the next few banks as well. For example, a 16k ROM will take 4 banks to program.

If you are using a Multi-Pak interface, after the programming completes you should type the command below:

POKE &HFF7F,&H33

This will reactivate the disk controller (or CoCo SDC) in slot 4.

ROM to BIN Conversions

As we have mentioned previously in order to program a ROM imagse into the CoCo Flash it needs to be a .bin file with a starting address of &H4000. Most ROMs won't be in this format so you will have to convert them. You may also have the issue of how to get the .bin file onto a disk image a color computer can read. Please note that if your ROM image is larger than 16k then you need to split it into 16k or smaller chunks. To split your file run this command:

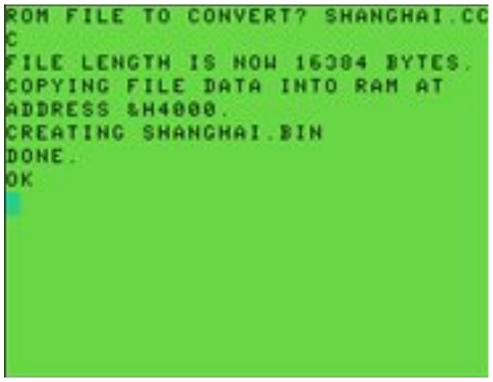
RUN "SPLIT.BAS"

This will prompt you for the given file to split when run. Each new chunk that results from a split has a unique extension to the new file. The first unique extension is 000, then 001, then 002 and so on. If the original file is a 32k .ccc file then you will need to program the 001 file before the 000 file.

The next step is to convert the raw files into a bin so that the Coco 3 can load the file. To do this run this:

RUN "CONV2BIN.BAS"

This will convert your file into a .bin. Once this is done you should pay attention to the newly made .bin file. This is because all the split files will have the same names once they have been converted. This will cause the new .bin file to overwrite the last file if you are converting a split file. There are two good ways to make sure each section of the split file is programed into the CoCo Flash. The first option is to program the file immediately after it has been converted and then to overwrite the file with the next split file. The other option is to rename the file so that it is not overwritten.



```
ROM FILE TO CONVERT? SHANGHAI.CC
C
FILE LENGTH IS NOW 16384 BYTES.
COPYING FILE DATA INTO RAM AT
ADDRESS &H4000.
CREATING SHANGHAI.BIN
DONE.
OK
```

CoCo Flash bank organization

When you overwrite a bank other banks will be erased. The table below shows how the banks are organized and which banks would be erased. All the banks in the same sector or essay address will be erased when any bank in that sector is overwritten. Note that you can safely write to any empty bank that does not have any data in it without affecting any other banks. For example, if banks 12, 21 and 22 have data in them and you write to bank 12 banks 21 and 22 as well all other banks in sector 8 will be erased. However, if banks 21 and 22 have data, but bank 12 is empty, you can write to bank 12 safely. In other words, you can always write to an empty bank.

Bank	sector	sector size	Bank	sector	sector size	Bank	sector	sector size
						~ pattern repeats ~		
2044	SA0	8k	12	SA8	64k	2028	SA134	64k
2045	SA0	8k	13	SA8	64k	2029	SA134	64k
2046	SA1	8k	14	SA8	64k	2030	SA134	64k
2047	SA1	8k	15	SA8	64k	2031	SA134	64k
0	SA2	8k	16	SA8	64k	2032	SA134	64k
1	SA2	8k	17	SA8	64k	2033	SA134	64k
2	SA3	8k	18	SA8	64k	2034	SA134	64k
3	SA3	8k	19	SA8	64k	2035	SA134	64k
4	SA4	8k	20	SA8	64k	2036	SA134	64k
5	SA4	8k	21	SA8	64k	2037	SA134	64k
6	SA5	8k	22	SA8	64k	2038	SA134	64k
7	SA5	8k	23	SA8	64k	2039	SA134	64k
8	SA6	8k	24	SA8	64k	2040	SA134	64k
9	SA6	8k	25	SA8	64k	2041	SA134	64k
10	SA7	8k	26	SA8	64k	2042	SA134	64k
11	SA7	8k	27	SA8	64k	2043	SA134	64k

CoCo Flash Buttons

There are two buttons on the CoCo Flash cartridge. These buttons are used for disabling cartridge auto-start which stops the menu from automatically starting on the CoCo Flash cartridge and performing a full reset which forces the cartridge back to default settings and returns you to the main menu. Disabling auto-start can be useful to prevent the Color Computer from crashing if bank 0 of the CoCo Flash becomes corrupt or invalid. The button closer to the audio jack is the reset to default button and the button closer to the led is the disable auto-start button.

Technical Details

Coco FLASH places an 8 megabyte FLASH ROM into the Color Computer address space at $8000-$ffef. Since the entire 8MB ROM cannot be mapped into the limited address space of the CoCo, a "banking" system is utilized, where the ROM location $8000 (as seen by the Color Computer) can be moved to any 4kB boundary of the FLASH ROM. Additionally, to support multi-bank cartridges like RoboCop, which used a 16kB "offset" register to select various 16kB "pages" of ROM, Coco FLASH emulates this register as well.

Thus, to determine the actual FLASH ROM location that resides in the Coco memory map, one can utilize the following formula:
FLASH ROM location = CoCo Memory Location – $8000 + (bank * 4096) + (page * 16384).

Coco FLASH Registers

To configure Coco FLASH, the unit provides 4 re-locatable registers at $ff64-$ff67. The registers are described as follows:

Address	Name	Function		
		Bit	Write	Read
$ff64	CONFIG	7	Enable FLASH Write at $c000 (1 = enabled)	X
		6	Read Bank or Device ID (1 = Device ID visible)	X
		5	Enable offset register (1 = enabled) (Note 1)	X
		4	Force single reset (1 = selected)	X
		3:2	Select SPI source 00 = none 01 = EEPROM 10 = Secondary SPI channel 11 = Third SPI channel (not currently implemented)	X
		1	Autostart (1 = on)	Left switch (1 = pressed)
		0	LED (1 = on)	Right switch (1 = pressed)
$ff65	BANK_LO	7:0	bits 7:0 of bank register	bits 7:0 of bank register or Device ID
				0 or Device Version
$ff66	BANK_HI	7:4	0	bits 10:8 of bank register or CONFIG jumper values
		3:0	bits 10:8 of bank register (bit 3 is 0)	SPI data in (Note 2)
$ff67	SPI	7:0	SPI data out	

Notes:

1. If the offset bit is enabled, any write to locations from $ff40 to $ff5f will store a new value in the offset register, shown in Table 2.
2. The SPI subsystem operates at main CPU speed. Thus, when writing a value to the SPI data register, one must wait 8 cycles before attempting to read from the register.

Table 2: Offset Register

Address	Name	Function	
		Write	Read
$ff40-$ff5f (Note 1)	OFFSET	16kB offset	X

Notes:

1. Any location utilized by Coco FLASH or the built-in Orchestra 90-CC will not store a new value into the offset register.

In addition, the Tandy Radio Shack Orchestra 90-CC cartridge is emulated within Coco FLASH. The cartridge defaults to a base address of $ff7a, but can be altered through software. The cartridge emulation is enabled by default, but can be disabled.

Orchestra 90-CC Registers

The Tandy/Radio Shack Orchestra 90-CC functionality has been slightly expanded, in that reads from the 2 IO locations will return valid data, as per the following table:

Address	Bit(s)	Value
$ff7a	7:4	Device Family ($1)
	3:0	Device ID ($1)
$ff7b	7:4	Device Version
	3:0	Config Jumpers

Cartridge Configuration Register

Both Coco FLASH and Orchestra 90-CC registers can be relocated under software control. To minimize conflicts and place as few constraints on the remapping system as possible, the system utilizes an unlocking mechanism and commands to reconfigure the cartridge.

To unlock the command system, one must send a special sequence of values to the command register, located at $ff80. The sequence is **$55,$aa,<DEVICE_FAMILY:DEVICE_ID>**. Since one cannot be sure the sequence has not already been started, it is recommended that the program read from $ff80 before sending the command sequence, which will reset the command system.

The device families and IDs are listed below:

FAMILY	ID	Device
$1	$1	Tandy/RS Orchestra 90-CC
$2	$1	Coco FLASH

Once the proper sequence is sent, a program must send a command to execute. Currently, only 1 command is supported:

Command	Value	Data Length	Data Value(s)
Set Base Address	$01	1	low byte of desired base address (setting high bit will hide device from IO memory)

To ensure command functionality works correctly when the device resides in the Tandy/RS Multi-Pak Interface (MPI), the device asserts SLENB on any write access to $ff80, which ensure the MPI will deliver the requested data.